A Security Protocol for Hierarchical Sensor Networks

[*] **Leonardo B. Oliveira , Hao Chi Wong, Antonio A. Loureiro, Daniel M. Barbosa** [†]

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Belo Horizonte, Minas Gerais — Brasil

{leob,hcwong,loureiro,danielmb}@dcc.ufmg.br

Abstract. *Wireless sensor networks are ad hoc networks comprised mainly of small sensor nodes with limited resources, and can be used to monitor areas of interest. Recent work has shown that hierarchical organizations can increase system throughput, decrease system delay, and save energy; and that energy savings increase as we increase the number of hierarchy levels in the network. Further gains can be obtained if heterogeneous nodes are used in the network. In this paper, we propose a solution for securing heterogeneous hierarchical sensor networks with arbitrary number of levels. Our solution relies exclusively on symmetric key schemes, albeit exploiting the fact that there are more powerful nodes in the network, which we assign heavier computation and communication loads. It is highly distributed, takes into account node interaction patterns that are specific to clustered wireless sensor networks, and enables data aggregation at cluster heads.*

1. Introduction

Wireless sensor networks (WSNs) are ad hoc networks comprised mainly of small sensor nodes with limited resources (low power, low bandwidth, and low computational and storage capabilities) and one or more base stations (BSs), which are much more powerful nodes that connect the sensor nodes to the rest of the world. They are used for monitoring purposes, providing information about the area being monitored to the rest of the system. Application areas range from emergency rescue operations to environmental protection.

Broadly speaking, a WSN may have a flat or a hierarchical organization. In *flat WSNs*, all nodes play similar roles in sensing, data processing, and routing. In particular, to lower energy consumption, nodes operate with limited radio transmission range, and are unable to communicate with everyone in the network directly. Communication node-to-BS is thus multi-hop, with ordinary nodes working as routers for each other.

In *hierarchical WSNs* (HSNs) [10], 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.

It has been shown [2, 10, 15] that hierarchical organization is a rather attractive alternative for WSNs. Due in large part to a smaller number of hops between any sensor node and a BS, HSNs increase the system throughput and decrease the system delay. The architecture is also attractive for the purpose of data aggregation. Instead of forwarding every single message it receives, a CH can aggregate multiple sensing reports with duplicate information, and forward a single report. Because communication consumes much more energy than computation [10], data aggregation will enable significant energy savings. It has also been shown that energy savings increase as we increase the number of hierarchy levels in the network.

---

[*] Researcher of the Brazilian National Research Council (CNPq)

[†] This work was partially supported by CNPq – process number 55.2111/2002-3

HSNs can be *homogeneous* [10], when all nodes except the BSs have comparable capabilities; or *heterogeneous* [3], when CHs are more powerful nodes.

It has been shown (e.g [15]) that heterogeneous networks can outperform their homogeneous counterparts significantly in terms of lifetime and degradation of the sensor coverage area, and in terms of data throughput and energy consumption.

Like any wireless ad hoc network, WSNs are vulnerable to attacks (e.g. [13, 20]). Besides the well-known vulnerabilities due to wireless communication and *ad hoc*ness, WSNs face additional problems. For instance, sensor nodes are small, cheap devices that are unlikely to be made tamper-resistant or tamper-proof. Also, these networks are often deployed in open, unprotected, or even hostile areas. This makes them easily accessible to random individuals and malicious parties alike. It is therefore crucial to add security to these networks, specially those that are part of mission-critical applications.

Even though WSNs are a subclass of MANETs (*Mobile Ad-hoc Networks*), they present some distinguishing characteristics in terms of scale, communication pattern, resource level and mobility. Typically WSNs have a much larger scale (tens or hundreds of thousands of nodes are envisioned), the network traffic flow is asymmetric (mainly from sensor nodes to BSs) and the nodes are much more resource-constrained and have low or no mobility. As a result, various types of solutions (including those that are security-related) proposed for MANETs are not applicable to WSNs. New solutions thus need to be developed.

1.1. Contributions

There have been a number of work on securing WSNs (Section 6), but very few specifically tailored to HSNs. Given recent results showing the advantages of HSNs, specially those consisting of heterogeneous nodes, we aim to devise security protocols for this type of networks in this paper. To the best of our knowledge, this is the first work focusing on securing heterogeneous HSNs with arbitrary number of levels. Because we assume that these networks consist largely of highly resource-constrained nodes, we use exclusively symmetric key mechanisms. We do exploit the fact that there are more powerful nodes in the network, which we assign heavier computation and communication loads.

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 provides not only secure initial setup of the network, but also mechanisms for securing later network reconfigurations triggered by addition of new nodes and unavailability of existing ones. Our solution uses lightweight group key based mechanisms whenever possible, falling back on more expensive, BS-mediated schemes 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.

This paper is organized as follows. In Section 2, we briefly survey existing organizations for HSNs, and discuss their vulnerabilities and needed security. In Section 3, we present the network model we assume in this work. In Section 4, we present our solution. In Section 5, we evaluate our solution from both security and energy consumption of view. Finally, we discuss related work in Section 6, and conclude in Section 7.

2. HSNs: Organization and Security

2.1. Organization

There are different ways of organizing HSNs [18]. For example, they can be homogeneous or heterogeneous (Section 1). They can be 2-tier networks with CHs in the top tier and their children (those that belong to the cluster headed by a CH) in the lower tier [10], or they can have a hierarchy of $N > 2$ nested tiers, where CHs in one tier are themselves children of nodes that are one level up [2]. CHs can be chosen in various ways. For example, they can be randomly chosen among the ordinary nodes of a homogeneous network [10], or they can be more powerful nodes that compose

a heterogeneous network [3]. Networks can also be clustered differently. For example, under k-hop clustering [9], clusters are formed such that members of a cluster are all within k-hops of each other. Alternatively [10], a subset of nodes can probabilistically self-select CHs, and the remaining nodes cluster around the CH that is geographically the closest. Routing can also vary from one HSN to another. In [10], for instance, communication is single hop both from children to CHs and from CHs to BSs. Alternatively, communication can be multihop within the clusters (i.e., from children to CH) and among CHs.

2.2. Security

Like any wireless sensor network, HSNs are vulnerable to a number of attacks [13, 20] including jamming, spoofing, replay, etc. But because they rely on CHs for critical functions such as data aggregation and routing, attacks involving CHs are the most damaging. If an adversary manages to become a CH, it can stage attacks such as sinkhole [13] and selective forwarding [14], thus disrupting potentially large fractions of the network. Of course, the adversary may leave the routing alone, and try to inject bogus sensor data into the network. Or it 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 HSN with data aggregation are: 1) access control; i.e., guarantee that only legitimate nodes can take part in the network (become CHs, join a cluster, and have other nodes in the network accept messages they send); 2) guarantee the authenticity, confidentiality, integrity and freshness of data being passed from one member of the network to another; 3) enable data aggregation at intermediate points as sensor reports are sent from a sensor node to a BS; 4) guarantee availability (minimizes the impact of attempts of DOS attacks).

We design our solution to met these goals in this paper.

3. Our Model

We assume heterogeneous networks with three 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 (CPU, radio transmission range, battery life), responsible for data aggregation and routing. Finally, the third class consists of a few BSs – laptop class equipments that will interface the network with the rest of the world. The Mica Motes [11] and Rockwell WINS nodes are examples of the first and second classes of nodes, respectively.

In our model, we assume that each node is statically assigned a hierarchic level prior to deployment (based, e.g., on its resource level), with ordinary sensor nodes being assigned level 1. We further 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-j nodes are always more powerful than level-i nodes (including having a transceiver with longer transmission range) if $j > i$, we can conclude that if a level-$(h+1)$ node (A) is within a level-h node (B)'s radio range, then B is within A's range.

The hierarchy level is used for clustering. We assume a very simple algorithm: 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 the clusters, with the children of a cluster communicating directly with their CH. Communication with the BSs are multi-hop, with the message being transmitted from a node to its CH successively until it reaches the BS. The BSs can, however, communicate directly with any member of the network.

Nodes do not move once deployed, but can become unavailable (by energy exhaustion or any other reason). When this happens, its children will seek 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 good starting point for investigating security in heterogeneous HSNs.

We assume clock-driven (as opposed to event-driven) networks: reports of interesting events are sent to one's CH at regular intervals called epochs. At each CH, reports from the children are processed and aggregated, and only the aggregate information is passed up.

We assume that nodes have local clocks to keep track of elapsed time, for the purposes of evaluating freshness of keys and timing out on certain events. They do not need to be synchronized.

Attacks to WSNs may come from *outsiders* or *insiders*. In networks protected by cryptographic access control mechanisms, outsiders do not have credentials (e.g., keys or certificates) to show that they are members of the network. Insiders are nodes that have these credentials. Insiders may not always be trustworthy, given that they may be otherwise trustworthy nodes that have been compromised, or they may be bogus nodes that have stolen the credentials from a legitimate node of the network. The solution we propose here is meant to protect the network from attacks by outsiders only. In the rest of this paper, we use intruders to mean outside attackers.

In our model, there are two ways through which a key can be compromised: 1) cryptanalysis using messages collected while they are in transit; and 2) node tampering. We assume that an intruder 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. A Suite of Security Protocol for Hierarchical Sensor Networks

In this section we present a suite of security protocols for sensor networks as modeled in Section 3. Our solution aims to address the problems discussed in Section 2.2. We first give an overview of our solution; the details appear subsequently.

4.1. Overview

One of the first concerns in auto-configuring a network after its physical deployment is to allow only legitimate nodes to participate in the process. Various cryptographic solutions have been proposed to implement this access control restriction in the context of both MANETs(e.g. [4, 19]) and WSNs [1, 16]. All these mechanisms are based on sharing of some secret data (cryptographic keys) among the legitimate nodes of the network to bootstrap the process.

Public key mechanisms are inapplicable in the context of WSNs, because of sensor nodes' resource constraints. Thus, most of the solutions rely on symmetric key mechanisms. There are basically three general approaches to predistributing the keys: 1) pairwise key sharing between the BS and each of the remaining nodes [16]; 2) pairwise key sharing between ordinary nodes, which can be complete (e.g. [5]– each node shares a key with each of the remaining nodes) or random [6, 8]; and 3) a global group keying [5, 1]. Under the first approach, when two ordinary nodes want to communicate with each other, the BS functions as the key generation center. After authenticating the two parties, it generates and distributes a pairwise key to the two nodes. Under the second approach, nodes that share a key at deployment have a secure link between them, and can thus communicate securely with each other without any previous work. Those that do not can use these links to set up their own. Under the third approach, everyone in the network shares the same key, and can thus communicate securely with each other. In any of these cases, any node that shares a key with another will be able to authenticate the other as a legitimate member of the network.

Each of these approaches have their advantages and disadvantages depending on the network organization at hand. In the context of HSNs with data aggregation, approach 1) is rather costly in terms of communication, given that all nodes need to contact the BS to authenticate the nodes they come into contact (their CHs and their children) and to obtain the necessary keys for setting up the corresponding secure communication channels. In addition to being costly, the BS is a bottleneck, which makes this approach impractical for HSNs.

Approach 2) is completely distributed, and does not suffer from high communication costs or from having a bottleneck. However, to guarantee a connected network, each node need to be preloaded with a large number of keys (most of them unnecessary), which is quite wasteful in the type of heterogeneous HSNs we assume (Section 3). Remember that in our model each node interacts with a restricted set of nodes during the initial setup. Nodes in a level h only interact with those in levels $h-1$ and $h+1$, and once the clusters are formed, this set is further reduced: a node only interacts 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.

Approach 3) is the best in terms of cost. Each node only stores one key, and no additional keys need to be generated or exchanged. However, networks that depend exclusively on a network-wide group key are known to be vulnerable. When a node is compromised, all links in the network are compromised.

In this work, we use a hybrid approach. Prior to deployment, each node is preloaded with two keys: a network-wide group key and a key that it shares with the BS only. We use the group key only for setting up the network. Using this key, nodes in the network organize themselves into clusters and exchange pairwise keys needed for later network operation, i.e., those for securing the links between a node and its CH. After the network is set up, the group key is discarded. The other key – shared between the node and the BS – will be used for orphan adoption, which we explain later.

The pairwise CH-child keys serve two purposes: they increase the network's resilience against attacks (to avoid a wholesale compromise of the network if a node ever gets compromised) and enable data aggregation at the CHs. The exchange of these keys is secure because we assume that an adversary will take more than t units of time to compromise the group key, and the network would have been set up before then.

Sometimes a network will need additional nodes. We use the scheme sketched above to handle addition of new nodes. The only difference from the initial scenario is the value of the group key. Given the original network-wide group key would have expired, the new nodes are preloaded with a new group key. For the same scheme to work, this new group key needs to be propagated to all level-h nodes, where $h = J + 1$ and J is the hierarchy level of the nodes being added. Note that the number of nodes in the network that need to get this key is fairly small compared to the total number of nodes in the network. Thus, this key propagation would not incur a prohibitively high cost.

During the operation of a network, nodes can become silent for various reasons including energy exhaustion and node capture. When this happens, children of silent nodes need to find other clusters to join. Given that the only trust association an orphan has with the network is the key it shares with the BS, we propose a scheme that relies on this key to get the orphan nodes back in the network. This key will not only allow the orphan to join a new cluster securely, 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 diagrams below, we use single capital letters (e.g., A, B) to denote network nodes; calligraphic capital letters (e.g., $\mathcal{G}$) to denote groups of nodes; and $|$ to denote concatenation. $A \rightarrow B : m$ denotes 'A sends a message m to B in a single hop'; $A \rightarrow\!\!\rightarrow B : m$ denotes 'A sends a message m to B in multiple hops'; and $A \Rightarrow \mathcal{G} : m$ denotes 'A broadcasts a message m to group $\mathcal{G}$'. Finally $\{m\}_K$ denotes 'message m encrypted by key K'.

4.2. Protocol Description

In our scheme, nodes are preloaded with the following information prior to deployment: the node's id, the node's hierarchy level, a network-wide group key, a key it shares with the BS, and the current time.

4.2.1. Network Setup

Adoption advertisement being broadcast by level-h nodes:

1. $A_h \Rightarrow \mathcal{G}_{h-1}$: $\{\textsf{adoption-ad} \mid h \mid id_A\}_{K_\mathcal{G}}$
2. $B_h \Rightarrow \mathcal{G}_{h-1}$: $\{\textsf{adoption-ad} \mid h \mid id_B\}_{K_\mathcal{G}}$

...

Nodes from $\mathcal{G}_{h-1}$ (e.g., M_{h-1}, N_{h-1}) choose their CH and respond:

3. $M_{h-1} \rightarrow B_h$: $\{\textsf{adoption-req} \mid id_M \mid id_B\}_{K_\mathcal{G}}$
4. $N_{h-1} \rightarrow A_h$: $\{\textsf{adoption-req} \mid id_N \mid id_A\}_{K_\mathcal{G}}$
5. $O_{h-1} \rightarrow A_h$: $\{\textsf{adoption-req} \mid id_O \mid id_A\}_{K_\mathcal{G}}$

...

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

6. $A_h \rightarrow N_{h-1}$: $\{\textsf{send-key} \mid id_A \mid id_N \mid K_{A,N}\}_{K_\mathcal{G}}$
7. $A_h \rightarrow O_{h-1}$: $\{\textsf{send-key} \mid id_A \mid id_O \mid K_{A,O}\}_{K_\mathcal{G}}$

...

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 X
$K_\mathcal{G}$:	the group key
$K_{X,Y}$:	pairwise key shared between nodes X and Y

Figure 1: The setup protocol.

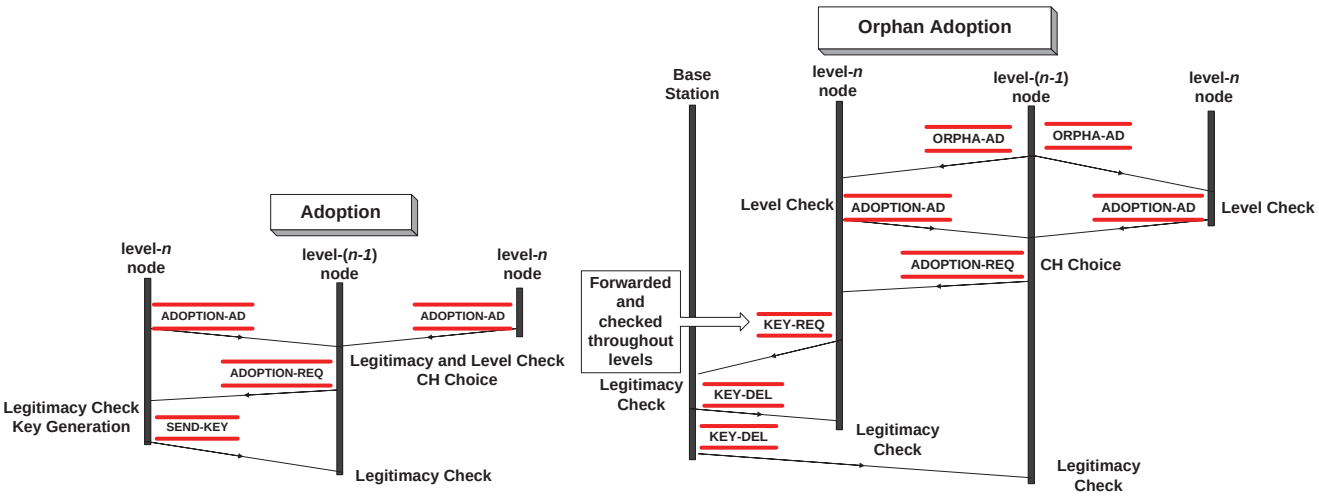
Two things happen during the network setup phase in LHA-SP: clustering and key distribution. They happen in multiple stages, in a top-down fashion. First, level-$(N-1)$ nodes cluster around level-N nodes (N is the highest hierarchy level of any node in the network), and keys that will be shared pairwise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

At each stage, nodes in the higher level, h, broadcast an adoption-ad message looking for level-$(h-1)$ nodes within their radio range (Steps 1 and 2). This message includes the hierarchy level and the identity of the broadcasting node so that the right subset of nodes (those in one level down) know they are the intended recipients and who broadcast the message.

Nodes in level $h-1$ do not simply react to the first broadcast they hear; instead, they collect multiple advertisements, and use a criterion that best matches the current application to choose their CH. For example, they could choose the source of the strongest signal (criterion used by *LEACH* [10]. Once they choose a CH, they send an adoption-req message to the chosen node (Steps 3 – 5). This message includes both the ids of the requesting node and of the chosen CH.

Upon receiving an adoption request from a node X, the CH generates a symmetric key, and sends it back to X in a send-key message (Steps 6 – 7).

Note that all these message exchanges are encrypted using the group key. At each step, a node checks whether a message it received originated from a legitimate network member by decrypt-



(a) Adoption protocol. (b) Orphan adoption protocol.

Figure 2: Adoption and Orphan Adoption procedures

ing it using the same key. Also note that all communications are single-hop.

The group key has a preset validity period after which it expires (key expiration is determined by each sensor's local clock) and is discarded by each of the nodes. Thus, the setup protocol should be completed before the group key expires.

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

Note that we chose to authenticate messages 1 – 5 (Fig. 1) via encryption, instead of a message authentication code (MAC). In single-hop transmissions target nodes can receive the messages as they are are sent (possibly with non-malicious message corruptions), and the messages can be protected using mechanisms such as CRC. Doing so, we save on energy that would be spent to send additional MAC bytes.

4.2.2. Network Operation

Once the network is set up and the normal operation begins, there will be two types of communications: child-CH communication, which consists mainly of sensing reports; and CH-child communication, which consists mainly of network management messages. We address each in turn.

In child-CH communication, a node simply encrypts the message destined to its CH with the key they share (Fig. 3). For freshness, a nonce is added before the message is encrypted. Note that, at each hop, the recipient of a message can decrypt it, examine its content, and carry out data aggregation (e.g., step 3), before sending the aggregated result forward.

In CH-child communication, information that flows the opposite direction, i.e., from the BS down to the rest of the network, can be destined to a particular node or a subset of the nodes.

If the communication is destined to a single node, then our keying scheme (pairwise keying) is completely adequate. In cases where a CH needs to send the same information to all its children, the best mechanism would be authenticated broadcast, which cannot be done with our CH-child pairwise key sharing. However, we expect each CH to have a reasonable small number of children (between 4 % or 10 % of the WSN must be composed of CHs for maximum energy efficiency, according to [15]). And given that network management messages typically will not occur frequently, it is not impractical for the CHs to deliver these messages to each child separately, encrypted with the key they share (Fig. 4).

Note that the purpose of using pairwise key sharing is to limit the scope of a key and the

One hop data transmission:

1. $A_h \rightarrow D_{h+1}$: $\{\mathsf{sensing} \mid n_A \mid m_A\}_{K_{A,D}}$
2. $B_h \rightarrow D_{h+1}$: $\{\mathsf{sensing} \mid n_B \mid m_B\}_{K_{B,D}}$

...

Level-$(h+1)$ nodes transmit aggregate data to its CH:

3. $D_{h+1} \rightarrow G_{h+2}$: $\{\mathsf{sensing} \mid n_D \mid \mathcal{F}(m_A \ldots m_B)\}_{K_{D,G}}$

...

Level-$(H-1)$ nodes transmit aggregate data to the base station S:

4. $H_{H-1} \rightarrow S$: $\{\mathsf{sensing} \mid n_H \mid m_H\}_{K_{H,S}}$

...

5. $J_{H-1} \rightarrow S$: $\{\mathsf{sensing} \mid n_J \mid m_J\}_{K_{J,S}}$

Symbols as previously defined, with the following additions:

n_X : Nonce generated by node X
m_X : Message being sent by node X (may be a sensing report or aggregate data)
$\mathcal{F}$: data aggregation function

Figure 3: Child-CH communication protocol.

CH-Child communication

1. $A_h \rightarrow B_{h-1}$: $\{\mathsf{mngt} \mid n_A \mid m\}_{K_{A,B}}$

...

2. $A_h \rightarrow D_{h-1}$: $\{\mathsf{mngt} \mid n_A \mid m\}_{K_{A,D}}$

Figure 4: CH-Child communication protocol.

extension of network compromise when a single node is compromised. If we are willing to accept lower levels of security, then we can increase the scope of these keys. For instance, instead of having a unique key for each child, several children can share a key. Thus, when a message needs to be disseminated to all the children of a CH, fewer encryptions and transmissions will be needed.

It is true that we can use mechanisms such as μTESLA [16] for authenticated broadcast, but it requires time synchronization.

4.2.3. Network Maintenance

During the lifetime of a network, nodes can come and go: existing nodes may leave 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 group key and the clock time will have new values. The group key is now a newly generated key (the initial one has expired), and the time is the current time given by the operator preloading these values. The new group key is intended to be the trust association between the nodes being added and those pre-existing ones that will potentially adopt them. Thus, both the new group key and the current time also need to be known by all pre-existing level-h nodes, where $h = J + 1$ and J is the hierarchy level of nodes being added. The BS can transmit these values single hop to the

Adding new nodes

1. $A_h \Rightarrow \mathcal{G}_{h+1}$: $\{\textsf{new-node-ad} \mid h\}_{K'_{\mathcal{G}}}$
2. $\mathcal{B}_{h+1} \Rightarrow \mathcal{G}_h$: $\{\textsf{adoption-ad} \mid (h+1) \mid id_B\}_{K'_{\mathcal{G}}}$
3. $A_h \rightarrow B_{h+1}$: $\{\textsf{adoption-req} \mid id_A \mid id_B\}_{K'_{\mathcal{G}}}$

Level-$(h+1)$ nodes (including B_{h+1}) generate and distribute pairwise keys to the new children:

4. $B_{h+1} \rightarrow A_h$: $\{\textsf{send-key} \mid id_B \mid id_A \mid K_{B,A}\}_{K'_{\mathcal{G}}}$

Symbols as previously defined, with the following addition:

$K'_{\mathcal{G}}$: a new group key different from $K_{\mathcal{G}}$

Figure 5: Node addition protocol.

intended recipients.

Fig. 5 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). This message includes the hierarchy level h of the node broadcasting the message. Those nodes at level $h+1$ that hears 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).

Note that all these message exchanges are encrypted using the new group key. At each step, the nodes check whether a received message originated from a legitimate node by decrypting by it using the same key. The nodes proceed with the protocol only when the check is successful.

Just like before, the new group key expires after a predefined period of time, before which all new nodes should have joined the network. Note that this scheme can be used for joining two networks as well.

Orphan adoption

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

1. $A_h \Rightarrow \mathcal{G}_{h+1}$: $\textsf{orphan-ad} \mid h$
2. $B_{h+1} \rightarrow \mathcal{G}_h$: $\textsf{adoption-ad} \mid (h+1) \mid id_B$
3. $A_h \rightarrow B_{h+1}$: $\textsf{adoption-req} \mid m_1, \textsf{MAC}(K_{AS}, m_1)$
4. $B_{h+1} \rightarrow C_{h+2}$: $\{\textsf{key-req} \mid m_1 \mid n_B, \textsf{MAC}(K_{AS}, m_1), \textsf{MAC}(K_{BS}, m_1 \mid n_B)\}_{K_{BC}}$
5. $C_{h+2} \rightarrow\!\rightarrow S$: $\{\textsf{key-req} \mid m_1 \mid n_B, \textsf{MAC}(K_{AS}, m_1), \textsf{MAC}(K_{BS}, m_1 \mid n_B)\}_{K_{CD}}$

The base station S authenticates both A and B, and generates K_{AB}

5. $S \rightarrow A_h$: $\textsf{key-del} \mid \{K_{AB}\}_{K_{SA}}, \textsf{MAC}(K_{SA}, id_B \mid n_A \mid \{K_{AB}\}_{K_{SA}})$
5. $S \rightarrow B_{h+1}$: $\textsf{key-del} \mid \{K_{AB}\}_{K_{SB}}, \textsf{MAC}(K_{SB}, id_A \mid n_B \mid \{K_{AB}\}_{K_{SB}})$

Symbols as previously defined, with the following additions:

$$m_1 \;=\; id_A \mid id_B \mid n_A$$

Figure 6: Orphan adoption protocol.

We assume that the network provides means for children of a cluster to learn the unavailability of its CH. This can be achieved by periodically pinging the CH, by using mechanisms such as

watchdog [14], or through a notice from the BS.

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 propose a scheme in which the BS works as an authentication authority and key distribution center (Figs. 2(b) and 6).

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 between them.

Each orphan then chooses one among all those that sent a reply, and responds with adoption-req (step 3). This message has a component that is encrypted 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).

key-req message is doubly encrypted: the inner encryption uses the key shared between the CH and the BS, whereas the outer encryption uses the key shared between the CH and the CH's own CH. The outer encryption offers link level security (and will be replaced at each hop), whereas the inner encryption is intended for the BS to verify the originator of the request. Note that the BS also needs to verify the originator of the adoption request component of the message.

After checking the authenticity of both the node seeking to be adopted and the adopting node, the BS generates a symmetric key and sends it single hop to the orphan node and its adopting CH. 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

Because sensor nodes are highly resource-constrained, the choice of which cryptographic algorithms to use should be determined not only by the (security) strength of the algorithm, but also by the amount of resource it consumes. In this work, we take advantage of the building blocks from SPINS [16], a suite of lightweight symmetric key based security primitives for highly constrained WSNs. We briefly describe these building blocks here; the original SPINS paper [16] has more details.

To save program memory, SPINS implements all cryptographic primitives from one single block cipher; RC5 [17] was chosen because of its small code size and its efficiency. Encryption and decryption in SPINS are stream ciphers that result from using RC5 in the counter (CTR) mode. The counter value, which is usually enclosed with each encrypted transmission, is kept as part of the local state at both ends of the communication. This practice aims to keep the message short and save on energy spent on transmitting the message. When the communicating parties become desynchronized, a small counter synchronization protocol can be executed. Message authentication code (MAC) is implemented using RC5 under the CBC-MAC [7] mode: the target message is encrypted under CBC mode; and the MAC code is the output from the last stage. Aside from being used to produce authentication codes, the MAC function serves also to generate pseudo-random numbers (e.g., nonces) needed by the security module. The function MAC(K, c) can be used to produce 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. The MAC function is also used for key generation. By using different values of p in MAC(χ, p), different computationally secure keys can be derived from the master key χ.

In what follows, we review each of our protocols in light of concrete cryptographic primitives we use to implement them. In the setup protocol (Fig. 2(a)) and the node addition protocol 5, each of the encryptions actually need a counter value (given that these encryptions are implemented as RC5 operating under CTR mode). Thus, each of the ciphertexts being transmitted has a counter value appended to it.

Because the counter value determines the one-time pad produced by RC5, and one-time pads should not be used twice, no two nodes should use the same counter value. To avoid this, we assign different non-overlapping ranges of values to each node. Each node is expected to start with the smallest value in its range, and successively use increasing values in the range.

In the CH-Child communication (Fig. 4) and the child-CH communication (Fig. 3) protocols, the nonces in the messages are used to guarantee freshness. Given that different one-time pads are used with different transmissions under CTR-mode encryptions, these nonces can be omitted. And unlike in the two protocols above, we keep the counters in each communicating parties' local state here and also in the orphan adoption protocol. This is more manageable for these three protocols, because their communications are pairwise, and counter synchronization between two parties is easier to maintain.

5. Security Analysis and Energy Overhead Estimation

5.1. Security Analysis

5.1.1. 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 a key (in this case, the group key) or tamper with a node, and 2) that this amount of time exceeds the time 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 the group key, which is known only by the members of the network.

The pairwise keys which are generated by the CHs and distributed to each of their children are encrypted by the group key before they are transmitted. This means that any member of the network 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, 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 node that had the group key 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.

5.1.2. Network Operation

The setup protocol has securely replaced a network-wide group key by pairwise keys shared by CHs and each of of their children. 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, but allows data aggregation to take place at the CH. Replay of old messages is prevented by the use of nonces.

The group key expires right after the network setup and is not used thereafter. Thus compromise of a single node has limited scope, and would compromise only the links protected by the keys found in this node.

5.1.3. Network Maintenance

Adding New Nodes: The node addition protocol follows quite closely the initial setup protocol. Thus, all the discussion in Section 5.1.1 applies here. The new group key, used to bootstrap the operation, is known only to legitimate and interested parties: it is preloaded to the nodes being added, and delivered securely to the relevant CHs by the BS.

Orphan Adoption: The node addition protocol follows quite 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 key distribution center. The BS authenticates both the adoption-req message (step 3, Fig 6) from the orphan and the key-req message (step 4, Fig 6) 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 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 have any trust associations (keys) with the nodes that can potentially adopt them, the messages sent in steps 1 and 2 in the protocol (Fig. 6) 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 (as in step 2) in response to orphan-ad messages. The intruder can simply quit the protocol here or try to submit a key-req message (as in 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 re-send the orphan-ad message.

5.2. Energy Consumption Model

In order to evaluate the impact of the proposed scheme regarding energy consumption, we have derived an energy consumption model for our protocol. The objective of this model is to allow a designer that opts for our approach to estimate energy consumption, no matter is their number of level, cluster size, or node architecture. This flexibility is introduced in the model by means of parameters such as: costs to carry out reception (RX), transmission(TX), decryption (D), encryption (E), generation of a key (GK), number of nodes (N), and number of levels (H). Note that because values from a respective parameter can differ for each level depending on the hardware that is being used, it was introduced an index so that these values can be distinguished. For example, RX_i corresponds to the energy necessary for a node from level i to perform a reception. In addition, it is note worthing that some parameter values (D,E,RX,TX) varying with the message size.

To better explain how the model works, we opted for abstracting it as a tree, at which in sensors, number of levels and edges corresponding to, respectively, nodes, height, and procedures of reception and transmission.

Setup: For the purpose of facility we assume no partitioned network, level-(N) nodes only interacting with level-($N+1$) nodes.

First, level-($N+1$) nodes send an encrypted message – $N_{i+1} \cdot (E_{i+1} + TX_{i+1})$. Next, level-($N$) nodes receive the message, choose their CHs, and respond – $N_i \cdot (RX_i + D_i + E_i + TX_i)$. Afterward, CHs receive answers from level-(N) nodes , and generate and distribute pairwise keys – $N_i \cdot (RX_{i+1} + D_{i+1}) + GK_{i+1} + E_{i+1} + TX_{i+1})$. Finally, level-($N$) receive the keys and decrypt the message $(N_i \cdot (RX_i + D_i))$. So, the total setup cost is:

$$\sum_{i=1}^{H-1} N_i \cdot (E_i + TX_i + 2RX_i + 2D_i + RX_{i+1} + D_{i+1} + GK_{i+1} + E_{i+1} + TX_{i+1}) + N_{i+1} \cdot (E_{i+1} + TX_{i+1})$$

Child-CH Communication: In normal operation, each edge of the tree corresponds to an E, TX, RX and D, with the exception of edges that link the level-H nodes to the BS, since the energy

consumption of the BS are not taking into consideration, these edges correspond only to one E and one TX, each.

Therefore, the energy consumption in this flow of communication is given by the following equation:

$$\left(\sum_{i=1}^{H-1} N_i \cdot (E_i + TX_i) + \sum_{i=2}^{H} N_i \cdot (RX_i + D_i) + N_H(E_H + TX_H)\right) \cdot \frac{lifetime}{epoch}$$

CH-Child Communication: Likewise in Child-CH communication, energy consumption can also be modeled from the number of edges here. Now, however, it is only the values RX and D, corresponding to the edges that link the level-H, that are taking into account. The equation is given as follows:

$$N_H(RX_H + D_H) + \sum_{i=2}^{H} N_i \cdot (E_i + TX_i) + \sum_{i=1}^{H-1} N_i \cdot (RX_i + D_i)$$

6. Related Work

WSNs are a special class of MANETS, and much work (e.g, [21, 4, 19]) has been proposed for securing MANETS in general. Most of these works target traditional MANETS, however, and are not applicable for WSNs because they assume laptop- or palmtop-level resources, which are orders of magnitude larger than those available in sensor networks. Public key based solutions are such an example.

A smaller number of works have specifically targeted resource-constrained WSNs. Of these, some [20, 13] have focused on vulnerabilities. Wood and Stankovic [20] surveyed a number of denial of service attacks against WSNs, and discussed some possible countermeasures. Karlof *et al* [13] focused on routing layer attacks. After discussing a number of attacks in general, they showed how some of the existing sensor protocols are vulnerable to these attacks.

Among those offering cryptographic solutions, some [8, 6] have focused on the key management issue without being it to a particular network organization, whereas others [16, 1] assumed flat and homogeneous networks. (We discussed the tradeoffs of different key distribution schemes previously in Section 4.1.). In particular, Perrig *et al* [16] 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. In Pebblenet [1], the network is protected by a a single network-wide group key, which is rekeyed at regular intervals. At each rekeying, one of the "fittest" nodes is chosen to generate the new key, and an efficient algorithm disseminates it among all nodes in the network.

HSNs have quite particular organization patterns, and one can take them into account to design tailored solutions. Carman *et al* [5] have suggested using higher powered nodes for key generation and management functions. More recently, Bohge and Trappe [3] have 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.

There has also been work on detecting misbehaving nodes. Marti *et al* [14] proposed a watchdog scheme that enables network nodes to detect selective forwarding attacks staged by their next hop neighbors. Pires Jr. *et al* [12] considered attacks where a malicious node manipulates its transmission power with the intent of fooling others about its presence or proximity. They then proposed a detection scheme based on the strength of a signal and the geographical position of the signal's originator.

7. Conclusion

In this paper, we proposed a security 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. 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.

We use cryptographic building blocks used by SPINS to implement our protocols, and all the cryptographic primitives are based on RC5.

To estimate energy consumption incurred by our solution, we derived an energy consumption model for it. We plan to estimate the impact of security related operations on the overall energy consumption.

References

[1] Stefano Basagni, Kris Herrin, Danilo Bruschi, and Emilia Rosti. Secure pebblenets. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 156–163. ACM Press, 2001.

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

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

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

[5] David W. Carman, Peter S. Kruus, and Brian J. Matt. Constraints and approaches for distributed sensor network security. Technical report, NAI Labs, The Security Research Division, Network Associates, Inc., 2000.

[6] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy*, May 2003.

[7] Data encryption standard modes of operation. Federal Information Processing Standards Publication, 1981.

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

[9] Yaacov Fernandess and Dahlia Malkhi. K-clustering in wireless ad hoc networks. In *Proceedings of the second ACM international workshop on Principles of mobile computing*, pages 31–37. ACM Press, 2002.

[10] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *IEEE Hawaii Int. Conf. on System Sciences*, pages 4–7, january 2000.

[11] J. Hill and D. Culler. Mica: A wireless platform for deeply embedded networks. *IEEE Micro*, 22(6), 2002.

[12] Waldir Ribeiro Pires Jr., Thiago H. de Paula Figueiredo, Hao Chi Wong, and Antonio A. F. Loureiro. Malicious node detection in wireless sensor networks. In *2004 IEEE International Parallel and Distributed Processing Symposium*, Santa Fe, NM, April 2004.

[13] Chris Karlof and David Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. In *Elsevier's AdHoc Networks Journal, Special Issue on Sensor Network Applications and Protocols*, volume 1, pages 293–315, December 2003.

[14] Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Mobile Computing and Networking*, pages 255–265, 2000.

[15] Enrique J. Duarte Melo and Mingyan Liu. The effect of organization on energy consumption in wireless sensor networks. In *IEEE Globecom 2002*, November 2002.

[16] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. *Wireless Networks*, 8(5):521–534, September 2002.

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

[18] Sameer Tilak, Nael B. Abu-Ghazaleh, and Wendi Heinzelman. A taxonomy of wireless micro-sensor network models. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(2):28–36, 2002.

[19] Lakshmi Venkatraman and Dharma P. Agrawal. A novel authentication scheme for ad hoc networks. In *IEEE Wireless Communications and Networking Conference*, pages 1268–1273, 2002.

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

[21] Lidong Zhou and Zygmunt J. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, 1999.