On the Design of Secure Protocols for Hierarchical Sensor Networks

Leonardo B. Oliveira*
University of Campinas (UNICAMP), Brazil
Supported by FAPESP grant 2005/00557-9
E-mail: leob@ic.unicamp.br
*Corresponding author

Hao Chi Wong
Palo Alto Research Center (PARC), CA,
E-mail: hcwong@parc.com

Antonio A. F. Loureiro
Federal University of Minas Gerais (UFMG), Brazil
E-mail: loureiro@dcc.ufmg.br

Ricardo Dahab
University of Campinas (UNICAMP), Brazil
E-mail: rdahab@ic.unicamp.br

AbstractWireless sensor networks (WSNs) are ad hoc networks comprised mainly of small sensor nodes with limited resources, and can be used to monitor areas of interest. In this paper, we propose a solution for securing heterogeneous hierarchical WSNs with an arbitrary number of levels. Our solution relies exclusively on symmetric key schemes, is highly distributed, and takes into account node interaction patterns that are specific to clustered WSNs.

Keywords: security protocols; wireless sensor network security; hierarchical sensor networks; heterogeneous sensor networks

Reference

Biographical notes: Leonardo B. Oliveira received his B.S. (2003) and M.S. (2004) degrees in Computer Science from Federal University of Minas Gerais, Brazil. He is currently pursuing a doctoral degree at University of Campinas, Brazil. Leonardo's primary research interests include security and cryptography in sensor ad hoc networks. Hao Chi Wong is a researcher at Xerox PARC, CA. She holds a PhD from Carnegie Melon University, 2000. She was a visiting professor at Federal University of Minas Gerais in 2003 and 2004. Her main research areas are security and cryptography protocols in wireless sensor networks. Antonio Loureiro is an Associate Professor of Computer Science at the Federal University of Minas Gerais, Brazil. Professor Loureiro holds a PhD in Computer Science from the University of British Columbia, Canada, 1995. His main research areas are mobile computing, computer networks and distributed systems. In the last 10 years he has published over 70 papers in international conferences and journals. Ricardo Dahab is an Associate Professor of the Institute of Computing at the University of Campinas, Brazil. Professor Dahab holds a PhD from the University of Waterloo, Canada, 1993. His main research areas are cryptography, wireless network security, and graph theory.

1 INTRODUCTION

Wireless sensor networks (WSNs) are ad hoc networks comprised mainly of small sensor nodes with limited resources and one or more base stations (BSs), which are much more powerful laptop-class nodes that connect the sensor nodes to the rest of the world (Estrin et al., 1999;

Pottie and Kaiser, 2000). They are used for monitoring purposes, providing information about the area being monitored to the rest of the system. Application areas range from battlefield reconnaissance and emergency rescue operations to surveillance and environmental protection.

Like any wireless ad hoc network, WSNs are vulnerable to attacks (Karlof and Wagner, 2003; Wood and Stankovic, 2002). Besides the well-known vulnerabilities due to wireless communication and ad hocness, WSNs face additional problems. For instance, sensor nodes are small, cheap devices that are unlikely to be made tamper-resistant or tamper-proof. Also, they are often deployed in unprotected, or even hostile areas, which makes them more vulnerable to attacks. It is therefore crucial to add security to these networks, specially those that are part of mission-critical applications.

WSNs may be organized in a variety of different ways, and a solution designed for a flat network will unlikely be optimal for a clustered network. (In Section 2, we briefly survey different sensor network organizations.) To be effective and efficient, a solution needs to be tailored to the particular network organization at hand.

In this paper we present LHA-SP, a suite of secure protocols (setup, operation, and maintenance) for *heterogeneous hierarchical* sensor networks with *arbitrary number of levels*.

We chose to target this class of networks because it has been shown (Melo and Liu, 2002) that, when compared to flat networks, they present a number of advantages including increased system throughput and decreased system delay, and increased energy savings as the number of hierarchy levels in the network is increased.

Our solution considers networks consisting largely of highly resource-constrained nodes, and uses exclusively symmetric key mechanisms. Lightweight group key based mechanisms are used whenever possible; more expensive, BS-mediated schemes are used whenever necessary. Our solution prevents intruders from taking part in network activities, tampering with or injecting messages into the network, as well as eavesdropping on communication between legitimate nodes. It is highly distributed and takes into account node interaction patterns specific to clustered WSNs. To our knowledge, LHA-SP is the first work focusing on securing heterogeneous hierarchical WSNs with arbitrary number of levels.

This paper is organized as follows. In Section 2, we briefly survey existing organizations for hierarchical WSNs, and discuss their vulnerabilities and needed security. In Section 3, we present our network model. In Section 4, we present our solution. We evaluate our solution from the security point of view in Section 5, and from the performance point of view in Section 6. Finally, we discuss related work in Section 7, and conclude in Section 8.

2 HIERARCHICAL WSNs

2.1 Organization

WSNs may be organized in different ways. In *flat* WSNs (Akyildiz et al., 2002), all nodes play similar roles in sensing, data processing, and routing. In *hierarchical* WSNs (Estrin et al., 1999), on the other hand, the network is typically organized into clusters, with ordinary cluster members and the cluster heads (CHs) playing different roles. While ordinary cluster members are responsible for sensing, the CHs are responsible for additional tasks such as collecting and processing the sensing data from their cluster members, and forwarding the results towards the BS.

Hierarchical networks can differ among themselves in various ways. They can be *homogeneous*, when all nodes except the BSs have comparable capabilities; or *heterogeneous*, when some nodes (typically the CHs) are more powerful than others. In two-level networks, CHs are found in the top level, and their children (those that belong to the cluster headed by a CH) in the lower level. In H-level networks ($H > 2$), there is a hierarchy of H nested levels, where CHs in one level are themselves children of nodes that are one level up (Belding-Royer, 2003). CHs can be randomly chosen among the ordinary nodes of a homogeneous network (as in LEACH (Heinzelman et al., 2000)), or they can be more powerful nodes that compose a heterogeneous network (Mhatre et al., 2005). Clustering can also differ from one network to another. For example, under a k-hop clustering (Fernandess and Malkhi, 2002), members of a cluster are all within k-hops of each other. Alternatively, a subset of nodes can probabilistically self-select CHs, and the remaining nodes cluster around the CH that is geographically the closest (Heinzelman et al., 2000).

2.2 Security

Like any WSN, hierarchical WSNs are vulnerable to a number of attacks (Karlof and Wagner, 2003; Wood and Stankovic, 2002) including jamming, spoofing, and replay. In these networks, attacks involving CHs are particularly damaging, because CHs are responsible for critical functions such as data aggregation and routing. If an adversary manages to become a CH, it can stage attacks such as sinkhole (Karlof and Wagner, 2003) and selective forwarding (Marti et al., 2000), thus disrupting potentially large fractions of the network.

Adversaries may leave the routing alone, and try to inject bogus sensor data into the network. Or they may choose to simply eavesdrop on communication between legitimate nodes, obtaining information that is being gathered by the BSs.

At a high level, the main security goals in a hierarchical WSN are: 1) access control, i.e., allow only legitimate nodes to take part in the network (e.g., become CHs and join a cluster); 2) guarantee the authenticity, confidentiality, integrity and freshness of data being passed from one

member of the network to another; and 3) guarantee availability (minimize the impact of attempts of DoS attacks). In this work, we design our solution to meet these goals while enabling data aggregation at intermediate points as sensor reports are sent from a sensor node to a BS;

3 OUR MODEL

We assume heterogeneous networks with two broad classes of nodes. The first class consists of a large number of highly resource-constrained sensing nodes. The second class consists of a smaller number of non-sensing nodes with various levels of resources (e.g., CPU, transmission range, and energy) responsible for data aggregation and routing.

Each node is statically assigned a hierarchy level prior to deployment (based, e.g., on its resource level), with ordinary sensor nodes being assigned level 1. We assume that nodes are deployed with some care, in such a way that level-h nodes always have level-$(h+1)$ nodes within their communication range. Assuming that level-$(h+1)$ nodes are always more powerful than level-h nodes, if a level-$(h+1)$ node A is within a level-h node B's radio range, then B is within A's range.

We use the hierarchy level for clustering. Nodes in one level seek to cluster around the nodes in the next level up in such a way that the network has, at the end of the clustering process, nested clusters where level-h nodes are CHs for level-$(h-1)$ nodes and children of level-$(h+1)$ nodes.

Communication can then be single hop within a cluster, with the children of a cluster communicating directly with its CH. Communication with the BS is multi-hop: a message goes from a node to its CH successively until it reaches the BS. The BS can, however, communicate directly with any member of the network.

A node does not move once deployed, but can become unavailable (e.g., by energy exhaustion). When this happens, its children will try to join another cluster.

This network organization is rather static: a node's hierarchy/resource level determines whether it will be a CH, and the clustering structure formed at the initial setup of the network does not change unless a CH becomes unavailable. Nonetheless, we believe it is a reasonable starting point for investigating security in heterogeneous hierarchical WSNs.

We assume clock-driven networks: sensing reports are sent to one's CH at regular intervals. At each CH, the reports are aggregated, and only the result is passed up. Nodes have local clocks to keep track of elapsed time, for the purposes of evaluating freshness of keys and timing out on certain events. Local clocks do not need to be synchronized.

Attacks to WSNs may come from *outsiders* (those that are not legitimate members of the network) or *insiders* (those that are legitimate members of the network). The solution we propose here is meant to protect the network from attacks by outsiders only. In our model, keys can be compromised through cryptanalysis or node tampering. We assume that an attacker using either approach will succeed only after a non-negligible amount of time t, and the network can be considered secure for t units of time after deployment. We assume that BSs are trusted.

4 LHA-SP

In this section we present LHA-SP, a suite of **s**ecure **p**rotocols for **h**ierarchical **a**d hoc WSNs as modeled in Section 3. Our goal is to address the problems discussed in Section 2 (access control; authenticity, confidentiality, integrity and freshness of communications; and availability). We first give an overview of our solution (Section 4.1), then the protocol details (Section 4.2), and finally the protocol implementation (Section 4.3) .

4.1 Overview

One of the first concerns in setting up a WSN is to allow only legitimate nodes to participate in the network. To implement this access control, various cryptographic solutions (e.g., Perrig et al. (2002); Zhu et al. (2003a); Bohge and Trappe (2003)) have been proposed. None of them is optimized for the type of networks we consider, mainly because of their key distribution schemes. In our model, each node interacts with a restricted set of nodes during the initial setup. Level-h nodes interact only with level-$(h-1)$ and level-$(h+1)$ nodes, and once the clusters are formed, this set is further reduced: a node interacts only with its CH and children. In addition, after the initial configuration, the set of nodes that a given node interacts with will change only when a CH dies and its children seek new CHs. Thus we need keys that allow legitimate nodes to recognize those that are one level up and one level down, as well as keys to protect their communications with their CH and children. Next, we first show why existing key distribution schemes do not adequately solve our problem, and then sketch our solution.

Given that public key mechanisms are inapplicable to WSNs (because of sensor nodes' resource constraints), most existing solutions rely on mechanisms that predistribute symmetric keys. There are basically three general approaches to predistributing the keys: 1) pairwise key sharing between the BS and each of the remaining nodes (e.g., Perrig et al. (2002)); 2) pairwise key sharing between ordinary nodes, which can be complete (e.g., Carman et al. (2000)) or random (e.g., Eschenauer and Gligor (2002)); and 3) a global group keying (e.g., Basagni et al. (2001)).

In the first approach, the BS works as the key distribution center (KDC). This is rather costly in terms of communication, given that all nodes need to contact the BS to obtain keys they need to share with their CHs and children. In addition, the BS is a bottleneck.

In the second approach, two nodes that share a key at deployment have a secure link between them; those that do not, can use these links to set up their own secure links.

This approach is completely distributed, and does not suffer from high communication costs or from having a bottleneck. However, to give a key to each CH-child link, each node would need to be preloaded with a large number of keys (most of them unnecessary), which is quite wasteful in the type of networks we assume.

In the third approach, everyone in the network or in the vicinity shares the same key. This is the best in terms of cost. Each node only stores one or just a few keys, and no additional keys need to be generated or exchanged. However, when a node is compromised, all links secured by the key stored on it are compromised as well.

In this work, we use a hybrid approach. Prior to deployment, each node is preloaded with: an adoption key, a ring of clustering keys, and a pairwise key it shares with the BS only.

The adoption and clustering keys are both group keys used for setting up the network. They are so named because the former is used to adopt nodes, while the latter to cluster around CHs. By using them, nodes in the network organize themselves into clusters and exchange pairwise keys for securing the links between a node and its CH, needed for later network operation. Once the network is set up, the adoption and clustering keys become invalid and are erased from node memory. The other key – shared between the node and the BS – will be used for orphan adoption, which we explain later.

The pairwise CH-Children keys enable hop-by-hop authentication, allowing data aggregation at the CHs. They also increase the network's resilience against attacks, (avoiding a wholesale compromise of the network if a node ever gets compromised).

Sometimes a network needs additional nodes. We handle addition of nodes the way we handle the initial setup, but using new adoption and clustering keys, which are preloaded to the new nodes, as well as propagated to all level-$(h+1)$ nodes (h is the hierarchy level of the new nodes).

When a node becomes orphan, the only trust association between it and the network is the key it shares with the BS. We use this key to get an orphan node back in the network. This key will not only allow the orphan to join a new cluster, but also obtain a shared key between it and its new CH. Note that even though the rejoining process depends on the BS, we assume that only few nodes will become orphans each time, and there will not be resource contention at the BS.

Notation

In the protocol specifications below, we use single capital letters (e.g., A, B) to denote network nodes; calligraphic capital letters (e.g., $\mathcal{G}$) to denote sets in general; $|$ to denote concatenation; $\{m\}_k$ to denote "encryption of m using key k"; and $\mathsf{MAC}(k, m)$ to denote "message authentication code (MAC) of m using key k". $A \rightarrow B: m$ denotes "A sends message m to B in single hop"; $A \rightarrow\!\!\rightarrow B: m$ denotes "A sends message m to B in multiple hops"; and $A \Rightarrow \mathcal{G}: m$ denotes "A broadcasts message m to group $\mathcal{G}$ in a single hop".

4.2 Protocol Description

4.2.1 Key Predistribution

In our scheme, nodes are preloaded with the following information prior to deployment: the node's id, the node's hierarchy level, an adoption key, a ring of clustering keys, a key it shares with the BS, and the current time. CHs also have information about how many keys will be used within a cluster. We explain the need for this parameter later, when we discuss security levels vs. key scopes.

The distribution of the adoption and clustering keys is carried in such a way that, at the end of the protocol, level-h nodes' ring is the set of all level-$(h+1)$ adoption keys. Below, we describe this procedure.

1. For each level-h a distinct ring r_h of distinct clustering keys is computed.
2. The same ring r_h is then assigned to all level-h nodes.
3. For each level-$(h+1)$ node, the adoption key is chosen by picking at random a single key from r_h.

This allows nodes to employ the key they share to authenticate themselves during the adoption procedure. It is worth noting that level-h node memory may not be enough to store a ring as larger as the number of level-$(h+1)$ nodes. In this case, the same adoption key will be used for more than one CH. Actually, as we will see later in Section 5, the size of the rings will dictate the security of the network setup.

4.2.2 Network Setup

The setup phase in LHA-SP consists of clustering and key distribution. They take place in multiple stages, in a top-down fashion. First, level-$(H-1)$ nodes cluster around level-H nodes (H is the highest hierarchy level of any node in the network), and keys that will be pairwise shared between a level-$(H-1)$ node and its level-H CH are generated and distributed. Then the same protocol is carried out between level-$(H-2)$ and level-$(H-1)$ nodes, and successively, until level-1 nodes are clustered around level-2 nodes and the keys for communication between them are set. We describe the protocol executed at each of these stages (Fig. 1) below.

At each stage, level-h nodes broadcast a adoption-ad message looking for level-$(h-1)$ nodes within their radio range (Step 1).

Besides the identity of the broadcasting node, this message includes the hierarchy level, and a MAC along with the identity of the adoption key used to produce the MAC. Therefore, those nodes in one level down know 1) who broadcast the message, 2) that they are the intended recipients, and 3) the key to be used to check the MAC.

Adoption being broadcast by level-h nodes (e.g. A_h, B_h, C_h):

1. $A_h \Rightarrow \mathcal{G}_{h-1}$: $\mathsf{adoption\text{-}ad}, h, id_A, id_{k_i}, \mathsf{MAC}(k_i, id_{k_i} \mid h \mid id_A)$
   $B_h \Rightarrow \mathcal{G}_{h-1}$: $\mathsf{adoption\text{-}ad}, h, id_B, id_{k_j}, \mathsf{MAC}(k_j, id_{k_j} \mid h \mid id_B)$
   $C_h \Rightarrow \mathcal{G}_{h-1}$: $\mathsf{adoption\text{-}ad}, h, id_C, id_{k_j}, \mathsf{MAC}(k_j, id_{k_j} \mid h \mid id_C)$
   ...

Nodes from $\mathcal{G}_{h-1}$ (e.g., M_{h-1}, N_{h-1}, O_{h-1}, P_{h-1}) choose their CHs (e.g., B_h, A_h, C_h) and respond:

2. $M_{h-1} \rightarrow B_h$: $\mathsf{adoption\text{-}req}, id_M, id_B, \mathsf{MAC}(k_j, id_M \mid id_B)$
   $N_{h-1} \rightarrow A_h$: $\mathsf{adoption\text{-}req}, id_N, id_A, \mathsf{MAC}(k_i, id_N \mid id_A)$
   $O_{h-1} \rightarrow C_h$: $\mathsf{adoption\text{-}req}, id_O, id_C, \mathsf{MAC}(k_j, id_O \mid id_C)$
   $P_{h-1} \rightarrow A_h$: $\mathsf{adoption\text{-}req}, id_P, id_A, \mathsf{MAC}(k_i, id_P \mid id_A)$
   ...

Level-h nodes (e.g., A_h) generate and distribute pairwise keys to be shared with each of their children (e.g. N_{h-1}, P_{h-1}):

3. $A_h \rightarrow N_{h-1}$: $\mathsf{send\text{-}key}, id_A, id_N, \{k_{A,N}\}_{k_i}, \mathsf{MAC}(k_i, id_A \mid id_N \mid \{k_{A,N}\}_{k_i})$
   $A_h \rightarrow P_{h-1}$: $\mathsf{send\text{-}key}, id_A, id_P, \{k_{A,P}\}_{k_i}, \mathsf{MAC}(k_i, id_A \mid id_P \mid \{k_{A,P}\}_{k_i})$
   ...

The various symbols denote:

X_h :	a node X from level h
$\mathcal{G}_h$:	the group of all nodes from level h
h :	hierarchy level
id_X :	id of node/key X
k_i, k_j :	adoption keys
$k_{X,Y}$:	pairwise key shared between nodes X and Y

Figure 1: **The setup protocol.**

Level-$(h-1)$ nodes collect multiple advertisements, and use some some criterion to choose their CHs. For example, they could choose the source of the strongest signal in a period of time. Once they choose a CH, they send an adoption-req message to the chosen node (Step 2). This message includes both the ids of the requesting node and of the chosen CH, and is protected with the adoption key of the chosen CH.

Upon receiving an adoption request from a node, the CH generates a symmetric key, and sends it back to the node in a send-key message (Step 3).

Note that all these message includes a MAC produced using adoption/clustering keys. At each step, a node checks the MAC of the received message. The nodes proceed with the protocol only when the check is successful.

The adoption and clustering keys have a preset validity period (as determined by each sensor's local clock) after which they expire and are discarded by each of the nodes. Thus, the setup protocol should be completed before the key expire.

At the end of this protocol (after all $H-1$ stages have been executed), each node will have acquired $c+1$ pairwise keys: one shared with its CH, and the remaining c shared between it and each of its children.

4.2.3 Network Operation

Once the network is set up and the normal operation begins, there will be two types of communications: child-CH communications, which consist mainly of sensing reports and CH-children communications, which consist mainly of network management messages.

In child-CH communications, a child A_h simply produces a MAC and encrypts the message m_A with the key it shares with the CH D_{h+1}. For freshness, a nonce n_A can be added before encryption.

$$A_h \rightarrow D_{h+1} : n_A, \{m_A\}_{k_{A,D}}, \mathsf{MAC}(k_{A,D}, n_A \mid \{m_A\}_{k_{A,D}})$$

At each hop, the CH can check MACs and decrypt messages it received. Thus, the CH examines their content and performs data aggregation before sending the aggregate result forward.

Information that flows the opposite direction, i.e., from the BS to the rest of the WSN, can be destined to a particular node or a subset of the nodes. If the information is destined to a single node, our pairwise keying scheme is completely adequate. In cases where the information is destined to a larger number of nodes, it can be distributed, multi-hop, through the intermediate CHs. Note that whenever a CH needs to send the same information

to several of its children, the best mechanism would be an authenticated broadcast, which cannot be done with our CH-children pairwise keying. However, we can use the pairwise keys to bootstrap the scheme proposed by LEAP (Zhu et al., 2003a), in which broadcasts are authenticated using keys in a hash key chain. Alternately, we can group all the children in a cluster in a few groups, and have each group share a key. E.g., given a cluster with 10 children, there will be 10 keys if we use pairwise keys between the CH and each of its child. There will be 5 keys if each key is shared between the CH and 2 of its children. And one key if all members of the cluster share the same key. The idea is that, when a CH needs to broadcast a message to multiple nodes in the cluster, it can make fewer transmissions: one for each group that shares a key (instead of one for each child). This scheme thus trades security (the scope of a key) with efficiency. In any case, we expect each CH to have a reasonable small number of children (according to Melo and Liu (2002), between 4% and 10% of a network must be composed of CHs for maximum energy efficiency). And given that there are typically few network management messages, it is not impractical for the CHs to deliver these messages to each child separately, encrypted with the key they share.

4.2.4 Network Maintenance

During the lifetime of a network, nodes come and go: existing nodes may depart from the network (e.g., by energy exhaustion) and new nodes may be added. We handle these changes as follows.

Adding New Nodes To securely add new nodes to the network, we follow the general scheme used in the initial deployment. Nodes about to be added are preloaded with the same set of data as in the initial deployment; however, the ring of clustering keys and the clock time will have new values. The ring now is composed of a newly generated set of key (the initial ones have expired), and the time is the current time given by the operator preloading these values. These keys are intended to be the trust association between the nodes being added and those already in the network.

To allow nodes being added to be adopted, a new adoption key and the current time need to be known by all pre-existing level-$(h+1)$ nodes, where h the level of nodes being added. Again, for each level-$(h+1)$ node an adoption key is chosen by picking at random a key from the new ring and the BS can transmit these values using single hop communication to the intended recipients.

Fig. 2 shows the node addition protocol. Unlike the initial setup protocol, here new nodes seeking to join the network advertise their intention through a new-node-ad message (Step 1), which includes the hierarchy level h of the node broadcasting the message. Those at level $h+1$ that hear this broadcast reply with adoption-ad, signaling their intention to adopt. The rest of the protocol is identical to the initial setup protocol (Fig. 1).

Just like before, the new group key expires after a predefined period of time, before which all new nodes should have joined the network.

Orphan adoption We assume that the network provides means for children of a cluster to learn the unavailability of its CH. This can be achieved, e.g., by periodically pinging the CH, or by using mechanisms such as watchdog (Marti et al., 2000).

Whenever a CH becomes unavailable, it is desirable for the orphans to join another cluster. Given that the pairwise key shared between an orphan and the BS is the only trust association shared between the orphan and the network, we use the BS as an authentication authority and KDC. The protocol (Fig. 3) works as follows.

First, the orphan nodes broadcast the orphan-ad message searching for a new CH (Step 1). This message includes the level h of the orphan. Upon receiving an orphan-ad message, candidate CHs (i.e., those one level up) reply with adoption-ad (Step 2). Neither message is protected, given that the communicating parties do not share any keys.

Each orphan then chooses one among all those that sent a reply, and responds with adoption-req (Step 3). This message is authenticated by a MAC, produced with the key the orphan shares with the BS. It is not destined to the chosen CH, but will be included in the following (key-req) message the CH sends to the BS (step 4).

For the key-req message (Step 4), the CH adds its own MAC (produced using the key it shares with the BS) to the MAC from the orphan. It then adds another MAC using the key it shares with its own CH. The former MACs are intended for the BS to verify the originators of the request, whereas the latter offers link level security (and will be replaced at each hop).

After checking the authenticity of both the orphan and the adopting CH, the BS generates a symmetric key and sends it, single hop, to both. The orphan node is now back on the network, and there is a secure communication channel between it and its CH.

4.3 Protocol Implementation

Given the resource-constraints, the protocols specified above need to have efficient implementations. Thus, cryptographic algorithms need to be chosen not only by their security strength, but also by the amount of resource they consume. In this work, we take advantage of the building blocks from SPINS (Perrig et al., 2002), a suite of lightweight symmetric key based security protocols for highly resource-constrained WSNs. We briefly describe these building blocks below.

To save memory, SPINS implements all cryptographic primitives using one single block cipher; RC5 (Rivest, 1995) was chosen because of its small code size and its efficiency. Encryption and decryption in SPINS are stream ciphers obtained from using RC5 in the counter (CTR)

1. $A_h \Rightarrow \mathcal{G}_{h+1}$: $\mathsf{new\text{-}node\text{-}ad}, h$
2. $B_{h+1} \Rightarrow \mathcal{G}_h$: $\mathsf{adoption\text{-}ad}, (h+1), id_B, id_{k_j}, \mathsf{MAC}(k_j, id_{k_j} \mid (h+1) \mid id_B)$
3. $A_h \rightarrow B_{h+1}$: $\mathsf{adoption\text{-}req}, id_A, id_B, \mathsf{MAC}(k_j, id_A \mid id_B)$
4. $B_{h+1} \rightarrow A_h$: $\mathsf{send\text{-}key}, id_B, id_A, \{k_{B,A}\}_{k_j}, \mathsf{MAC}(k_j, id_B \mid id_A \mid \{k_{B,A}\}_{k_j})$

All symbols as previously defined;

Figure 2: **Node addition protocol.**

Node A_h being adopted by node B_{h+1}

1. $A_h \Rightarrow \mathcal{G}_{h+1}$: $\mathsf{orphan\text{-}ad}, h$
2. $B_{h+1} \rightarrow \mathcal{G}_h$: $\mathsf{adoption\text{-}ad}, (h+1), id_B$
3. $A_h \rightarrow B_{h+1}$: $\mathsf{adoption\text{-}req}, m, \mathsf{MAC}(k_{A,S}, m)$
4. $B_{h+1} \rightarrow C_{h+2}$: $\mathsf{key\text{-}req}, m', \mathsf{MAC}(k_{B,S}, m'), \mathsf{MAC}(k_{B,C}, m' \mid \mathsf{MAC}(k_{B,S}, m'))$
5. $C_{h+2} \rightarrow\!\rightarrow S$: $\mathsf{key\text{-}req}, m', \mathsf{MAC}(k_{B,S}, m'), \mathsf{MAC}(k_{C,D}, m' \mid \mathsf{MAC}(k_{B,S}, m'))$

BS S authenticates A and B, and generates $k_{A,B}$

6. $S \rightarrow A_h$: $\mathsf{key\text{-}del}, id_A, n_A, \{k_{A,B}\}_{k_{A,S}}, \mathsf{MAC}(k_{A,S}, id_B \mid n_A \mid \{k_{A,B}\}_{k_{A,S}})$
7. $S \rightarrow B_{h+1}$: $\mathsf{key\text{-}del}, id_B, n_B, \{k_{A,B}\}_{k_{B,S}}, \mathsf{MAC}(k_{B,S}, id_A \mid n_B \mid \{k_{A,B}\}_{k_{B,S}})$

Symbols as previously defined, with the following additions:

$m = id_A \mid id_B \mid n_A$
$m' = m \mid \mathsf{MAC}(k_{A,S}, m) \mid n_B$
n_X : nonce produced by node X

Figure 3: **Orphan adoption protocol.**

mode. Message authentication code (MAC) is implemented using RC5 under the CBC-MAC (des, 1981) mode: the target message is encrypted under CBC mode, and the message authentication code is the output from the last stage. The same MAC function is used to generate pseudo-random numbers (e.g., nonces) needed by the security module. MAC(k, c) produces a sequence of pseudo-random numbers if the value of c is incremented after each generation. Following good security practice, SPINS uses different keys for different cryptographic functions, all of them derived from a master key χ. The MAC function is also used for this derivation. Using different values of p in MAC(χ, p), different computationally secure keys can be derived from the master key. Thus, one can, e.g., derive different keys for encryption and MAC code. Or even different keys for different communication directions; i.e., one key for communications from A to B, and another from B to A.

We use the building blocks described above to implement our protocols. In case of encryption and decryption, a counter value is actually needed in each operation, as they are implemented by RC5 under CTR mode. Because the counter value determines the one-time pad produced by RC5, and one-time pads should not be used twice for security reasons, all encryptions produced using a given key should use different counter values.

In our proposal, counters are dealt with differently depending on the type of keys used. When pairwise keys are used, as e.g. in child-CH communications, counters are not sent between the parties. Instead, they are kept at both ends of the link, and incremented after each encryption (this is the approach used by SPINS). This is feasible because when pairwise link keys are used, the two communicating parties can keep track of counter values that have been used in conjunction with their link key. (The parties can actually get de-synchronized. But they can either try successive increments, or execute simple synchronization protocols to re-synchronize.) When group keys are used, as e.g. in the setup protocol, counters can no longer be synchronized implicitly as above, because not all nodes will hear all transmissions encrypted using the group key, and therefore, would not know which counter has or has not been used. In these cases, we append the counter value being used to each ciphertext. To prevent different nodes from using the same counter, we assign different non-overlapping ranges of values to each node in the network. Each node is expected to start with the smallest value in its range, and successively use increasing values in successive encryptions.

5 SECURITY ANALYSIS

5.1 Network setup

The security of our setup protocol depends on two assumptions: 1) that an adversary will take a certain amount of time to compromise the group key or tamper with a node, and 2) that this amount of time exceeds that required to set up the network.

Under these two assumptions, our protocol guarantees that only the legitimate nodes of the network can become CHs, join a cluster, distribute keys and receive them. This is because all message exchanges in the setup protocol are encrypted with adoption or clustering keys, which are known only by the members of the network.

The pairwise keys generated by level-h nodes and distributed to each of their children are encrypted by an adoption key before they are transmitted. This adoption key is also included in the key ring of clustering keys shared among level-$h-1$ nodes and thus they could potentially eavesdrop on communications intended to some other node, and learn the value of a pairwise key it should not know. However, according to our assumptions, 1) legitimate members of the network would not eavesdrop (misbehave, in general), unless they have been tampered with; and 2) node tampering would take longer than the network setup time. Thus, at the end of the protocol, every legitimate node would have assured its place in the network topology, and each link would have associated with it a pairwise key, known only by the CH that generated it and the child that is its intended recipient.

Note that it is possible for an adversary to capture all this encrypted traffic for later evaluation, after an adoption key is compromised. Using this approach, the adversary can obtain pairwise keys that were encrypted with this key before being exchanged, and use them for eavesdropping or impersonation. The scope of the compromise is limited to clusters whose CHs have employed the key to establish pairwise keys.

This, in turn, depends on the size of the clustering key ring. E.g., let $\|r_h\|$ and $\|h\|$ be the size of the key ring and the number of level-$(h+1)$ nodes, respectively. If $\|r_h\|$ is big enough so that to each level-$(h+1)$ node it was assigned a distinct key, then only one cluster will be compromised. On the other hand, if the $\|r_h\| \leq \|h\|$, the number of compromised clusters will be, on average, $\frac{\|h\|}{\|r_h\|}$

5.2 Network Operation

During the network operation, communication between any node and its CH is secured by the pairwise key they share. This ensures confidentiality and authentication of communication between the two, prevents bogus nodes from tampering with and injecting messages, and allows data aggregation to take place at the CH. Replay of old messages is prevented by the use of nonces (which is actually dispensable, given that each new encryption is produced with a different counter value).

The adoption and clustering keys expire right after the network setup and are not used thereafter. Thus compromise of a single node has limited scope, and would compromise only the links protected by the keys found in the compromised node.

5.3 Network Maintenance

5.3.1 Adding New Nodes

The node addition protocol follows quite closely the initial setup protocol. Thus, the discussion in Section 5.1 applies here. The new adoption and clustering keys, used to bootstrap the operation, are known only to legitimate and interested parties: the ring of clustering keys are preloaded to the nodes being added, and the adoption keys are delivered securely to the relevant CHs by the BS.

5.3.2 Orphan Adoption

The goal of our orphan adoption protocol is to re-insert an orphan (and the subtree rooted at it) securely into the network routing topology, and to provide it with a key to communicate with the rest of the network securely .

Our proposal relies on the BS as an authentication authority and KDC. The BS authenticates both the adoption-req message (step 3, Fig 3) from the orphan and the key-req message (step 4, Fig 3) from the new CH, before it generates and delivers the requested key. Both the requests and the key delivery are protected by the pairwise keys shared between the BS and the nodes. This means that 1) only requests from legitimate members of the network will be processed; and 2) only the orphan and its new CH will learn the value of the new key, which will be used to secure the communication between them.

Note that because the orphans do not share any trust associations (keys) with the nodes that can potentially adopt them, the messages sent in steps 1 and 2 (Fig. 3) are not protected. This is a source of vulnerability. For instance, a bogus node can send a large number of orphan-ad messages to the network, and try to trigger a response to each of its messages, with the intent of consuming the resources of some of the nodes in the network. Another possible attack is for an intruder to impersonate a potential adopter, and send an adoption-ad message (step 2) in response to orphan-ad messages. The intruder can simply quit the protocol here or try to submit a key-req message (step 3). In any case, the orphan will be left waiting for a key that will never come, and the adoption process will never be completed. We can address the first attack by limiting the number of orphan-ad messages a potential CH will handle per period of time. This is reasonable because we assume that only a small number nodes will become orphans at the same time. To handle the second attack, an orphan can set a waiting time, and if it does not hear from the BS before this time expires, it will contact another potential adopter.

6 PERFORMANCE EVALUATION

In this section, we consider the overhead incurred by our protocols, as compared to a stripped down version of the protocols without the security devices. For example, the stripped down version of the setup protocol would consist of steps 1 and 2 (Fig. 1) only, and the messages exchanged in these steps would not be encrypted.

We evaluate the overheads in terms of computation, communication, and storage. Analysis of these metrics will reveal other costs (e.g. energy consumption and delay). We focus on the protocols for setup and network operation. In what follows, n_h denotes the total number of level-h nodes.

6.1 Communication and Computational Overhead

For the setup protocol (Fig. 1), security incurs the following cost:

- Each adoption-ad and adoption-req message transmission incurs one MAC generations, and c additional bytes for counter value and MAC. adoption-ad message sends are executed once by all the nodes in the network, except those at level 1; and adoption-req sends are executed once by all the nodes, except those at the highest level.

- Each adoption-ad and adoption-req message reception incurs one MAC check operation, and reception of c additional bytes for counter value and MAC. Each level-h node receives no more than n_{h+1} adoption-ad messages, and the set of all level-h nodes will receive a total of n_{h-1} adoption-req messages.

- send-key messages are exchanged only in the secure version of the protocol. Each transmission incurs one key and MAC generations, one encryption, and the message itself. Each reception incurs one decryption and MAC check operations, and the message reception itself. The set of all level-h nodes will send a total of n_{h-1} such messages, whereas each node in the network (except those at the highest level) will receive only one such message.

Our setup protocol is quite scalable. The number of interactions between a level-h node A and level-$(h+1)$ nodes is bounded by $n_{(h+1)}$; and that between A and level-$(h-1)$ nodes is bounded by the number of children in the cluster headed by A.

In CH-children communication, the overhead incurred by security will depend on the pattern of these communications with regard to the number of children a CH tries to reach each time. In the best case scenario, the CH has a message destined to a single child. The overhead is then simply one MAC generation at the CH, transmission of this MAC, and one MAC check at the child. (No explicit counter value need to be enclosed here for the same reason as in child-CH communication.) In the worst case scenario, the transmission from the CH is intended to reach all the children in the cluster. In such scenarios, our solution would be expensive. Instead of a broadcast, our solution requires that the CH sends a separate (protected) message to each of the children (respectively, group of children), because of our pairwise (respectively, group) keying scheme. CH-Children communications, however, are used for network management functions, and do no occur frequently. Thus, a high cost is likely to be tolerable.

For Child-CH communication, which occurs the most in a WSN, LHA-SP is quite efficient: it incurs one encryption and MAC generation at the sender, the transmission of the MAC itself, and one decryption and MAC check at the receiver. Note that the cryptographic operations we use have been shown (Perrig et al., 2002) to incur a very small overhead. Note also that unlike in the setup protocol, counters for encryption/decryption do not need to be explicitly enclosed in the messages. Instead, they can be kept at both ends of the communication (and incremented after each operation). Finally, the use of a MAC makes unnecessary the use of a Cyclic Redundancy Check (CRC), required in unsecured protocols to detect errors in messages.

6.2 Storage Overhead

The overhead in terms of space includes code space and RAM space for cryptographic functions and the keys.

To estimate the storage overhead for our network operation protocols (CH-children and child-CH communications), we modified the source code for Surge (Surge, 2004), an application in the TinyOS distribution that allows a node to periodically send data to the BS. We modified it to send and receive (RC5-based) encrypted data, instead of plaintext. We used cryptographic code from TinySec (Karlof et al., 2004).

In Table 1, "noSec" refers to the original Surge application, whereas "sec" refers to our modified Surge with encryption. Note that security incurs only 990 bytes in ROM (of which the motes have a total of 128K bytes) and 164 bytes in RAM (of which the motes have a total of 4K bytes). Storage overhead incurred by the encryption function is therefore negligible.

	MICA2DOT		MICA2	
Mode	noSec	sec	noSec	sec
ROM	15252	16242	15070	16060
RAM	1843	2007	1843	2007

Table 1: RAM and ROM memory for Motes (in bytes)

Regarding the keys, depending on the level of security required in the setup, a significant amount of the node memory may be used to keep adoption and clustering keys in this phase. However, as soon as the setup phase ends,

these keys will be erased and each level-h node, $h > 1$, only needs to keep two pairwise keys, and k keys shared with its children. In the worst case, when each child share a unique key with its CH, k is equal to the size of the cluster. Level-1 nodes have not children, and have to keep just two keys. Therefore the storage cost is most of time $O(k)$ for level-h nodes ($h > 1$), and $O(1)$ for level-1 nodes.

As a whole, we conclude that LHA-SP is efficient and scales gracefully in terms of computation, communication, and storage costs.

7 RELATED WORK

WSNs are a subclass of MANETS, and much work (e.g., Zhou and Haas (1999); Capkun et al. (2003); Hubaux et al. (2001); Capkun and Hubaux (2003); Venkatraman and Agrawal (2002); Hu et al. (2002); Zhang and Lee (2000)) has been proposed for securing MANETS in general. These studies are not applicable to WSNs because they assume laptop- or palmtop-level resources, which are orders of magnitude larger than those available in WSNs. Public key based solutions are such an example.

Among the studies specifically targeted to resource-constrained WSNs, some (Karlof and Wagner, 2003; Wood and Stankovic, 2002) have focused on attacks and vulnerabilities. Wood and Stankovic (Wood and Stankovic, 2002) surveyed a number of denial of service attacks against WSNs, and discussed some possible countermeasures. Karlof and Wagner (Karlof and Wagner, 2003) focused on routing layer attacks, and showed how some of the existing WSN protocols are vulnerable to these attacks.

Of those offering cryptographic solutions, a reasonable number (e.g., Carman et al. (2000); Eschenauer and Gligor (2002); Yea et al. (2004); Zhu et al. (2003a); Chan et al. (2003); Zhu et al. (2003b); Liu and P.Ning (2003); Liu and Ning (2003); Huang et al. (2003); Pietro et al. (2003); Du et al. (2004); Huang et al. (2004); Kannan et al. (2004); Hwang and Kim (2004); Çamtepe and Yener (2004); Liu et al. (2005); Du et al. (2005); Pietro et al. (2005)) have focused on efficient key management schemes without tying them to a particular network organization. We discussed the trade-offs of different key distribution schemes previously in Section 4.1. And recently, the cryptography community in WSNs has been investigating more efficient techniques of public key cryptography. By using Elliptic Curve Cryptography Miller (1986); Koblitz (1987), for example, it has been shown (e.g.,Gura et al. (2004); Malan et al. (2004); Blaß and Zitterbart (2005)) that sensor nodes are indeed able to compute public key operations. However, public key authentication in the context of WSNs is still an open problem, as they cannot afford a conventional public key infrastructure.

Perrig *et al.* (Perrig et al., 2002) offered a solution for flat and homogeneous networks. They proposed SPINS, a symmetric key based protocol suite for providing baseline security (confidentiality, authentication, integrity, freshness) and authenticated broadcast. Their solution uses pairwise key sharing between each of the nodes and the BS. When two ordinary nodes need to communicate securely between them, the BS works as a key distribution center.

Hierarchical WSNs have quite particular organization patterns, and one can take them into account to design tailored solutions. Carman *et al.* (Carman et al., 2000) have suggested using higher powered nodes for key generation and management functions, but did not offer concrete protocols. Kong *et al.* (Kong et al., 2002) and Bohge and Trappe (Bohge and Trappe, 2003) devised solutions for concrete hierarchical and heterogeneous networks. However, they both assume more powerful nodes, and use public key cryptography. More specifically, the former relies on RSA certificates to guarantee authentication. The amount of computation and space resources required by RSA certificates makes this solution infeasible in our context. In addition, it proposes *end-to-end* transport layer security, which prevents data aggregation at intermediary hops. The latter proposes an authentication framework for a concrete 2-tier network organization, in which a middle tier of more powerful nodes were introduced between the BS and the ordinary sensors to carry out authentication functions. However, except for the lowest tier nodes, all other nodes perform public key operations. More recently, Ferreira *et al.* (Ferreira et al., 2005) and Oliveira *et al.* (Oliveira et al., 2006) proposed SLEACH and SecLEACH, respectively. These works rely exclusively on symmetric key schemes, but they are only adequate for LEACH-like hierarchical WSNs protocols.

There has also been some work on detecting misbehaving nodes. E.g., Marti *et al.* (Marti et al., 2000) proposed a watchdog scheme that enables network nodes to detect selective forwarding attacks staged by their next hop neighbors.

Detecting and dealing with bogus data has also been focus of research. Zhu *et al* (Zhu et al., 2004) proposed an interleaved hop-by-hop authentication scheme to prevent injection of false data into sensor networks. The proposal makes sure that the BS can detect a false report when no more than a certain number t of nodes are compromised. Yea *et al* (Yea et al., 2004) proposed SEF, a statistical en-route filtering mechanism for detecting and dropping bogus reports while being forwarded. It allows both the BS and the en-route nodes to detect false data with a certain probability. Przydatek *et al* (Przydatek et al., 2003) proposed SIA, a framework for secure information aggregation in WSNs which makes use of random sampling strategies for allowing an user to infer about the legitimacy of a value.

Other efforts have focused on more specific types of attacks. Hu *et al* (Hu. et al., 2003) studied and offer solutions for wormhole attacks, whereas Newsome *et al* (Newsome et al., 2004) investigated sybil attacks in the context of WSNs. Finally, Deng (Deng et al., 2003) *et al* address secure in-network processing, and propose a collection of mechanisms for delegating trust to aggregators that *a priori* are not trusted by common sensors. The mechanisms address both dissemination and aggregation of data.

8 CONCLUSION

In this paper, we proposed a solution for securing heterogeneous hierarchical WSNs with arbitrary number of levels. Our solution provides security for network setup and reconfiguration, as well as for the normal network operation traffic. Our scheme sets up pairwise keys between a CH and each of its children (or group of children) using lightweight group key based mechanisms whenever possible, falling back on more expensive, BS-mediated mechanisms whenever necessary.

Our solution is highly distributed, takes into account node interaction patterns that are specific to clustered WSNs, and enables data aggregation at CHs.

We also evaluated the overhead incurred by our solution. The results showed that the overhead incurred by our protocols in terms of energy consumption ranges from small to tolerable. We conclude that our solution is practical.

References

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422.

Basagni, S., Herrin, K., Bruschi, D., and Rosti, E. (2001). Secure pebblenets. In *2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 156–163. ACM Press.

Belding-Royer, E. M. (2003). Multi-level hierarchies for scalable ad hoc routing. *Wirel. Netw.*, 9(5):461–478.

Blaß, E.-O. and Zitterbart, M. (2005). Towards Acceptable Public-Key Encryption in Sensor Networks. In *The 2nd Int'l Workshop on Ubiquitous Computing*. ACM SIGMIS.

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

Capkun, S., Buttyan, L., and Hubaux, J. P. (2003). Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 2(1):17.

Capkun, S. and Hubaux, J.-P. (2003). Biss: building secure routing out of an incomplete set of security associations. In *ACM workshop on Wireless security (WISE'03)*, pages 21–29. ACM Press.

Carman, D. W., Kruus, P. S., and Matt, B. J. (2000). Constraints and approaches for distributed sensor network security. Technical report, NAI Labs, The Security Research Division, Network Associates, Inc.

Çamtepe, S. A. and Yener, B. (2004). Combinatorial design of key distribution mechanisms for wireless sensor networks. In *9th European Symposium on Research Computer Security (ESORICS04)*, pages 293–308, Sophia Antipolis, France. Lecture Notes in Computer Science.

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

Deng, J., Han, R., and Mishra, S. (2003). Security support for in-network processing in wireless sensor networks. In *1st ACM workshop on Security of ad hoc and sensor networks (SASN'03)*, pages 83–93. ACM Press.

des (1981). Data encryption standard modes of operation. Federal Information Processing Standards Publication.

Du, W., Deng, J., Han, Y. S., Chen, S., and Varshney, P. (2004). A key management scheme for wireless sensor networks using deployment knowledge. In *Conference of the IEEE Communications Society (INFOCOM'04)*.

Du, W., Deng, J., Han, Y. S., Varshney, P. K., Katz, J., and Khalili, A. (2005). A pairwise key pre-distribution scheme for wireless sensor networks. *ACM Transactions on Information and System Security*. Also appeared in 10th ACM CCS '03.

Eschenauer, L. and Gligor, V. D. (2002). A key management scheme for distributed sensor networks. In *9th ACM conference on Computer and communications security (CCS'03)*, pages 41–47. ACM Press.

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

Fernandess, Y. and Malkhi, D. (2002). K-clustering in wireless ad hoc networks. In *2nd ACM international workshop on Principles of mobile computing*, pages 31–37. ACM Press.

Ferreira, A. C., Vilaça, M. A., Oliveira, L. B., Habib, E., Wong, H. C., and Loureiro, A. A. F. (2005). 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.

Gura, N., Patel, A., Wander, A., Eberle, H., and Shantz, S. C. (2004). Comparing elliptic curve cryptography and rsa on 8-bit cpus. In *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 119–132, Cambridge, MA, USA.

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

Hu, Y.-C., Perrig, A., and Johnson, D. B. (2002). Ariadne:: a secure on-demand routing protocol for ad hoc networks. In *8th annual international conference on Mobile computing and networking*, pages 12–23. ACM Press.

Hu., Y.-C., Perrig, A., and Johnson, D. B. (2003). Packet leashes: A defense against wormhole attacks in wireless sensor networks. In *Conference of the IEEE Communications Society (INFOCOM'03)*.

Huang, D., Mehta, M., Medhi, D., and Harn, L. (2004). 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.

Huang, Q., Cukier, J., Kobayashi, H., Liu, B., and Zhang, J. (2003). 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.

Hubaux, J.-P., Buttyn, L., and Capkun, S. (2001). The quest for security in mobile ad hoc networks. In *2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 146–155. ACM Press.

Hwang, J. and Kim, Y. (2004). 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.

Kannan, R., Ray, L., and Durresi, A. (2004). 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.

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

Karlof, C. and Wagner, D. (2003). 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. Also apeeared in 1st IEEE International Workshop on Sensor Network Protocols and Applications.

Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of computation*, 48:203–209.

Kong, J., Luo, H., Xu, K., Gu, D. L., Gerla, M., and Lu, S. (2002). Adaptive Security for Multi-layer Ad-hoc Networks. *Wireless Communications and Mobile Computing, Wiley Interscience Press*, 2(5):533–547. Special Issue.

Liu, D. and Ning, P. (2003). Location-based pairwise key establishments for static sensor networks. In *1st ACM workshop on Security of ad hoc and sensor networks (SASN03)*, pages 72–82. ACM Press.

Liu, D., Ning, P., and Li, R. (2005). Establishing pairwise keys in distributed sensor networks. *ACM Transactions on Information and System Security (TISSEC)*, 8(1):41–77. Also appeared in 10th ACM CCS '03.

Liu, D. and P.Ning (2003). 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.

Malan, D. J., Welsh, M., and Smith, M. D. (2004). A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In *1st IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON'04)*, Santa Clara, California.

Marti, S., Giuli, T. J., Lai, K., and Baker, M. (2000). Mitigating routing misbehavior in mobile ad hoc networks. In *Mobile Computing and Networking*, pages 255–265.

Melo, E. J. D. and Liu, M. (2002). The effect of organization on energy consumption in wireless sensor networks. In *IEEE Globecom 2002*.

Mhatre, V., C. Rosenberg, D. K., Mazumdar, R., and Shroff, N. (2005). A minimum cost heterogeneous sensor network with a lifetime constraint. *IEEE Transaction on Mobile Computing*, 4(1):4–15.

Miller, V. (1986). Uses of elliptic curves in cryptography, advances in cryptology. In *Crypto'85, Lecture Notes in Computer Science*, volume 218, pages 417–426. Springer-Verlag.

Newsome, J., Shi, R., Song, D., and Perrig, A. (2004). The sybil attack in sensor networks: Analysis and defenses. In *IEEE International Conference on Information Processing in Sensor Networks (IPSN 2004)*.

Oliveira, L. B., Wong, H. C., Bern, M., Dahab, R., and Loureiro, A. A. F. (2006). SecLEACH – a random key distribution solution for securing clustered sensor networks. In *5th IEEE International Symposium on Network Computing and Applications (NCA'06)*, Cambridge,MA,USA. to appear.

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

Pietro, R. D., Mancini, L. V., and Mei, A. (2003). 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.

Pietro, R. D., Mancini, L. V., and Mei, A. (2005). Efficient and resilient key discovery based on pseudo-random key pre-deployment. *Wireless Networks. special issue of*

IEEE WMAN '04 Best Papers. To appear. Also appeared in 4th International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks 2004 (WMAN04).

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

Przydatek, B., Song, D., and Perrig, A. (2003). SIA: Secure information aggregation in sensor networks. In *ACM SenSys 2003*.

Rivest, R. L. (1995). The RC5 encryption algorithm. In Preneel, B., editor, *Fast Software Encryption*, pages 86–96. Springer. (Proceedings Second International Workshop, Dec. 1994, Leuven, Belgium).

Surge (2004). *Getting Started Guide Revision A – Document 7430-0022-04*. Crossbow Technology, Inc., 41 Daggett Dr., San Jose, CA 95134.

Venkatraman, L. and Agrawal, D. P. (2002). A novel authentication scheme for ad hoc networks. In *IEEE Wireless Communications and Networking Conference*, pages 1268–1273.

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

Yea, F., Luo, H., Lu, S., and Zhang, L. (2004). Statistical en-route filtering of injected false data in sensor networks. In *Conference of the IEEE Communications Society (INFOCOM'04)*.

Zhang, Y. and Lee, W. (2000). Intrusion detection in wireless ad-hoc networks. In *6th annual international conference on Mobile computing and networking*, pages 275–283. ACM Press.

Zhou, L. and Haas, Z. J. (1999). Securing ad hoc networks. *IEEE Network*, 13(6):24–30.

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

Zhu, S., Setia, S., and Jajodia, S. (2004). An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks. In *IEEE Symposium on Security and Privacy*, pages 259–271.

Zhu, S., Xu, S., Setia, S., and Jajodia, S. (2003b). 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.