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

Leonardo B. Oliveira*
UNICAMP, Brazil
leob@ic.unicamp.br

Hao C. Wong, M. Bern
PARC, Palo Alto, CA
{hcwong,bern}@parc.com

Ricardo Dahab
UNICAMP, Brazil
rdahab@ic.unicamp.br

A. A. F. Loureiro
UFMG, Brazil
loureiro@dcc.ufmg.br

Abstract

Clustered sensor networks have been shown to increase system throughput, decrease system delay, and save energy. While those with rotating cluster heads, such as LEACH, have also advantages in terms of security, the dynamic nature of their communication makes most existing security solutions inadequate for them. In this paper, we show how random key predistribution, widely studied in the context of flat networks, can be used to secure communication in hierarchical (cluster-based) protocols such as LEACH. To our knowledge, it is the first work that investigates random key predistribution as applied to hierarchical WSNs.

1 Introduction

Wireless sensor networks (WSNs) [6] are emerging as a technology for monitoring different environments of interest and they find applications ranging from battlefield reconnaissance to environmental protection. When embedded in critical applications, WSNs are likely to be attacked [13, 19]. Aside from the well known vulnerabilities due to wireless communication, WSNs lack physical protection and are usually deployed in open, unattended environments, which makes them vulnerable to attacks. It is thus crucial to devise security solutions to these networks.

An important issue one needs to tackle when using cryptographic methods to secure a network is key distribution (KD), which has been intensively studied (e.g., [3–5, 10, 11, 14,17,20,21]) in the context of WSNs. Note, however, that a large number [1] of WSN architectures have been proposed and a KD solution that is well suited to one architecture is likely not to be the best for another, as different network architectures exhibit different communication patterns.

Cluster-based organization (e.g., [8]) has been proposed for ad hoc networks in general and WSNs 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 base stations (BSs). Clustered WSNs were first proposed for various reasons including scalability and energy efficiency. Those with rotating CHs, like LEACH [8], 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, making it harder for an adversary to identify the routing elements and compromise them [13].

Adding security to LEACH-like protocols is challenging, as its dynamic and periodic rearranging of the network's clustering (and changing links) makes KD solutions that provide long-lasting node-to-node trust relationships (to be sure, provided by most existing solutions) inadequate. And even though there is previous work [7] on security for LEACH, it does not address all the problems.

In this paper, we focus on providing efficient security to pairwise node-to-CH 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 present numbers on how the various parameters impact the trade-offs between cost and security. Our main contributions are: 1) to have provided an efficient solution for securing pairwise communications in LEACH; and;2) to have shown how random key predistribution can be used to secure dynamic hierarchical (cluster-based) sensor networks protocols,

To be sure, random key predistribution has been studied profusely [10], but always in the context of flat WSNs. Due to this fact, these studies have not taken into consideration communication patterns of hierarchical (cluster-based) networks and thus cannot be applied, as is, to them. To the best of our knowledge, ours is the first that investigates random key predistribution as applied to hierarchical (cluster-based) WSNs with rotating CHs.

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 LEACH's communications and why existing solutions are inadequate. We present

*Supported by FAPESP under grant 2005/00557-9

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 (Low Energy Adaptive Clustering Hierarchy) [8] was proposed to balance energy drainage among nodes. It assumes that every node can directly reach a BS by transmitting with high enough power. However, to save energy, sensor nodes send their messages to their CHs, which then aggregate the messages, and send the aggregate to the BS. To prevent energy drainage of a restricted set of CHs, LEACH randomly rotates CHs among all nodes in the network, from time to time, thus distributing aggregation- and routing-related energy consumption among all nodes in the network. LEACH thus works in rounds. In each round, it uses a distributed algorithm to elect CHs and dynamically cluster the remaining nodes around the CHs. The resulting clustering structure is used by all sensor-BS communications for the remaining of the round.

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.

The setup consists of three steps. In Step 1 (*advertisement* step), 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 decide to do so broadcast a message (adv) advertising this fact, at a level that can be heard by everyone in the network. To avoid collision, a carrier sense multiple access protocol is used. In Step 2 (*cluster joining* step), the remaining nodes pick a cluster to join based on the largest received signal strength of an adv message, and communicate their intention to join by sending a join_req (join request) message. Once the CHs receive all the join requests, Step 3 (*confirmation* step) 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 all transmitters and receivers are calibrated, balanced and geographically distributed clusters should result.

Once the 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.

Setup phase

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

Steady-state phase

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

The various symbols denote:

$A_i, H, BS:$	An ordinary node, a cluster head, and the base station, respectively
$\mathcal{G}:$	The set of all nodes in the network
$\Rightarrow, \rightarrow:$	Broadcast and unicast, transmissions respectively
$id_X:$	Node X's id
$d_X:$	Sensing report from node X
$\langle id_X, t_X \rangle:$	Node X's id and its time slot t_X in its cluster's transmission schedule
$\mathsf{adv}, \mathsf{join\_req}, \mathsf{sched}:$	String identifiers for message types
$\mathcal{F}:$	Data aggregation function

Figure 1. LEACH protocol

2.2 Security vulnerabilities

Like most routing protocols for WSNs, LEACH is vulnerable to a number of security attacks [13], including jamming, spoofing, replay, etc. However, because it is a cluster-based protocol, relying fundamentally on the CHs for data aggregation and 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.

Note that LEACH is more robust against attacks than most other routing protocols [13]. In contrast to more conventional multihop schemes where nodes around the BS are especially attractive for compromise (because they concentrate all network-to-BS communication flows), CHs in LEACH 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 manipulating 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 KD.

3.1 Why Existing KDs Are Inadequate

There are a number of KD schemes in the security literature [18], 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 [20].

Some KD schemes (e.g., [4, 10, 11, 14, 17, 20, 21]). have been specifically designed for WSNs. While they are well-suited for network organizations they were designed for, they are inadequate for others . These schemes typically assume that a node interacts with a quite static set of neighbors and that most of its neighborhood is discovered right after the deployment. However, clusters in LEACH are formed dynamically (at random) and periodically, which changes interactions among the nodes and requires that any node needs to be ready to join any CH at any time.

For instance, LEAP [20], a proposed scheme based on local distribution of keys among nodes in a neighborhood, is rather efficient for flat networks where nodes interact with a rather static set of neighbors. However, if LEAP were used to secure communication in LEACH, a new KD could be required per round. This not only would be inefficient, but also infeasible, as LEAP relies on a master key that is erased from nodes' memory as soon as the first KD is performed.

In what follows, we discuss the network model assumed in LEACH, and the requirements it sets for key distribution.

3.2 KD 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 pairwise authentication for the remaining (node-to-CH and CH-to-BS) communications.

Symmetric-key authenticated broadcasts for WSNs, both global (μTESLA [16]) and local (LEAP [20]), share the core idea of using a one-way key chain (a sequence of keys $k_1, \ldots k_n$, where k_{i+1} is generated from k_i by applying a one-way hash function $f()$, i.e., $k_{i+1} = f(k_i)$) to achieve authentication. These schemes cannot be applied, as is, to LEACH because: 1) the key chain would require significant storage space in 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. (Each node needs to store one key for every other 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.)

Setup phase

1.1. $H \Rightarrow \mathcal{G}$: $id_{A_i}, id_H, \mathsf{mac}_{k_H}(id_H \mid c_H \mid \mathsf{adv})$

A_i : $\mathrm{store}(id_H)$

BS : if $\mathsf{mac}_{k_H}(id_H \mid c_H \mid \mathsf{adv})$ is valid, $\mathrm{add}(id_H, \mathcal{V})$

1.2. $BS \Rightarrow \mathcal{G}$: $\mathcal{V}, \mathsf{mac}_{k^j}(\mathcal{V})$

1.3. $BS \Rightarrow \mathcal{G}$: k^j

A_i : if $\left(f\left(k^j\right) = k^{j+1}\right)$ and $\left(id_H \in \mathcal{V}\right)$, H is authentic

2. $A_i \rightarrow H$: $id_{A_i}, id_H, \mathsf{join\_req}$

3. $H \Rightarrow \mathcal{G}$: $id_H, (\ldots, \langle id_{A_i}, t_{A_i}\rangle, \ldots), \mathsf{sched}$

Symbols as previously defined, with the following additions:

k_X :	Symmetric key shared by node X and BS
k^j :	j-th key in a one-way key chain;
c_X :	Counter shared by node X and BS
$\mathcal{V}$:	An array of node ids
$f()$:	One-way hash function
$\mathsf{mac}_k(\mathrm{msg})$:	MAC calculated using key k
$\mathrm{store}(id_H)$:	Store id_H for future validation
$\mathrm{add}(id_H, \mathcal{V})$:	Add id_H to $\mathcal{V}$

Figure 2. F-LEACH's setup protocol

Pairwise authentication is also challenging to implement in LEACH, because of KD issues. Given that any node needs to be ready to join any CH (which could be any node in the network), it would need to have shared pairwise keys with every other node in the network. Just like in authenticated broadcast, this is neither practical, nor scalable.

3.3 Existing Work on Securing LEACH

Cryptographic protection for LEACH has been studied before. Ferreira et al. [7] proposed a scheme (henceforth refereed as F-LEACH) where each node has two symmetric keys: a pairwise key shared with the BS; and the last key of a key chain held by the BS, used in authenticated broadcast.

F-LEACH (Fig. 2) implements authentication for CHs' broadcasts in two smaller steps, leveraging 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[1] 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 messages from all CHs; compiles the list of legitimate CHs; and sends the list to the network using the μTESLA [16] broadcast authentication scheme (Steps 1.2 and 1.3). Ordinary nodes now know which of the adv messages 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. The other broadcast by the CHs (Step 3) is authenticated the same way. For clarity of presentation, we do not reproduce the full-blown authenticated version here.

Using only two keys per node, F-LEACH does not manage to provide a complete and efficient solution for node-to-CH authentication. In particular, join_req messages (Step 2) in the setup protocol are not authenticated; and ordinary nodes only share keys with the BS. This means that the CHs are prevented from verifying the sensing reports' MACs and, in turn, have to forward them, which incurs a considerable energy consumption.

In this paper, we propose an alternative, using random key predistribution, to set up keys for securing node-to-CH communication in LEACH. The solution is meant to protect the network from attacks by outsiders, i.e., adversaries that do not have credentials (e.g., keys or certificates) to show that they are members of the network Another rather ordinary trust assumption we make is that BSs are trusted.

4 SecLEACH – Random KD to LEACH

In this section, we first describe the main ideas behind random key predistribution schemes, (Section 4.1), then show how they can be used to secure node-to-CH communications in LEACH (Section 4.2). Note that we use LEACH to be concrete, but our proposal should have a wider applicability, and be easily adaptable to other similar protocols.

4.1 Random Key Predistribution

Random key predistribution for WSNs was first proposed by Eschenauer and Gligor [5], and has since been studied by several research groups [10]. 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.

[1]Note that MAC is often used to stand for medium access control in networking papers. Here, MAC stands for message authentication code.

To bootstrap security using Eschenauer and Gligor's original scheme [5], a network goes through three phases. In the first phase (*key predistribution*), 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 second phase (*shared-key discovery*), which takes place during network setup, all nodes broadcast the ids of the keys on their key rings. Through these broadcasts, a node finds 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* phase, 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.

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.

The initial assignment of key rings to nodes can also be done pseudorandomly [17, 21]. Pseudorandom schemes make both the key predistribution and the shared-key discovery more efficient.

4.2 SecLEACH – Protocol Description

In our solution, we propose to generate a large pool of S keys and their ids prior to network deployment. Each node is then assigned a ring of m keys drawn from the pool pseudorandomly [21], without replacement, as follows. For each node X, we use a pseudorandom function (PRF) to generate its unique id id_x. id_x is then used to seed a pseudorandom number generator (PRNG) of a large enough period to produce a sequence of m numbers. $\mathcal{R}_x$, the set of key ids assigned to X, can then be obtained by mapping each number in the sequence to its correspondent value *modulus s*. Also prior to deployment, for each node is assigned a pairwise key shared with the BS.

The LEACH clustering algorithm can then be run with the following modifications: when a self-elected CH broadcasts its adv message , it includes 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 and a nonce. In Step 2, ordinary nodes A_i compute the set of H's key ids (using the pseudorandom scheme described above), choose the closest CH with whom they share a key $k_{[r]}$, and send it a join_req message, protected by a MAC. The MAC is generated using $k_{[r]}$, and includes the nonce from H's broadcast in Step 1 (to prevent replay attacks), as well as the id r of the key chosen to protect this link (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 message in Step 2. A value computed from the nonce (*nonce*) and the reporting cycle (j) is also included to prevent replay. The CHs can now decrypt the sensing reports they receive, perform data aggregation, and send the aggregate result to the BS (Step 5). The aggregate result is protected using the symmetric key shared between the CH and the BS. For freshness, a counter (shared between the CH and the BS) is included in the MAC value as well.

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 j, the value of the "freshness token" (denoted by "*nonce+j*" 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 F-LEACH (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, the number of orphans will depend on the the size of the key pool, the size of the key ring, and the number of CHs, and will have an impact on the performance of the network.

In Section 5, we show the cost, efficiency, and security of SecLEACH, as well as the tradeoffs when we vary the various parameter values.

4.3 Security Analysis

SecLEACH provides authenticity, integrity, confidentiality, and freshness to node-to-node communications. The message in Step 2, Fig. 3, is encrypted with a key in the key pool; and a successful decryption of this message allows H to conclude that the message originated from a legitimate node in the network. Because the encrypted message includes the nonce from Step 1, H can also conclude that it is not a stale message being replayed. 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 message 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.

Because link keys used for node-to-CH communications are not pairwise in SecLEACH (i.e., a number of other nodes other than the end points of a compromised link may have the key used in the link), the biggest security issue in SecLEACH is likely to be its resiliency against node captures. We discuss this issue in Section 5.2.

5 Evaluation of our scheme

In flat WSNs 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 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 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 Parameters: 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. Once m is set, the choice of S will impact the system in two ways:

1. Its security level:

   Given a (S, m)-network, a network where each node is assigned m keys from a key pool of size S, $\frac{m}{S}$ is the

Setup phase

1. $H \Rightarrow \mathcal{G}:$ $id_H, nonce, \mathsf{adv}$
   $A_i:$ choose r such that $r \in \left(\mathcal{R}_H \cap \mathcal{R}_{A_i}\right)$
2. $A_i \rightarrow H:$ $id_{A_i}, id_H, r, \mathsf{join\_req}, \mathsf{mac}_{k_r}(id_{A_i} \mid id_H \mid r \mid nonce)$
3. $H \Rightarrow \mathcal{G}:$ $id_H, (\ldots, \langle id_{A_i}, t_{A_i}\rangle, \ldots), \mathsf{sched}$

Steady-state phase

4. $A_i \rightarrow H:$ $id_{A_i}, id_H, d_{A_i}, \mathsf{mac}_{k_{[r]}}(id_{A_i} \mid id_H \mid d_{A_i} \mid nonce + j)$
5. $H \rightarrow BS:$ $id_H, id_{BS}, \mathcal{F}\left(\ldots, d_{A_i}, \ldots\right), \mathsf{mac}_{k_H}(\mathcal{F}\left(\ldots, d_{A_i}, \ldots\right) \mid c_H)$

Symbols as previously defined, with the following additions:

$r:$ Id of the keys in the key ring
$\mathcal{R}_X:$ Set of key ids in node X's key ring
$k_{[r]}:$ Symmetric key associated with id r
$j:$ Reporting cycle within the current round

Figure 3. SecLEACH protocol

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:

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

   Note that given a fixed m, the larger the S, the smaller the P_s.

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. 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. 4 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 concrete numbers and the tradeoffs induced by different parameter values, we provide estimates on energy consumption 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 shows 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 ($\mid nonce \mid$ and $\mid r + \mathsf{mac}_{k[r]}(msg) - crc \mid$ bytes,

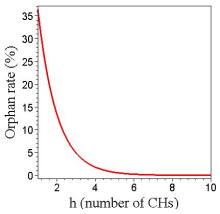

Figure 4. Orphan rate (sl=0.99)

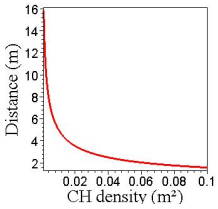

Figure 5. Avg. node-CH distance

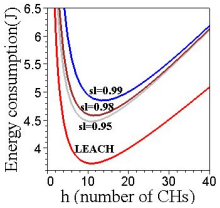

Figure 6. Energy consumpt (m=100)

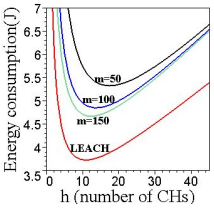

Figure 7. Energy consumpt (sl=0.99)

Fig. 3, Step 1 and 2, respectively), we expected this overhead will be amortized among the cycles of the subsequent steady-state phase. In addition, we do not consider the cost incurred by the crypto operations, as the operations we use have been shown [16] to incur a very small overhead compared to that incurred by communication.

Also, we set SecLEACH messages to be 36 bytes long (the default TinyOS message size [9]) and LEACH messages to be 30 bytes long. The difference is meant to account for the size difference between the MAC (8 bytes [16]) and CRC (2 bytes [12]) – the former present in SecLEACH, but absent in LEACH; and the latter present in LEACH, but absent in SecLEACH.

To estimate the energy consumption, we assume the same radio energy model used in LEACH [8] . In Figs. 6 and 7, the values are for one cycle of sensor data reporting in the steady-state phase. Fig. 6 shows the energy consumption in node-CH communication for different security levels. Note that the consumption level is smaller in LEACH than in any instantiations of SecLEACH, and larger values of sl lead to larger overheads. On the other hand, the higher the h, the smaller the overhead. For a given security level, larger key rings decrease the energy consumption (Fig. 7). Note that, in all cases, there is a value of h for which the energy consumption is minimum.

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. In the estimates, we assume a constant node density (i.e., the larger the n, the larger the network area, as well) and a single BS. The overheads were computed using the values of h for which the energy consumption, in each scenario, is minimum. It is worth mentioning that overhead in SecLEACH is due to two factors: the increased message size (20% larger) and the increased node-CH distance – the CH-BS distance in SecLEACH is not increased as compared to LEACH, as every CH shares a key with the BS.

5.2 Resiliency against node capture

In KD 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.

The resiliency of random key predistribution has been studied before [3] in the context of flat networks. The same analysis is applicable in our context.

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. [16] proposed a suite of efficient symmetric key based security building blocks. Eschenauer et al. [5] looked at random key predistribution schemes (which we discussed in Section 4.1), and originated a large number of follow-on studies (e.g., [3, 4, 10, 11, 14, 17, 21]). Most of the proposed KD schemes, probabilistic or otherwise (e.g., [20]), 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.

Among those specifically targeted to cluster-based sensor networks, Bohge et al. [2] 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. [15] propose solution that relies exclusively on symmetric key schemes and is suitable for networks with an arbitrary number of levels; and Ferreira et al. [7] proposed F-LEACH, which we discussed in Section 3.

sl	m		
	50	**100**	**150**
0.95	–	0.995	–
0.98	–	0.870	–
0.99	0.396	0.636	0.780

Table 1. Prob. P_s of key sharing as a function of security level sl and key ring size m

n	sl (m=100)			m (sl=0.99)		
	0.95	**0.98**	**0.99**	**50**	**100**	**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

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. Finally, we showed concrete numbers for these trade-offs.

References

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

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

[3] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy (S&P'03)*, pages 197–213, may 2003.

[4] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili. A pairwise key pre-distribution scheme for wireless sensor networks. *ACM Transactions on Information and System Security*, 2005. Also appeared in ACM CCS '03.

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

[6] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. In *Mobile Computing and Networking*, pages 263–270, Seattle, WA USA, 1999.

[7] A. C. Ferreira, M. A. Vilaça, L. B. Oliveira, E. Habib, H. C. Wong, and A. A. F. 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.

[8] 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.

[9] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *int'l Conf. on Architectural support for programming languages and operating systems*, pages 93–104, 2000.

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

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

[12] C. Karlof, N. Sastry, and D. Wagner. Tinysec: A link layer security architecture for wireless sensor networks. In *2nd ACM SensSys*, pages 162–175, Nov 2004.

[13] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. *Elsevier's Ad-Hoc Networks Journal, Special Issue on Sensor Network Applications and Protocols*, 1(2–3):293–315, 2003. Also appeared in 1st IEEE International Workshop on Sensor Network Protocols and Applications.

[14] D. Liu, P. Ning, and R. Li. Establishing pairwise keys in distributed sensor networks. *ACM Transactions on Information and System Security (TISSEC)*, 8(1):41–77, 2005. Also appeared in ACM CCS '03.

[15] L. B. Oliveira, H. C. Wong, and A. 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, 2005.

[16] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. *Wireless Networks*, 8(5):521–534, Sept. 2002. Also appeared in MobiCom'01.

[17] R. D. Pietro, L. V. Mancini, and A. Mei. Random key-assignment for secure wireless sensor networks. In *SASN '03: of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 62–71, New York, USA, 2003.

[18] B. Schneier. *Applied Cryptography*. Wiley, 1996.

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

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

[21] 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 Inter'l Conference on Network Protocols (ICNP'03)*, pages 326–335, Atlanta, 2003.