Dynamic Key Management in Wireless Sensor Networks: A Survey

Xiaobing He*, Michael Niedermeier and Hermann de Meer*

Department of Informatics and Mathematics, University of Passau, 94032, Passau, Germany,
hebing@fim.uni-passau.de, {michael.niedermeier, hermann.demeer}@uni-passau.de

Abstract

Wireless sensor networks (WSNs) have a vast field of applications, including environment monitoring, battlefield surveillance and target tracking systems. As WSNs are usually deployed in remote or even hostile environments and sensor nodes are prone to node compromise attacks, the adoption of dynamic key management is extremely important. However, the resource-constrained nature of sensor nodes hinders the use of dynamic key management solutions designed for wired and ad hoc networks. Hence, many dynamic key management schemes have been proposed for WSNs recently. This paper investigates the special requirements of dynamic key management in sensor network environments, and introduces several basic evaluation metrics. In this work, the state of the art dynamic key management schemes are classified into different groups and summarized based on the evaluation metrics. Finally, several possible future research directions for dynamic key management are provided.

Keywords: Dynamic key management, Wireless sensor networks, Security, Rekeying, Key revocation

1. Introduction

A wireless sensor network (WSN) consists of a large number of sensor nodes, which are powered by batteries, equipped with sensing, data processing and short-range radio communication components [1]. The applications of WSNs range from the most popular ones, like environment monitoring and home automation, to more demanding ones in military or security areas, like battlefield surveillance, targeting and target tracking systems. However, the wireless connectivity, the close interaction among sensor nodes and their unattended operation, as well as the absence of physical protection make WSNs vulnerable to a wide range of network-level attacks and even physical damage [2]. Even though sensor nodes can be equipped with built-in tamper-resistance mechanisms, the memory chips are still suffering from various memory read-out vulnerabilities [3].

Key management is a core mechanism to ensure security in network services and applications of WSNs. *Key management can be defined as a set of processes and mechanisms that support key establishment and the maintenance of ongoing keying relationships between valid parties according to a security policy [4].* Since sensor nodes in WSNs have constraints in their computational power and memory capability, security solutions designed for wired and ad hoc networks are not suitable

*corresponding author
Email address: hebing@fim.uni-passau.de (Xiaobing He)

for WSNs. The goal of key management in WSNs is to solve the problem of creating, distributing and maintaining those secret keys. Hence, techniques for reliable distribution and management of these keys are of vital importance for the security in WSNs.

Due to their importance, the key management systems for WSNs have received increasing attention in scientific literature, and numerous key management schemes have been proposed for WSNs [5, 6, 7, 8, 9]. Depending on the ability to update the cryptographic keys of sensor nodes during their run time (rekeying), these schemes can be classified into two different categories: static and dynamic. In static key management, the principle of key pre-distribution is adopted, and keys are fixed for the whole lifetime of the network. However, as a cryptographic key is used for a long time, its probability of being attacked increases significantly. Instead, in dynamic key management, the cryptographic keys are refreshed throughout the lifetime of the network. Dynamic key management is regarded as a promising key management in sensor networks. *Dynamic key management is a set of processes used to perform rekeying either periodically or on demand as needed by the network.* Since the keys of compromised nodes are revoked in the rekeying process, dynamic key management schemes enhance network survivability and network resilience dramatically.

1.1. Related work and contribution

Although many quality survey papers have been presented in the field of key management of WSNs, the scope of the survey presented in this paper still differs from the existing surveys in many aspects. For the last decade, researchers have started to focus their interest on key management. Numerous review papers including [10, 11, 12, 13, 14] are available, where the authors have examined and surveyed key pre-distribution schemes for key management. Further, [15] classified key management schemes based on attack models, [16] discussed application dependent key management schemes in WSNs, [17] categorized key management schemes into public key schemes, key pre-distribution schemes, dynamic key management and hierarchical key management, [18] organized key management schemes based on different key encryption mechanisms and [19] focused on key management in cluster-based sensor network architecture. However, to the best of our knowledge, no review paper is available where dynamic key management schemes are classified and discussed thoroughly. Considering the importance of dynamic key management in WSNs, a comprehensive survey becomes necessary at this stage.

In this paper, the state of the art dynamic key management schemes are classified into two categories: distributed schemes and centralized schemes, based on whether a central key controller is involved for new key generation or distribution. Then each category is further divided into several subcategories based on the cryptographic primitives and network structures. The purpose of this study is to provide a detailed view of dynamic key management schemes and to identify research directions that can be further pursued.

The rest of this paper is organized as follows: Section 2 introduces the basic requirements and evaluation metrics that should be considered when designing WSN-oriented dynamic key management solutions. Section 3 overviews the existing work in the area of dynamic key management. Section 4 outlines a summary of the surveyed schemes according to given evaluation metrics and points out future research directions in this field. Finally, conclusions are provided in Section 5.

2. Basic requirements and evaluation metrics

Dynamic key management can be considered as a branch of key management. All key management schemes should fulfill the following traditional security requirements: confidentiality, authentication, freshness, integrity and non-repudiation. The same holds for dynamic key management schemes. In addition, according to the features and the application environment of dynamic key management, some particular evaluation measurements are highlighted. Therefore, this section defines the most common metrics used for evaluating dynamic key management techniques in wireless sensor networks. In [13], evaluation metrics for pre-distribution key management are classified as security, efficiency and flexibility, according to the constraints of sensor nodes and networking. Based on this classification and the unique features of dynamic key management, we present the basic requirements and evaluation metrics for dynamic key management. It has to be noted that among the following metrics, *node revocation*, *forward and backward secrecy*, *collusion resistance* and *key connectivity* are only applicable for dynamic key management schemes, while the other metrics can also be applied to static key management schemes.

2.1. Security metrics

Dynamic key management schemes must provide the cryptographic keys in a secure manner, thwarting the activities of malicious nodes inside a network. Upon detecting a compromised sensor node, the current secret key of the compromised sensor node must be revoked and a new one must be generated and distributed to its associated sensor nodes, except the compromised one. Moreover, it is desirable for a dynamic key management scheme to maintain not only forward and backward secrecy, but also collusion resistance between the newly joined nodes and the compromised ones. In addition, resilience against node capture and node replication needs to be provided.

1. *Node revocation.* Once compromised sensor nodes are detected, an effective solution should be able to revoke them promptly from the network. Such mechanisms are useful to prevent a compromised node from deviating the network behavior by injecting false data or modifying data of trusted nodes.

2. *Forward and backward secrecy.* Forward secrecy is used to prevent a node from using an old key to continue decrypting new messages [20]. Backward secrecy is the opposite, it is used to prevent a node with the new key from going backwards in time to decipher previously received messages encrypted with prior keys [20]. Both forward and backward secrecy are used to defeat node capture attacks.

3. *Collusion resistance.* An adversary might attack the network by compromising a number of nodes in the network, making these nodes collude and collaboratively reveal all system keys and consequently capture the entire network. A good dynamic key establishment technique must resist the collusion of newly joined and compromised nodes.

4. *Resilience.* Resilience indicates the resistance against node capture, where the adversary physically attacks a sensor node and tries to recover secret information from its memory. It measures the impact of one captured node on the rest of the network. The resilience of a key management system is high if an adversary cannot affect any node except the captured one. In contrast, the resilience is low if the capture of a single node leads to the compromise of the whole network.

2.2. Efficiency metrics

The number of message transmissions for rekeying, the required number of the cryptographic keys and the amount of operations must be kept as low as possible, meanwhile, the size of the

cryptographic keys should be as short as possible. This prevents the network size from being bounded by the available energy resources and storage capacities of each node. The dynamic key distribution itself shall not put a heavy burden on the inherent resource-constrained sensor nodes in terms of:

1. *Memory.* The amount of memory required to store security credentials, such as keys (e.g., public/private keys, pairwise keys), a "user" certificate (e.g., IDs) and "trusted" certificates (e.g., neighboring nodes' reputation).

2. *Bandwidth.* The number and size of messages exchanged for the key generation process, node replenishment and node eviction.

3. *Energy.* The energy consumption involved in the key agreement process, data transmission and reception, as well as the computational procedure for the generation and distribution of new keys.

2.3. Flexibility metrics

Key establishment techniques should be flexible enough to function well in the wide range of scenarios covered by WSN applications. The most important flexibility metrics are:

1. *Mobility.* Most of the network architectures assume that sensor nodes are stationary (no movement). However, mobility of base stations or sensor nodes or both is necessary in certain applications [21]. Therefore, the key establishment should be able to distribute new keys to moved nodes, allowing them to communicate with their new neighbors. Key generation and distribution for moving nodes is more challenging, since mobility capacity becomes an important issue, in addition to energy and bandwidth.

2. *Scalability.* The number of sensor nodes deployed in the sensing area may in the order of hundreds or even thousands. Moreover, during the whole lifetime of the sensor network, nodes can join or leave. Thus, dynamic key management solutions should be scalable to different network sizes. Meanwhile, the security and efficiency features for small networks are required to be maintained when applied to larger ones.

3. *Key connectivity.* Key connectivity is defined as the probability that two nodes (or more) are able to establish keys after rekeying. Local connectivity considers the connectivity between any pair of neighboring nodes. In contrast, global connectivity refers to the connectivity of the whole network. In order to provide security continuity, high key connectivity after each rekeying process is essential.

3. Dynamic key management schemes in WSNs

This section reviews the state of the art dynamic key management schemes for WSNs. In general, according to whether a central key controller is involved for new key generation or distribution, almost all of the dynamic key management schemes can be classified as either distributed or centralized. On the one hand, according to the different cryptographic primitives that they are based on, the existing distributed dynamic key management can further be classified into three categories, namely, EBS-based, polynomial secret-sharing-based and deterministic sequence-number-based. On the other hand, centralized dynamic key management can also be classified into flat network-based, hierarchical network-based or heterogeneous network-based, according to their network structures.

In this paper, we discuss the major dynamic key management schemes proposed to date for WSNs, and highlight the security and performance properties of each scheme. Figure 1 shows

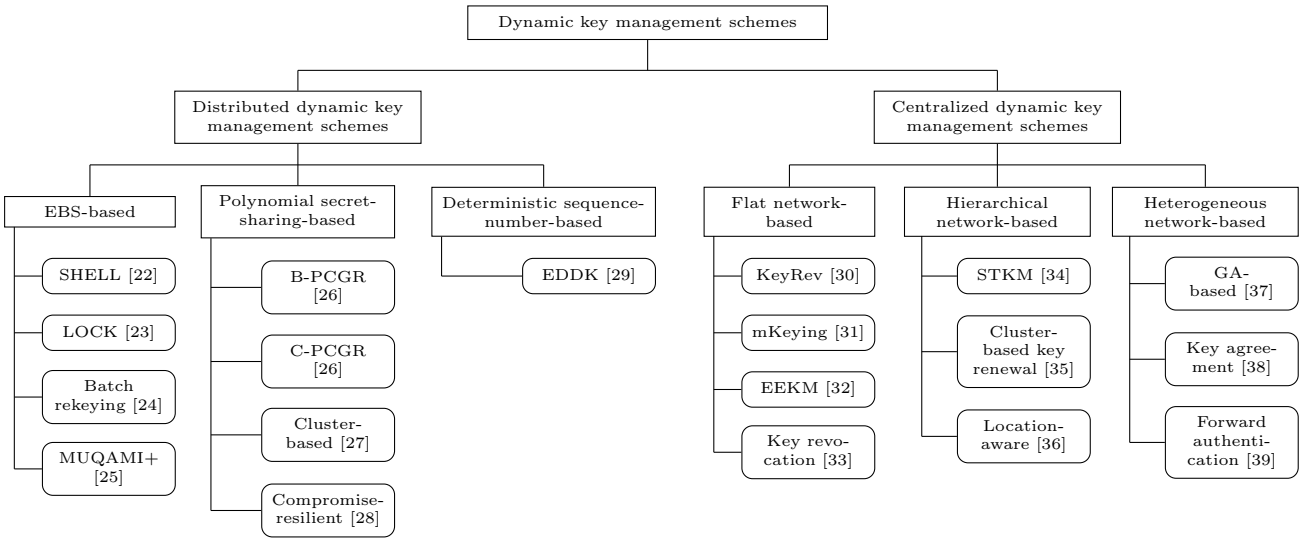


Figure 1: Classification, schemes and references

this classification, as well as a complete list of all the schemes considered in this survey and their references.

3.1. Distributed dynamic key management schemes

Distributed dynamic key management is a set of processes, in which no central key controller, such as a base station or third party, is involved in the rekeying process of sensor nodes. Instead, the key management is handled by multiple key controllers, which can either be predetermined or dynamically assigned.

Distributed dynamic key management schemes avoid a single point of failure and allow for better network scalability. However, they are prone to design errors because compromised sensor nodes can participate in node eviction process. In this subsection, distributed dynamic key management schemes are categorized into EBS-based, polynomial secret-sharing-based and deterministic sequence-number-based, according to different cryptographic primitives they are based on.

3.1.1. EBS-based key update schemes

The Exclusion Basis System (EBS) [40] is a combinatorial formulation of the group key management problem in wireless sensor networks. In EBS-based schemes, each node n_i is assigned k keys out of a pool of size $P = k + m$ ($1 < k, m < n$, where n is the number of sensor nodes in the network). Rekeying is triggered either periodically or once one or more nodes are captured (or suspected to be captured). In the rekeying process, replacement keys are generated, encrypted with all the m keys unknown to the captured nodes and finally distributed to other nodes that collectively know the m keys [40].

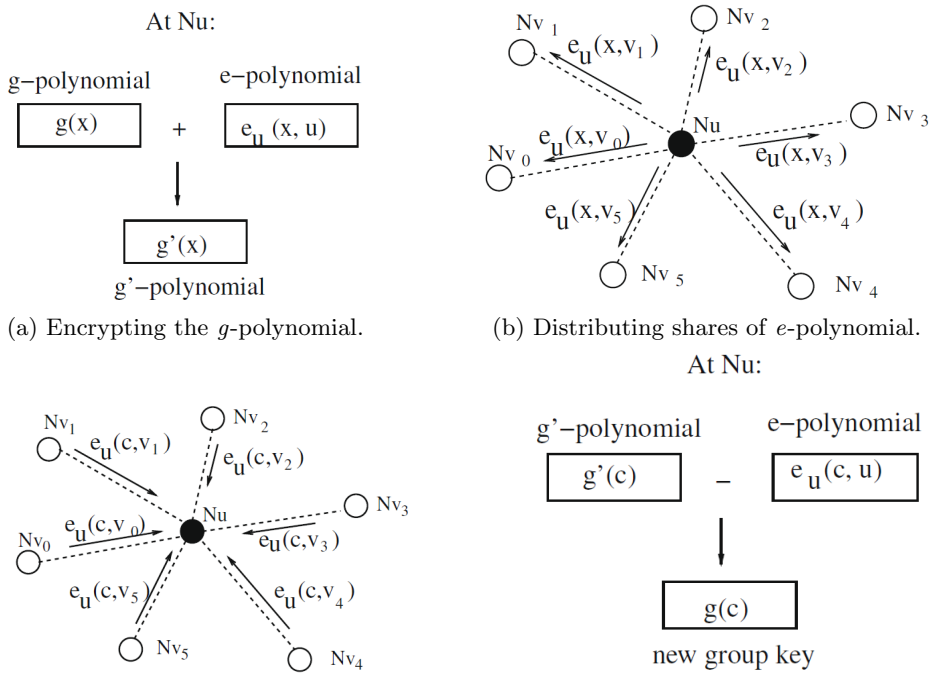
A distributed key management scheme [22], which has the attributes, Scalable, Hierarchical, Efficient, Location-aware and Lightweight (SHELL), is based on EBS. In SHELL, sensor nodes are grouped into clusters C_i, ($0 < i \leq n$, where n is the number of nodes in a cluster) and rekeying only occurs within these clusters. The network considered in SHELL consist of a command node, cluster heads (CHs), gateways and sensor nodes. The command node is assumed to be resource-rich and cannot be compromised. In SHELL, after bootstrapping, the gateway decides the number

of administrative keys needed for a cluster, afterwards, it sends the list of nodes along with the EBS table of key combinations to the command node. The command node designates a number of gateways for each cluster C_i to generate administrative keys. After generation, the key generation gateways encrypt and send these keys to the cluster head $G_{CH}[i]$, which then decrypts these messages and broadcasts their contents to the member nodes of its cluster.

To refresh communication key(s), the cluster head $G_{CH}[i]$ sends the new communication key(s) to its two key-generating gateways, which in turn encrypt them using the administrative keys of the sensor nodes and send the encrypted message to $G_{CH}[i]$. Finally, $G_{CH}[i]$ forwards the new communication key(s) to the nodes of its cluster. The administrative key(s) can also be changed in the similar way. Once a gateway is compromised, its recovery can be handled by either deploying a new gateway or by redistributing the nodes of the compromised gateway's cluster among other uncompromised gateways. If one sensor node in a cluster is compromised or fails to work, the data keys of the whole cluster must be changed. The EBS-based rekeying procedure is applied to evict the compromised sensor nodes. However, the EBS-based key management is prone to collusion attacks [41], which is particularly true when the value of m is selected to be relatively small (to minimize rekeying traffic). SHELL proposed a collusion prevention heuristic key assignment to increase the number of colluding nodes for capturing the whole network. Nodes in a close physical proximity are assigned key combinations with a lower Hamming distance than those far apart. To avoid the assignment of key combinations with a high Hamming distance to two neighboring nodes that share the same parent, swapping of the key combinations is exploited between the node under consideration and any node possesses previously assigned combinations.

Despite the successful rekeying and collusion prevention of SHELL, it has a number of drawbacks. Firstly, its structure and operation are highly complex: heterogeneous node operations and multiple (at least seven) types of keys are used. Due to its complexity, the energy consumption and cryptographic overhead are relatively high compared to other schemes. Secondly, it is supposed that the physical location of the nodes is known, however, the location information is not necessarily available in high density networks. Thirdly, by swapping key combinations, the probability for two sensor nodes that are neighbors and have the same parent to share the same key combinations increases, as these combinations have the minimal Hamming distance. As a result, the number of colluding neighboring nodes increases. Thus, capturing few nodes reveals most of the keys and consequently the entire network can be captured. [42] proposed a Hamming distance based dynamic key management to address this problem. Finally, SHELL relies on a centralized key generation gateway to perform rekeying. Consequently, the capture of key generation gateways provides the adversary more sensor node keys than the capture of a regular sensor node.

LOcalized Combinatorial Keying (LOCK) [23] is also an EBS-based dynamic key management scheme. This WSN model consists of a three level hierarchy, the base station (BS) at the top, followed by cluster leader nodes (CLs), then regular sensor nodes. In LOCK, no location information is used in the generation of new keys. During the initialization phase, sensor nodes of each cluster establish a set of backup keys shared the base station and these keys are unknown to their cluster leader. Once one node is captured, other nodes within the same cluster are rekeyed by a local rekeying mechanism such that the compromised node is unable to communicate with them. If the compromised node is a cluster head, the base station initiates a rekeying at the cluster head level. Similarly, nodes within the cluster governed by the compromised cluster leader are rekeyed with the base station. Unlike any other dynamic key management scheme, in LOCK, the capture of any node (including cluster leader) does not affect the normal operation of other clusters. Since



(a) Encrypting the g-polynomial.

(b) Distributing shares of e-polynomial.

(c) Collecting shares of e-polynomial.

(d) Computing the g-polynomial.

Figure 2: B-PCGR: Polynomial encryption, share distribution and key updating.

nodes are locally rekeyed, LOCK can reduce the time delay and energy consumption used for the cluster key generation and renewal.

Other EBS-based dynamic key management schemes can be found in [24, 25]. [24] proposed a batch rekeying scheme which supports three operations for member nodes: join, leave with collusion-resistant and leave with collusion-free. Compared to SHELL [22], in MUQAMI+ [25], the responsibility of a CH and a key generation node can be shifted seamlessly from one node to another. In addition, the collusion problem of EBS-based dynamic key management schemes was analyzed and a corresponding optimization model was proposed in [43].

3.1.2. Polynomial secret-sharing-based rekeying schemes

PCGR [26], a family of predistribution and local collaboration-based group rekeying was proposed to address the problem of node compromise. In PCGR schemes, sensor nodes are randomly assigned to multiple groups and the nodes in the same group share an unique key. Two schemes, basic PCGR (B-PCGR) and cascading PCGR (C-PCGR) were proposed. In B-PCGR, one-hop neighbors are required to protect their group key polynomials collaboratively. However, if one node u and a certain threshold number of its one-hop neighbors (μ+1, μ is a parameter picked by u) are compromised, the group key polynomial is revealed. C-PCGR, in which the shares of one node are distributed among its multi-hop neighbors, was proposed to address the above mentioned limitation of B-PCGR and achieve higher resilience against node compromise.

In B-PCGR, before the deployment of sensor nodes, a setup server decides the total number of groups and every node is preloaded with the group key polynomial (an unique s-degree univariate g-polynomial) of its group. After neighboring discovery, a sensor node u randomly picks a bivariate e-polynomial (called encryption polynomial, e.g., $e_u(x, y)$) to encrypt its g-polynomial (e.g., $g(x)$),

which results in the encrypted g-polynomial: g'-polynomial (e.g., $g'(x)$). As shown in Figure 2a, the encryption is conducted as $g'(x) = g(x)+e_u(x, y)$. After encryption, as illustrated in Figure 2b, node u distributes the shares of $e_u(x, y)$ to its direct neighbors v_i ($0 \leq i < n$, where n is the number of neighbors). Specifically, each neighbor v_i receives a share $e_u(x, v_i)$ from node u. After the shares distribution process, each node u keeps $g'(x)$ in its memory and removes $e_u(x, y)$ and $g(x)$.

The PCGR is assumed to be loosely time-synchronized and group rekeying is launched by each node periodically. Each node keeps a variable called the current group key version (denoted as c), which is initialized as 0 when the node is deployed. When it is time to update group keys, each innocent node v_i increases its current group key version by one, and returns its share $e_u(c, v_i)$ to each trusted node u, as shown in Figure 2c. Having received μ+1 shares, which are donated as $\{<v_i, e_u(c, v_i)>,\ i = 0, ..., \mu\}$, u constructs an unique μ-degree polynomial, and finally computes the new group key $g(c) = g'(c) - e_u(c, u)$, as shown in Figure 2d.

The C-PCGR was proposed based on B-PCGR, however, it differs from B-PCGR in the encryption and distribution of the polynomial shares and key updating phases. In C-PCGR, the e-polynomial shares of a sensor node u are distributed to its multi-hop neighbors, meanwhile, the e-polynomial shares are distributed/collected in a cascading way. By using a polynomial-based signature and authentication [26], PCGR can effectively detect false shares injected by an adversary.

However, PCGR schemes have five main weaknesses. First, as each node is required to store secret shares of its direct neighbors' e-polynomials, PCGR may not be applicable in networks with a dense deployment of sensor nodes. Second, although PCGR can reduce the amount of traffic and relieve traffic congestions by employing a return probability for its neighbors' share [26], the communication consumption is still too high to be applied to large-scale networks. Third, if a certain number of compromised sensor nodes continuously returns false polynomial shares to a sensor node u, a DoS attack can be launched. Fourth, if a large fraction of a sensor node's neighbors are compromised and new sensor nodes are not deployed around it, this sensor node may not be able to update its group key and thus be isolated. Fifth, in order to allow nodes to send information during key updating process, the previous group key is still used for communication among sensor nodes during a threshold time interval after the rekeying operation. In other words, a revoked node can still use the previous group key to decrypt new messages. Therefore, forward secrecy is not guaranteed.

[27] proposed a cluster-based group key management scheme. Instead of depending on the collaboration of neighboring nodes to acquire a group key, this scheme lets CHs generate and distribute the group key to the nodes within the cluster. The total number of groups is determined by the sink node. For each group, an unique $2t$-degree bivariate polynomial $g(x, y)$ is constructed, and each node (including the CH) u is loaded with its personal secret $g(u, y)$. After deployment, the algorithm proposed in [44] is adopted to help with CH election and cluster formation. Each CH uses an one-way hash function and the identifier (ID) of its member nodes to construct hierarchical keys, which are then unicasted to its corresponding member nodes. When a CH wants to derive the current group key, it broadcasts a request message to the nodes within its cluster. Upon the reception of this request message, each node u willing to trust the CH returns its personal secrets $g(u, y)$, where y is the current group key version. After successfully receiving t personal secrets, CH derives the current group key by following the threshold secret sharing scheme [45]. Afterwards, CH unicasts the current group key to its members using the hierarchical keys.

When a compromised CH is detected, the sink node informs the members of the compromised cluster to begin a new CH election process. If a normal node A is compromised, first, its CH deletes

the hierarchical key shared with it, then, the CH of the compromised node sends a group rekeying message to other CHs in the same group to invoke the group key update process. [27] addressed the problem of node isolation suffering in [26] and [46]. When a CH could not reconstruct the group polynomial, it sends a cluster dissolving message to the sink node and its innocent members. Subsequently, it deletes all its hierarchical keys with its members, and unicasts a joining message (including all the IDs of its innocent members) to other CHs. However, when a new node is added to the network, it may use an obsolete group key, as the new node is not synchronized with the group rekeying process. On the other hand, without time-synchronization, the group key used for communication between two arbitrary nodes may be inconsistent. Moreover, it does not indicate how to select the parameter t appropriately to achieve the desired level of security and reliability. In addition, a normal node under the control of an adversary may return false shares to the CH.

Motivated by the analysis results of Chadha's rekeying protocol [46], [28] proposed a robust pairwise rekeying protocol for hierarchical wireless sensor networks to prevent node capture attacks. The characteristic of the perturbation polynomial [47] is exploited for the proposed rekeying scheme. In [28], the rekeying protocol can be divided into three phases: system initialization, pre-distribution of perturbed polynomials and key establishment and rekeying.

In the system initialization phase, a large prime number q and the minimal integer l which satisfies $2^l > q$ are chosen based on the size of the network. An offline authority arbitrarily constructs a bivariate symmetric polynomial $f(x, y)$. Afterwards, the method described in [47] is used to construct the legitimate ID set S for sensor nodes and generate the perturbation polynomial set Φ. Regarding a positive integer r ($r < l$), any polynomial $\phi(x) \in \Phi$ should satisfy $\forall x \in S, \phi(x) \in 0, ..., 2^r - 1$.

In the pre-distribution of perturbed polynomials phase, before deployment, each cluster head l needs to be loaded with an unique ID $CH_l \in S$ and a perturbed polynomial $\overline{g}_{CH_l}(y) = f(CH_l, y) + \phi_{CH_l}(y) = g_{CH_l}(y) + \phi_{CH_l}(y)$. Similarly, each sensor node (SN) m is also preloaded with an unique ID $SN_m \in S$ and a perturbed polynomial $\overline{g}_{SN_m}(y) = f(SN_m, y) + \phi_{SN_m}(y) = g_{SN_m}(y) + \phi_{SN_m}(y)$. In addition to the unique ID and the perturbed polynomial, all sensor devices (SNs or CHs) are equipped with an one-way hash function $H^k(x)$, the returned hashed value of which is based on the most significant k bits of x.

In [28], the original pairwise key establishment is treated as a rekeying process. After the construction of the hierarchical network, CH_l randomly generates a new t-degree univariate rekeying polynomial function $p_{CH_l}(y)$ at the begin of each rekeying phase. For each of its member sensor nodes SN_m, CH_l updates the corresponding pairwise key $K_{CH_l,SN_m} = H^{l-r}(p_{CH_l}(SN_i))$. At the same time, CH_l uses the newly generated $p_{CH_l}(y)$ and the preloaded polynomial $\overline{g}_{CH_l}(y)$ to construct a master polynomial $w_{CH_l}(y) = p_{CH_l}(y) + \overline{g}_{CH_l}(y)$. Thereafter, the CH broadcasts its ID and the master polynomial $w_{CH_l}(y)$ to all of its member nodes.

After receiving the broadcast message, each SN_m evaluates the preloaded polynomial $\overline{g}_{SN_m}(y)$ and the received $w_{CH_l}(y)$ separately. Upon the evaluation, three candidate keys are calculated [28]. Subsequently, an encoded message $E(msg, K_{CH_l,SN_m})$ is sent using piggybacking from CH_l to SN_m as a unicast message. For SN_m, one of the previously calculated candidate keys that can successfully decode this message will be determined as the new pairwise key.

Compared with traditional polynomial based rekeying protocols, the proposed scheme achieves higher robustness against node capture attacks. Performance analysis results in this work show that their proposed protocol outperforms Chadha's scheme in terms of the total computation complexity and total communication overhead. Compared to Chadha's scheme, the compromised-

resilient pair-wise rekeying scheme achieves both forward and backward secrecy. However, it does not indicate a method to revoke captured/compromised nodes, especially for the CHs. Once a CH is captured, it may launch a revocation attack, resulting in significant traffic congestion and energy depletion. When the pairwise keys are updating, nodes may not be able to send data to their CH, since the pair-wise keys known to them (nodes and their CH) may be inconsistent. Moreover, this scheme does not indicate how to establish pairwise keys between newly deployed nodes and the existing ones.

3.1.3. Deterministic sequence-number-based scheme

After analyzing OTMK [48], which is vulnerable to resource exhausting attacks and DoS attacks, [29] proposed an energy-efficient distributed deterministic key management scheme (EDDK) to securely establish and maintain the pairwise keys and the local cluster key. It assumes each node A, before deployment, is loaded with a network-wide pseudorandom function f and a network-wide initial key K_I. Also, node A has a local cluster key which is shared with all its neighbors. The initial key K_I is used to compute the node's individual key, from which a separate encryption key and a message authentication code (MAC) key are derived. In EDDK, each node not only needs to store the pairwise keys and the local cluster keys but also needs to keep its own public and private keys. In addition, each node also needs to store a neighbor table to maintain the keys (including the pairwise key and the local cluster key) and the sequence numbers. EDDK has three phases: key establishment, data transfer and key maintenance. The last phase includes key update, compromised key revocation, new node joining and mobile node joining.

In the key establishment phase, each node A computes its individual key K_a, generates a sequence number SN_a and broadcasts a network joining message to determine its neighboring nodes. Finally, the pairwise key K_{ab} between neighboring nodes A and B is established with the individual keys (e.g., K_a and K_b) and the random sequence numbers (e.g., SN_a and SN_b): $K_{ab} = f(K_a \oplus K_b, SN_a \oplus SN_b)$.

In the key maintenance phase, the key update process occurs when one or more sensor nodes are compromised or when the lifetime of a pairwise key is reached. The node whose ID is smaller would generate a *pairwise-key* rekeying message and invoke the key update process. When the local cluster key of a node reaches its lifetime, the node broadcasts a similar key update message.

When a new node N wants to set up pairwise keys with its neighbors, it broadcasts a *new-join* message. Upon hearing the broadcasted *new-join* message, each new node W and existing node B verifies the elliptic curve digital signature algorithm (ECDSA) [49] signature of node N. After verifying the legality of node N, for two new nodes, they can follow the key establishment phase to calculate the pairwise key between them. For the pairwise key negotiation between a new node and an existing node, the existing node B broadcasts a response message. Thereafter, with their own private key and the other's public key, both nodes N and B could independently compute the pairwise key $K_{nb} = s_n P_b = s_b P_n$. If a node A moves to a new location, it will broadcast a *rejoin* message. After verification, all of the three kinds of pairwise key (pairwise key between two mobile nodes, pairwise key between a mobile node and a new node and pairwise key between a mobile node and an existing node) can be computed using one's private key and the other's public key.

In EDDK, as long as the ID of a node and the sequence number field are in the neighbor table, replay attacks will fail at the receiver node. The Sybil attacks and node replication attacks will also fail because the attacker does not possess all information (the sequence numbers and the pairwise key) required for message authentication. However, as one sensor node must establish the pairwise keys with all its neighbor and keep a neighbor table, EDDK may not be applicable

in dense networks where each sensor node has many neighbors. Furthermore, since the number of entries in a neighbor table is limited, there may be cases in which a trusted mobile node is unable to establish pairwise keys with its new neighbors and, as a result, cannot join the network. In order to defend against selective forwarding attacks, a promiscuous mode is used for each node to overhear data transmission among its neighboring nodes. However, higher energy consumption is associated with the promiscuous listening mode, which may influence the lifetime of the network.

3.2. Centralized dynamic key management schemes

Centralized dynamic key management is a set of mechanisms that uses a single central key controller, such as a base station or third party, to manage and replace key materials on the network's nodes. This central key controller is globally responsible for the key management.

Compared with distributed dynamic key management, it is impossible for compromised sensor nodes to sabotage the node eviction process. However, in centralized dynamic key management, there can potentially be a large latency in revoking/distributing cryptographic keys due to that messages should be routed multi-hops from the central key controller to the desired sensor nodes; while in distributed dynamic key management, the revocation/distribution of cryptographic keys is much faster due to the fact that messages need to be broadcasted only a few hops to reach the local destination [30].

In this subsection, centralized dynamic key management schemes are classified according to their network structures as flat, hierarchical and heterogeneous. In flat networks, all nodes play the same role or functionality while hierarchical network schemes aiming at constructing different hierarchies (cluster or tree) to reduce data communication for rekeying. Heterogeneous-based schemes utilize nodes with different energy, memory and computational capability to organize the whole network.

3.2.1. Flat network-based schemes

KeyRev [30] is an efficient scheme for removing compromised sensor nodes from WSNs. KeyRev assumes that all the sensor nodes can communicate with the base station directly. Each node maintains four kinds of keys: the pairwise key, the path key, the encryption key and the MAC key. In KeyRev, the lifetime of a WSN is partitioned into sessions. The session key is disseminated regularly by the base station to all the sensor nodes in the network. Both the encryption key and the MAC key are bound to the session key and are renewed with the update of the session key. Consequently, the key revocation problem is reduced to the session key update problem. The session key distribution can be divided into three phases: setup, broadcast and session key recovery.

The setup server randomly picks m $2t$-degree masking polynomial $h_i(x)$ $(i \in 1, 2, .., m)$. Then, each sensor node A_a is loaded with its personal secrets, $\{h_1(a), h_2(a), ..., h_m(a)\}$. The base station also loads the polynomial $h_i(x)$. For each session key K_i, the setup server randomly picks a t-degree polynomial $p_i(x)$ and constructs $q_i(x) = K_i - p_i(x)$.

In the broadcast phase, each node maintains a node revocation list which includes all the compromised sensor nodes' identifiers. For a given set of compromised group members $R = r_1, r_2, ..., r_w$ $(w \leq t)$ in session i, the base station broadcasts the shares of the t-degree polynomial $p_i(x)$ and $q_i(x)$ to non-compromised nodes: $B = R \cup \{P_i(x) = g_i(x)p_i(x) + h_i(x)\} \cup \{Q_i(x) = g_i(x)q_i(x) + h_i(x)\}$, where $g_i(x) = (x - r_1)(x - r_2)...(x - r_w)$.

Upon receiving this broadcast message, a non-compromised sensor node A_a computes $p_i(a)$ and $q_i(a)$, and finally, calculates the new session key $K_i = p_i(a) + q_i(a)$. Since the compromised sensor node r_j $(j \leq t)$ is not able to recover $p_i(j)$ and $q_i(j)$ (because $g_i(j) = 0$), it cannot recover the

new session key, making it impossible to derive the encryption key K_{encr} and the MAC key K_{mac}, resulting in the revocation of the compromised sensor node.

KeyRev is immune to revocation attacks in which a compromised sensor node can launch the session key distribution and update process. In KeyRev, in order to join the network, new sensor nodes must be loaded with pre-distributed key materials (include pairwise key, path key, encryption key and MAC key) and the m personal secrets. KeyRev heavily depends on the accuracy of the compromised nodes detection scheme and the timeliness of the session key update process. Once the session key is not updated in time, the encryption key and the MAC key will be disclosed. Thus, an adversary can make use of the captured keys to inject false data into the network. Moreover, the duration of each session is not determined and discussed, and, KeyRev was simulated and evaluated under ideal condition rather than environments with diverse attack models. In KeyRev, the base station is trustworthy, however, this assumption may not be true in security-critical scenarios. Based on KeyRev, mKeying [31], in which the assumption of the trusted base station can be removed, was proposed to address the problem of revoking both compromised base stations and compromised sensor nodes.

An Energy-Efficient Key Management protocol (EEKM) [32] was proposed for large scale WSNs. In EEKM, it is assumed that the base station can broadcast a message to all sensor nodes and the base station cannot be compromised. Sensor nodes are divided into different virtual and regional groups. Each node is preloaded with an initial master key (IK) to generate its group and pairwise keys. EEKM is a regional group-oriented rekeying strategy. In EEKM, different kinds of keys are used, and all of them, except the secret key shared with the base station, are derived from the initial master keys.

[33] presented a key revocation protocol for WSNs that follows the group communication paradigm. This scheme makes use of a key service to revoke compromised nodes from the group. The key service maintains a revocation-tree to remove a compromised node B. A sensor node A stores a key-set $KeyRing(A)$, which is a subset of the current keys associated with the internal nodes lying on the path from the root to the leaf associated with A. Once one internal sensor node B along the path is compromised, all keys in the $KeyRing(A)$ must be renewed. This scheme improves the protocol scalability and increases the network lifetime by reducing the number and the size of rekeying messages. However, the one-way functions used to authenticate rekeying messages share the same deficiencies with μTESLA [50], such as delayed authentication, supporting only limited number of broadcast senders and vulnerability against DoS attacks.

3.2.2. Hierarchical network-based schemes

In Spanning Tree Key Management (STKM) [34], a tree is built for rekeying. Each sensor node A is loaded with three keys: $K_{a,BS}$, $K_{BS,a}$ and K_r. $K_{a,BS}$ and $K_{BS,a}$ are shared with the base station to encrypt/decrypt the messages sent by A and BS separately. K_r is shared by all nodes of the network. The rekeying process is launched by the base station periodically to refresh K_r.

To construct the spanning tree, a *Hello* message, initiated by the base station, is broadcasted within the network until all nodes join the tree. Once a node A suspects a neighboring node (son node or father node) to be malicious, it sends a *REFRESH-REQ* message to its father on the tree. This message goes upward until it reaches the BS, which finally broadcasts a *REFRESH* message within the tree. When a son node of the BS receives the *REFRESH* message, it replaces the key K_r by the new one, and then encrypts and forwards the *REFRESH* message of the BS to its sons. This way, the *REFRESH* message goes downward within the tree till reaching all the sensor nodes. Finally, every sensor node gets a new global key.

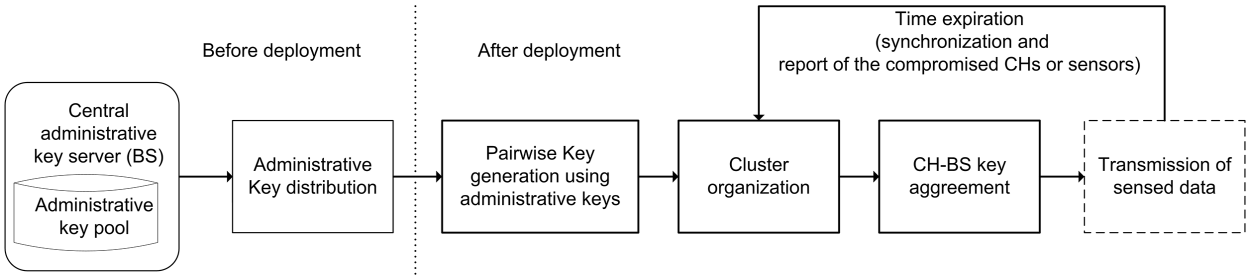


Figure 3: Procedure of cluster-based key renewal (Source [35]).

When a new node N wants to join the network, it broadcasts a *JOIN* message. Upon the reception of the *JOIN* message, each node A in the transmission range of the new node generates and broadcasts an acknowledge message. The new node sends out a *Father* message to declare the sender of his first received acknowledge message as his father. Afterwards, the father node adds the newly deployed node in its sons list, and the surrounding nodes that heard this *Father* message add the new deployed node to their neighbors list in the spanning tree.

In STKM, the storage overhead is acceptable for current sensor nodes, as each node only needs to keep two keys with the base station, one global key and d' (the number of sons) keys with its sons. Moreover, STKM has a low complexity communication: each node sends a message and receives d (the number of neighbors) from its neighbors. STKM can resist against node capture attacks. However, if captured nodes cannot be detected timely, they may continuously send *REFRESH-REQ* message to consume the energy of intermediate nodes and finally evict trusted nodes. With the growth of the network size, the communication overhead of the network associated with refreshing K_r grows dramatically. Furthermore, in STKM, all data encryption keys are derived from the global key. Once this key is captured, the entire data communication is exposed to adversaries. Finally, STKM neither introduces an approach to detect a compromised/captured node nor a method to determine the re-keying period.

[35] outlined two security flaws (the use of a single group key and updating the compromised keys with non-compromised keys) of the existing group key renewal schemes and proposed a lightweight key renewal scheme to evict the compromised nodes, which is depicted is shown in Figure 3. It includes four steps. The pairwise key generation step is only invoked at the network boot-up time, while the other three steps are repeated in a periodical manner until the extinction of the network. To secure the communication from the nodes to the BS, two kinds of keys are required. One kind is used for the data transmissions from nodes to their CH, which is called intra-cluster keys. The other kind is used for the communication from the CH to the BS which is called CH-BS keys. Each node establishes pairwise keys with its neighbors using its ID and the network-wise key. This scheme periodically renews the cluster organization rather than renewing the pairwise keys. The cluster organization procedure is invoked regularly using a timer. The node with the lowest identifier and the highest energy reserves among the neighbors becomes the CH during each cluster organization period.

Once a CH recognizes a compromised node, it appends the list of compromised nodes to the aggregated data which is finally transfered to the BS. When its timer expires, the BS broadcasts its clock time and the list of compromised nodes. Then the innocent nodes construct a secure cluster structure in the next cluster organization time by rejecting the communication with the blacklisted nodes. After the completion of the cluster organization process, each CH generates a CH-BS key by

XORing all pairwise keys shared with its members and sends the BS its membership. Since the BS is the dealer of the network-wide key, the pre-assigned keys of the CH as well as its members, it can easily generate the same CH-BS key using the received cluster membership information. Simulation results demonstrate that this approach is more compromise tolerant and more energy efficient than SHELL [22]. Nevertheless, it shares the high energy consumption problem with all hierarchal algorithms (e.g., LEACH [51]) related to cluster formation and maintenance. Moreover, if the time period for cluster organization is very short, it increases communication overhead significantly; if it is too long, this scheme will be attackable.

[36] proposed a location-aware dynamic session-key management for grid-based WSNs. The generation of dynamic session keys is based on the DARQ scheme [52], one-way hash function, a "two-way" mutual authentication and a symmetric encryption mechanism. One node with the largest residual energy in a grid will be selected as the cluster node. A base station sends a request packet to the cluster node in its grid to know whether there has been an interesting event. Upon receiving the request packet, the cluster node sends a reply packet back to the base station . When a cluster head detects an event, it broadcasts a packet to all the cluster nodes. In the proposed scheme, after each data transaction, both the session key (to secure communications between the cluster node and the sensor node) and the message key (to encrypt and decrypt the updated-key messages) of each sensor node are updated. Compared to DARQ, the location-aware dynamic session-key management scheme is more robust against various attacks, however, the energy consumption is much higher than that of DARQ. In addition, the proposed scheme does not have a method to revoke compromised sensor nodes/cluster node or to distribute keys for newly joined sensor nodes.

3.2.3. Heterogeneous network-based schemes

[37] used the concept of genetic algorithms to design appropriate key-generating functions for rekeying. The network consists of the sink node, headers and sensor nodes. The sink node is responsible for generating appropriate key-generating functions (sets of code slices) and distributing them to headers and sensor nodes. Each possible key generating function is encoded as a chromosome. Those chromosomes which satisfy certain power-consumption constraints and have relatively high fitness values [37] are selected for rekeying. The energy consumption of this scheme is controllable, as it only chooses chromosomes with a relative low power-consumption. However, the code-slice pool should be very large, otherwise, most of the chromosomes for different rekeying processes may be the same. Furthermore, it works with the strong assumption that an adversary cannot compromise any sensor node in a certain time limit t.

[38] proposed a key agreement algorithm for WSNs using public key cryptography. The network is composed of a gateway and some sensor nodes. The gateway is less resource constrained and tamper-resistant. The gateway is assumed to be able to detect nodes' capturing and the sensor nodes are aware of their locations. After deployment, sensor nodes establish pairwise keys among them according to a specific routing algorithm, instead of loading full pairwise keys of all the neighboring nodes. The sensor nodes are grouped into clusters. The gateway generates a session key for each cluster and sends it to CHs. This session key can be updated periodically upon the discovery of a captured node or when a node reaches its lifetime. When a new node wants to join the network, it needs to be preloaded with its public and private keys and the gateway's public key. This approach achieves good storage efficiency, as it only establishes pairwise keys for sensor nodes in the Routing-Path-Table of the gateway, furthermore, both forward and backward secrecy are achieved. However, a number of captured nodes may collude and collectively reveal all the

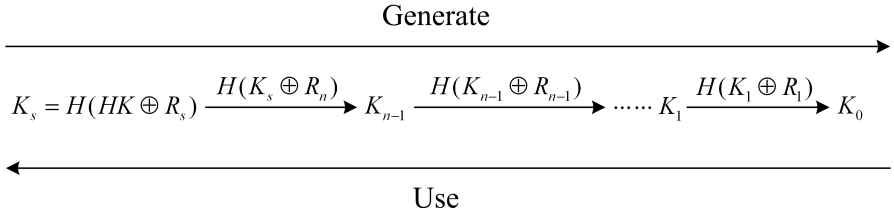


Figure 4: Construction of an one-way hash chain.

session keys.

[39] provided a forward authentication key management scheme for heterogeneous sensor networks. This heterogeneous network includes the BS, powerful high-end sensors (H-sensors) and low-end sensors (L-sensors). The H-sensors work as cluster heads. It assumes that H-sensors can directly communicate with the BS, while L-sensors can only communicate with each other through a H-sensor. The proposed scheme loads the same hash function into the BS, H-sensors and L-sensors. The BS generates key-chains for H-sensors. Thereafter, the keys in the key-chain are used for transmitting messages between a H-sensor and its member L-sensors. Figure 4 shows the construction of an one-way hash chain. When a compromised node is discovered by the BS, the BS broadcasts a revocation message to all H-sensors. Then, the corresponding H-sensor forwards the revocation message to its member L-sensors to remove the ID of the compromised node. Afterwards, the H-sensor uses the pairwise key to encrypt new communication key for L-sensors. When the last key in the key-chain has been used and the H-sensor has sufficient power, a new key-chain will be created and the new key will be distributed with the pairwise key. This proposed scheme decreases the memory and computational requirements of L-sensors and the more powerful H-sensors are robust to various attacks, for example, guessing attacks, replay attacks and man-in-the-middle attacks.

4. Summary and Future Research Directions

Table 1 summarizes our analysis of the surveyed dynamic key management schemes, considering the security metrics. In Table 1, regarding resilience, "High" means that the compromised node cannot affect non-compromised nodes, "Medium" refers to the compromised node only affects his neighbors, and "Low" denotes the compromise of one node leads to the compromise of the whole network. Table 2 summarizes the efficiency and flexibility properties of the surveyed schemes. Here, memory refers to the memory consumption per normal node. Processing describes operations required per key generation and distribution. Speaking of scalability, "High" refers to that no further cost (storage, communication and computation) is induced when one node is added to the network, "Medium" implies the cost induced is reasonable, while "Low" means that high costs are generated with the addition of one node. The upper part of both Table 1 and 2 shows the distributed key management schemes, while the lower part presents the centralized ones.

As pointed out in Section 3.2, centralized dynamic key management schemes rely on a single authority to generate or distribute new keys. However, this base station or third party can become a single point of failure as well as an obvious attack target. Therefore, new schemes should be proposed to secure the base station or third party. On the other hand, the privilege deprivation protocol presented in [53] can be applied to restrict the privilege of the centralized authority immediately when its failure and compromise has been detected.

Table 1: Summary of security analysis

Scheme	Security metrics			
	Forward and backward secrecy	Collusion resistance	Resilience	Node revocation
SHELL [22]	Both	Partial	Depends on k and the probability that two nodes share a key	Revoke a compromised CH through cluster reorganization or revoke a non-CH node locally
LOCK [23]	Both	Partial	Depends on k and the probability that two nodes share a key	Revoke a compromised CH with the BS or revoke a non-CH node locally
B-PCGR [26]	Backward secrecy	At most μ	μ-secure	Revoke the group key
C-PCGR [26]	Backward secrecy	At most μ	μ-secure	Revoke the group key
Cluster-based [27]	Both	At most t	t-secure	Remove the hierarchical key of the compromised node and revoke the group key
Compromise-resilient [28]	Both	Yes	High	N/A
EDDK [29]	Both	Yes	High	Remove the keys for revoke ID and revoke the local cluster key
KeyRev [30]	Both	No	t-secure	Revoke the session key
EEKM [32]	No	No	Medium	Revoke the group key and update the initiate master key
Key revocation [33]	No	No	Low	Revoke all the keys of nodes in the path
STKM [34]	No	No	Medium	Remove node ID and revoke the global key
Cluster-based key renewal [35]	No	No	Medium	Revoke compromised CHs and normal nodes through cluster reorganization
Location-aware [36]	No	No	Low	N/A
GA-based [37]	Both	No	Medium	No
Key agreement [38]	Both	No	High	Remove the ephemeral random key and revoke the session key
Forward authentication [39]	No	No	Medium	Remove node ID

Security and efficiency are two conflicting objectives that a dynamic key management scheme need to achieve. On the one hand, strong security protocols usually require larger amounts of memory and higher power consumption. On the other hand, the constraints on hardware resources of the sensor platform prohibit sensor nodes from being equipped with strong security contributes. The summary results in Table 1 and Table 2 show there is no one-size-fits-all solution for dynamic key management in WSNs. However, there are still several opportunities for dynamic key management to utilize the constrained resources of sensor nodes effectively. An obvious manner to improve the efficiency and scalability of most schemes is to reduce the amount of information exchanged between nodes. This is the case of the cluster-based key renewal scheme [35], which does not need keys of member nodes to be transfered to generate a new CH-BS key. The PCGR and cluster-based group key management scheme can be modified in this manner. Compared to processing or storage, the communication between nodes is much more expensive in current sensor node platforms [54].

Another observation concerning security is the use of public key cryptography, which supports end-to-end security. From this survey, it can be seen there are only few solutions based on public

Table 2: Summary of efficiency and flexibility analysis

Scheme	Efficiency and flexibility metrics				
	Memory	Processing	Mobility	Scalability	Key connectivity
SHELL [22]	k Keys + Keys' identifier	m Enc/Dec	No	Low	Probability that two nodes share a key, say p_1
LOCK [23]	k Keys + Keys' identifier	m Enc/Dec	No	Medium	p_1
B-PCGR [26]	g' Polynomial + Shares of neighbor's e-polynomial	$2n$ Enc/Dec + o($(n+1)\ s^2 + \mu^3$) Mul/Div	No	High	100%
C-PCGR [26]	g' Polynomial + Shares of neighbor's 0-level and 1-level e-polynomial	$2n$ Enc/Dec + o($(2n+1)\ s^2 + 2\mu^3$) Mul/Div	No	High	100%
Cluster-based [27]	2 + 1 Cluster identifier + 1 Personal secret	$2t$ Enc/Dec + O($t\ log^2 t$) Polynomial evaluation	No	High	100%
Compromise-resilient [28]	1 + 1 Secret value + 1 t-degree perturbed polynomial	O((n_a - n_c + 1) t) Mul + (n_a - n_c + 3) Hash function	No	N/A	100%
EDDK [29]	5 + Neighbor table	1 Enc/Dec + 1 Pseudorandom function	Yes	Low	100%
KeyRev [30]	5 + Node revocation list + m Personal secrets	1 Hash function + 2 Polynomial evaluation	No	High	100%
EEKM [32]	8 + Key material box	12 Hash function + Extra	No	Low	100%
Key revocation [33]	3 + s Keys	1 Hash function + (o($m\ log_m n_1$)) Enc + s Dec (for m-ary tree)	No	Low	Probability that two nodes are in the same path
STKM [34]	3 + Number of sons	$\sum_{i=1}^{n_1}$ (1 Enc + d Dec)	No	Medium	100%
Cluster-based key renewal [35]	2 + Number of neighbors	2 XOR	No	Medium	100%
Location-aware [36]	3 + Neighbor table	5 Enc/Dec + 14 Hash function + 1 Add + t Div + Extra	No	N/A	100%
GA-based [37]	1	k Enc/Dec	No	N/A	Probability that two nodes are sharing the same code slice
Key agreement [38]	4	1 Enc/Dec + 1 Mul/Div	No	High	Probability that two nodes are in the same routing path
Forward authentication [39]	3	1 Enc/N_{HL} Enc/Dec + N_{HL} Hash function	No	Medium	0

n: Number of one node's trusted neighbors, t: Number of personal secrets, n_a: Number of all sensor nodes, n_c: Number of compromised sensor nodes,
m: Number of member nodes, n_1: Total number of sensor nodes, d: Number of neighbors, N_{HL}: Number of L-sensors in one cluster, t: Number of aggregation data for a cluster node

key cryptography. A promising approach for key management in WSNs is to combine the merits of both public and symmetric cryptographic techniques, in which each node is equipped with a public key system to establish end-to-end symmetric keys with other nodes as that is used in EDDK [29]. Another critical issue that arises from this approach is the development of more efficient public key algorithms and their implementations which can be widely used on current sensor nodes [3].

In many existing schemes, the revocation of a compromised node requires many keys to be revoked [34, 29, 30], which leads to serious impacts on the network connectivity. Future research should especially seek techniques for compromised node discovery and develop efficient methods to revoke those nodes. In dynamic key management, keys are supposed to be updated periodically. However, none of the existing protocols have decided and analyzed the key update period, which may depend on node failure detection, compromised node discovery, data traffic volume and extra processing load incurred by all nodes. Also, dynamic key management in special WSNs, such as sparse WSNs [55] and mobile WSNs [56] are still open research fields. Moreover, the authentication delay introduced in key-chain-based broadcast authentication mechanisms cannot satisfy real-time applications. Furthermore, there are many potential ways to disrupt the time-synchronization required in broadcast authentication techniques. Hence, the development of time-synchronization independent broadcast authentication mechanisms is another promising area for researchers.

5. Conclusion

In this paper, we presented an overview of state of the art dynamic key management schemes in WSNs. With the wide application of WSNs, as one of the fundamental security issues, dynamic key management is attracting more attention from the researchers and industrial engineers and many schemes were already proposed. We discussed the basic requirements of dynamic key management in WSNs, surveyed the proposed schemes for these environments and highlighted the security and performance advantages and disadvantages of each scheme. Finally, we have summarized and analyzed these techniques according to the discussed evaluation metrics. In summary, it is not possible to find one single perfect scheme can perform well in all evaluation metrics as each of them has some definite strengths, weaknesses and suitability for specific situations. The ultimate objective of this study is to encourage more researchers to design and improve potential proposals in dynamic key management for wireless sensor networks.

Acknowledgment

The research leading to these results was supported by "Regionale Wettbewerbsfähigkeit und Beschäftigung", Bayern, 2007-2013 (EFRE) as part of the SECBIT project (http://www.secbit.de), the European Community's Seventh Framework Programme through the EuroNF Network of Excellence (IST, FP7 Call 1, ICT-2007-1-216366) and the European Community's Seventh Framework Programme through the EINS Network of Excellence (grant agreement no. [288021]).

[1] L. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A Survey on Sensor Networks, IEEE Commun. Mag. 40 (2002) 102–114.

[2] T. Kavitha, D. Sridharan, Security Vulnerabilities In Wireless Sensor Networks: A Survey, Journal of Information Assurance and Security (JIAS) 5 (2010) 31–44.

[3] W. Hu, H. Tan, C. Corke, W. C. Shih, S. Jha, Toward trusted wireless sensor networks, ACM Transactions on Sensor Networks (TOSN) 7 (2010) 5:1 – 5:25.

[4] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone, Handbook of Applied Cryptography, CRC Press, 2 edn., 1999.

[5] L. Eschenauer, V. D. Gligor, A key-management scheme for distributed sensor networks, in: Proceedings of the 9th ACM conference on Computer and communications security, 2002.
[6] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, A. Khalili, A pairwise key predistribution scheme for wireless sensor networks, ACM Trans. Inf. Syst. Secur. 8 (2) (2005) 228–258.
[7] B. Zhou, S. Li, Q. Li, X. Suna, X. Wang, An efficient and scalable pairwise key pre-distribution scheme for sensor networks using deployment knowledge, Computer and Communication 32 (2009) 124–133.
[8] Y. Zhang, W. Yang, K. Kim, M. Park, An AVL Tree-Based Dynamic Key Management in Hierarchical Wireless Sensor Network, in: International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2008.
[9] J. M. Kim, T. H. Cho, A*-based key tree structure generation for group key management in wireless sensor networks, Comput.Commun. 31 (2008) 2414–2419.
[10] D. Sun, B. He, Review of Key Management Mechanisms in Wireless Sensor Networks, ACTA AUTOMATICA SINICA 32 (6) (2006) 900–906.
[11] A. Barati, M. Dehghan, H. Barati, A. A. Mazreah, Key Management Mechanisms in Wireless Sensor Networks, in: Proceedings of the 2nd International Conference on Sensor Technologies and Applications, 2008.
[12] S. Chattopadhyyay, A. K. Turuk, A Survey on Key Pre-distribution Scheme in Homogeneous Wireless Sensor Networks, in: Proceedings of the 2nd International Conference on Advanced Computing and Communication Technologies for high performance Applications, 2010.
[13] M. A. S. Jr, P. S. L. M. Barreto, C. B. Margi, T. C. M. B. Carvalho, A survey on key management mechanisms for distributed Wireless Sensor Networks, Computer networks 54 (2010) 2591–2612.
[14] R. Manoj, N. Dhinakaran, A Survey of Key Predistribution Schemes for Key Management in Wireless Sensor Networks, in: Proceedings of the 12th International Conference on Networking, VLSI and signal processing (ICNVS'10), 2010.
[15] H. Lee, Y. H. Kim, D. H. Lee, J. Lim, Classification of Key Management Schemes for Wireless Sensor Networks, in: ASWAN international workshops on Advances in Web and Network Technologies, and Information Management, 2007.
[16] S. M. K. R. Raazi, S. Lee, A Survey on Key Management Strategies For Different Applications of Wireless Sensor Networks, Journal of Computing Science and Engineering 4 (2010) 23–51.
[17] Y. Xiao, V. K. Rayi, B. Sun, X. Du, F. Hu, M. Galloway, A survey of key management schemes in wireless sensor networks, Comput. Commun. 30 (11–12) (2007) 2314–2341.
[18] J. Zhang, V. Varadharajan, Wireless sensor network key management survey and taxonomy, Journal of Network and Computer Applications 33 (2010) 63–75.
[19] B. Panja, S. K. Madria, A Survey of Dynamic Key Management Schemes in Sensor Networks, IGI Global, 326–351, 2010.
[20] S. Mishra, Key management in large group multicast, Tech. Rep., Department of Computer Science, University of Colorado, 2002.
[21] F. Ye, H. Luo, J. Cheng, S. Lu, L. Zhang, A Two-Tier Data Dissemination Model for Largescale Wireless Sensor Networks, in: Proceedings of the 8th ACM/IEEE International Conference on Mobile Computing and Networking, 2002.
[22] M. F. Younis, K. Ghumman, M. Eltoweissy, Location-Aware Combinatorial Key Management Scheme for Clustered Sensor Networks, IEEE Transactions on Parallel and Distributed Systems (TPDS) 17 (2006) 865–882.
[23] M. Eltoweissy, M. Moharrum, R. Mukkamala, Dynamic key management in sensor networks, IEEE Commun. Mag. 44 (2006) 122–130.
[24] C. Lo, C. Huang, S. Chen, An efficient and scalable EBS-based batch rekeying scheme for secure group communications, in: Proceedings of IEEE Military Communications Conference, 2009.
[25] S. M. K. Raazi, H. Lee, S. Lee, Y. Lee, MUQAMI+: a scalable and locally distributed key management scheme for clustered sensor networks, Annuals of Telecommunications 65 (2010) 101–116.
[26] W. Zhang, S. Zhu, G. Cao, Predistribution and local collaboration-based group rekeying for wireless sensor networks, Ad Hoc Networks 7 (2009) 1229–1242.
[27] Y. Zhang, Y. Shen, S. Lee, A Cluster-Based Group Key Management Scheme for Wireless Sensor Networks, in: Proceeding of the 12th Asia-Pacific Web Conference, 2010.
[28] S. Guo, Z. Qian, Smart Wireless Sensor Networks, chap. A Compromise-resilient Pair-wise Rekeying Protocol in Hierarchical Wireless Sensor Networks, InTechOpen, 18:1–18:6, 2010.
[29] X. Zhang, J. He, Q. Wei, EDDK: Energy-Efficient Distributed Deterministic Key Management for Wireless Sensor Networks, EURASIP J. Wirel. Commun. Netw. 2011 (2011) 1–11.
[30] Y. Wang, B. Ramamurthy, X. Zou, Y. Xue, An efficient Scheme for Removing Compromised Sensor Nodes

from Wireless Sensor Networks, Tech. Rep., Department of Computer Science and Engineering, University of Nebraska-Lincoln, 2007.
[31] Y. Wang, B. Ramamurthy, Y. Xue, A Key Management Protocol for Wireless Sensor Networks with Multiple Base Stations, in: Proceeding of IEEE International Conference on Communications (ICC), 2008.
[32] K. Paek, U. Song, H. Kim, J. Kim, Energy-Efficient Key-Management (EEKM) Protocol for Large-Scale Distributed Sensor Networks, Journal of Information Science and Engineering (JISE) 24 (2008) 1837–1858.
[33] G. Dini, I. M. Savino, An efficient key revocation protocol for wireless sensor networks, in: IEEE Symposium on a World of Wireless Mobile and Multimedia Networks, 2006.
[34] M.-L. Messai, M. Aliouat, H. Seba, Tree based protocol for key management in wireless sensor networks, EURASIP J. Wirel. Commun. Netw. 2010 (2010) 59:1–59:13, ISSN 1687-1472.
[35] G. Wang, S. Kim, D. Kang, D. Choi, G. Cho, Lightweight Key Renewals for Cluster Sensor Networks, Journal of Networks 5 (2010) 300–312.
[36] C. L. Chen, I. H. Lin, Location-Aware Dynamic Session-Key Management for Grid-Based Wireless Sensor Networks, Sensors 10 (2010) 7347–7370.
[37] C. Wang, T. Hong, G. Horng, W. Wang, A GA-Based Key-Management Scheme in Hierarchical Wireless Sensor Networks, International Journal of Innovative Computing, Information and Control (IJICIC) 5 (2009) 4693–4702.
[38] M. H. Eldefeawy, M. K. Khan, K. Alghathbar, A Key Agreement Algorithm with Rekeying for Wireless Sensor Networks using Public Key Cryptography, in: IEEE International Council of Societies of Industrial Design, 2010.
[39] J. Y. Huang, I. E. Liao, H. W. Tang, A Forward Authentication Key Management Scheme for Heterogeneous Sensor Networks, EURASIP J. Wirel. Commun. Netw. 1 (2011) 1–10.
[40] M. Eltoweissy, M. H. Heydari, L. Morales, I. H. Sudborough, Combinatorial Optimization of Group Key Management, J. Network and Sys. Mgmt., Special Issue on Network Security 12 (2004) 33–50.
[41] M. Moharrum, R. Mukkamala, M. Eltoweissy, TKGS: verifiable threshold-based key generation scheme in open wireless ad hoc networks, in: Proceeding of the 13th International Conference on Computer Communications and Networks, 2004.
[42] R. Divya, T. Thirumurugan, A Novel Dynamic Key Management Scheme Based On Hamming Distance for Wireless Sensor Networks, International Journal of Scientific and Engineering Research (IJSER) 2 (2011) 1–12.
[43] F. Kong, C. Li, Q. Ding, F. Jiao, Q. Gu, Collusion Problem of the EBS-Based Dynamic Key Management Scheme, Journal of Software 20 (9) (2009) 2531–2541.
[44] K. Kim, Y. Zhang, W. Yang, M. Park, An authentication protocol for hierarchy-based wireless sensor networks, in: Proceeding of the 23rd IEEE International Symposium on Computer and Information Science, 2008.
[45] A. Shamir, How to share a secret, Communications of the ACM (CACM) 22 (1979) 612–613.
[46] A. Chadha, Y. Liu, S. K. Das, Group Key Distribution via Local Collaboration in Wireless Sensor Networks, in: Proceedings of IEEE Communications society Conference on Sensor and Ad Hoc Communications and Networks, 2005.
[47] W. Zhang, M. Tran, S. Zhu, G. Cao, A Random Perturbation-Based Scheme for Pairwise Key Establishment in Sensor Networks, in: ACM International Symposium on Mobile Ad Hoc Networking and Computing, 2007.
[48] J. Deng, C. Hartun, R. Han, S. Mishra, A Practical Study of Transitory Master Key Establishment For Wireless Sensor Networks, in: Proceeding of the 1st Conference on Security and Privacy in Communication Networks, 2005.
[49] R. L. Rivest, M. E. Hellman, J. C. Anderson, J. W. Lyons, Responses to NIST's proposal, Communications of the ACM (CACM) 35 (1992) 41–54.
[50] A. Perrig, R. Szewczyk, V. Wen, D. Culler, J. D. Tygar, SPINS: Security Protocols for Sensor Networks, Wireless Networks 8 (2002) 521–534.
[51] M. J. Handy, M. Haase, D. Timmermann, Low energy adaptive clustering hierarchy with deterministic cluster-head selection, in: Proceedings of the 4th International Workshop on Mobile and Wireless Communications Networks, 2002.
[52] T. S. Chen, Y. S. Chang, H. W. Tsai, C. P. Chu, Data Aggregation for Range Query in Wireless Sensor Networks, in: Proceedings of IEEE Wireless Communications and Networking Conference, 2007.
[53] W. Zhang, H. Song, S. Zhu, G. Cao, Least Privilege and Privilege Deprivation: Towards Tolerating Mobile Sink Compromises in Wireless Sensor Networks, in: ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 2005.
[54] P. Desnoyers, D. Ganesan, P. Shenoy, TSAR: A two sensor storage architecture using interval skip graphs, in: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems, 2005.

[55] G. Anastasi, M. Conti, M. D. Francesco, An Analytical Study of Reliable and Energy-Efficient Data Collection in Sparse Sensor Networks with Mobile Relays, in: Proceeding of the 6th European Conference on Wireless Sensor Networks, 2009.

[56] I.-H. Chuang, W.-T. Su, C.-Y. Wu, J.-P. Hsu, Y.-H. Kuo, Two-Layered Dynamic Key Management in Mobile and Long-Lived Cluster-Based Wireless Sensor Networks, in: IEEE Wireless Communications and Networking Conference, 4145 –4150, 2007.