On the Key Revocation Schemes in Wireless Sensor Networks

Dieynaba Mall and Karim Konaté
Department of Mathematics and Computer Science
Université Cheikh Anta Diop de Dakar
Dakar, Senegal
dieynaba.mall@ucad.edu.sn, karim.konate@ucad.edu.sn

Al-Sakib Khan Pathan
Department of Computer Science
International Islamic University Malaysia
Kuala Lumpur, Malaysia
sakib@iium.edu.my

Abstract—Among all security issues, key management is the most attractive mechanism to ensure security of applications and network services in wireless sensor networks. Key management includes two important aspects namely: Key distribution, which constitutes the task of distributing secret keys to nodes in the network and Key revocation, which refers to the task of securely withdrawing the key information relating to any compromised network node. While in the literature, key distribution has been extensively studied; key revocation has received relatively little attention. In this paper, we present a survey on the state-of-the-art key revocation techniques and also present the security and performance analysis that highlights the advantages and disadvantages of each scheme.

Keywords—Key; Management; Revocation; Security; Sensor; Wireless

I. INTRODUCTION

A wireless sensor network (WSN or simply, sensor network throughout this paper) is a specialized ad hoc network composed of a large number of low cost and limited-power sensor nodes. It is designated to perform a wide variety of missions that are difficult for humans to carry out.

Due to the insecure nature of the wireless communication medium and dynamic changes of the network topology, WSNs are vulnerable to a wide range of attacks and are thus difficult to secure. In addition to this, because of the constrained nature of sensors in a WSN, usually being limited in energy and computing power as well as their field of application, security issues in such a network must be carefully carried out. Cryptographic techniques are essential to guarantee the security services such as confidentiality, integrity, and authentication and to resist against the increasing number of attacks. Thus, an effective management of cryptographic keys is always required to provide communication security in sensor networks.

Generally a WSN is designed for deployment in open, in an unmonitored environment, thus exposing nodes to physical attacks. Any key management solution in WSNs must then be provided with the ability to revoke the cryptographic keys of captured nodes. Otherwise, the entire network’s operation could be compromised by adversaries.

We seek to provide a better understanding of the current research issues related to the revocation problem in WSNs. To do so, we present a survey on the state-of-the-art techniques and highlight the advantages and disadvantages of various schemes. Moreover, we compare and evaluate these solutions based on each of the two categories of the schemes.

The remainder of this paper is organized as follows: in section II, we review the solutions related to the issue of key revocation in sensor networks, followed by a security and performance analysis in section III. This paper is finally concluded in Section IV.

II. REVOCATION SCHEMES IN WIRELESS SENSOR NETWORKS

In wireless sensor networks, revocation schemes can be categorized as either centralized or distributed, depending on the level of involvement of a designated centralized authority.

A. Definitions and Notations

The major terms used in this paper are defined as follows:
- **Neighboring node** – is a node located within the communication range of a given node;
- **Target node** – is the node to be revoked in the revocation process;
- **Participant** – any node sharing a pairwise key with the target node is a participant. We can distinguish two categories of participants depending on their proximity to the target node.
  1. **Local participant** – which is a neighboring node of the target node;
  2. **Non-local participant** – which is a node located at greater distance;
- **Local neighborhood Broadcast** – is a broadcast limited to the neighborhood of a given node. More specifically, this refers to a multi-hop broadcast that originates within a given neighborhood and attains all the nodes inside that neighborhood;
- **Centralized authority** – is an entity sharing a pairwise key with every network node and which is responsible for conducting the centralized revocation of any node in...
the network (for instance, the base station (BS) in a WSN);

- **Revocation Attack** – In this attack, an adversary uses the distributed approach of a revocation protocol to selectively revoke uncompromised nodes from the network. Regarding the centralized approach, this attack can be independently initiated by a single adversary.

The notations used throughout this paper are listed in the Table I.

**B. Centralized Revocation Schemes**

In centralized key revocation schemes, a single designated authority is responsible for conducting revocation decisions.

1) **EG Scheme:** The issue of key revocation in sensor networks was first addressed by Eschenauer and Gligor which was presented in [1] - the first centralized key revocation scheme in WSNs. In their work, it is assumed that a mobile controller node which has a large communication range (for example the BS) performs the revocation process.

Before conducting revocation, this controller generates a signature key $K_c$, encrypts it using a pairwise key $K_{ei}$ ($K_{ei}$ is preloaded by each node) and unicasts it to each uncompromised sensor node. In this way, to revoke a compromised node, it will broadcast a message signed by $K_c$ containing all the key identifiers possessed by a compromised node. After receiving and verifying this message, the corresponding key is removed from the nodes’ key rings.

2) **KeyRev Scheme:** In [2], Wang, Ramamurthy, and Zou proposed another centralized key revocation protocol KeyRev which uses key updating techniques to obsolesce the keys owned by the compromised nodes and removes them from the network. Their scheme does not depend on a specific key distribution scheme but to describe their proposal, the authors assume that the basic random key distribution scheme [1] is used.

In KeyRev, the authors propose using only two kinds of keys namely: encryption key $k_{encr}$ and message authentication code (MAC) key $k_{mac}$ for secure communication in the network.

These keys are generated so that each of them is bounded to a session key $k_j$. Both of these keys will be changed when the session key $k_j$ is updated. To do so, Wang, Ramamurthy, and Zou suggest using a session key distribution scheme based on [11].

In the setup phase of this scheme, a setup server loads for each session $j$, on each node $i$, the personal secret derived from a $2t$-degree masking polynomial $h_i(x)$ randomly chosen by the setup server. This server also loads this polynomial on the BS. In this scheme’s broadcast phase, the BS broadcasts to the non-revoked nodes a message whose format is: $B = \{R\} \cup \{P_j(x)\} \cup \{Q_j(x)\}$, where $P_j(x)$ and $Q_j(x)$ are obtained using the received list $R$ or set of revoked group members in session $j$, $R = \{r_1, r_2, ..., r_w\}$, with $w \leq t$ and a $t$-degree polynomial $p_j(x)$ randomly picked in the setup phase by the setup server. To recover the session key, any non-revoked node $i$ receiving such a broadcast message will calculate the polynomials $P_j(x)$ and $Q_j(x)$ at point $i$ to get the new session key. Therefore, it is possible to stop the compromised sensors from deriving $k_{encr}$ and $k_{mac}$ by stopping them from obtaining the new session key. Thus, the compromised nodes can be removed securely from the network.

3) **DLS Scheme:** To reinforce the security of the KeyRev scheme, Park, Zhang, and Kim suggest in [3] the idea of dynamic sessions to reduce the lifetime of compromised nodes in the network. The authors propose managing the existing interval time of session keys which includes two aspects: the revocation time and the active time.

After generation and distribution of a session key in the revocation time, the authors turn into managing the active time which is longer than the revocation time. They partition it into session level to get possibility to adjust it dynamically and to provide containment of the next attack. The active time of an attacker is from starting the active time of each session until revocation time of the compromised node.

In their solution, the authors take into account the situation in which sensor nodes are compromised by continuous attack. To achieve precaution against this attack model, they propose employing both dynamic size of session scheme and event driven scheme.

In dynamic size of session scheme, when the BS detects compromised nodes at previous session, next session level has to start off from initial level (usually at level 1). It is used simultaneously with dynamic size of session and event driven scheme in which when the BS detects compromised nodes, it directly distributes a new session key. By adopting this way, the compromised nodes can be revoked quickly without waiting for the completion of current session.

**C. Distributed Revocation Schemes**

In distributed or decentralized revocation solutions, no centralized authority is used. The revocation decisions are made instead by the neighbors of a compromised node.

---

**TABLE I.** LIST OF NOTATIONS

| n | Total number of nodes in the network |
| m | Total number of participants for each node |
| t | Threshold or number of votes required to revoke a node |
| $E_k(M')$ | Message $M'$ encrypted using the key $k$ in authenticated encryption mode |
| $s_{total}$ | Total number of revocation sessions available against each node |
| $H(y)$ | Cryptographic hash of the value $y$ |
These nodes later vote to determine whether to revoke a node or not and the revocation takes effect only when the votes tally exceeds a specified threshold.

1) **CPS Scheme**: To reduce the insufficiencies related to the scheme presented by Eschenauer, and Gligor in [1], Chan, Perrig, and Song propose in [4] another revocation solution which is based on the distributed approach. In this solution, the authors assume that keys are randomly-pairwisely distributed in the network. Therefore, each node shares a unique pairwise key with $m$ other distinct participants randomly selected in order to facilitate node-to-node authentication. Each of these $m$ participants is assigned a random voting key $k_i$. In addition to the pairwise keys associated with a target node, each of the target node’s participants carries a preloaded vote $hash(k_i)$ which will be used as a message to inform that the target node has been compromised. To efficiently authenticate the $m$ hash values, the authors propose to use a Merkle tree [12]. Therefore, each participant must also store the $log m$ hash values required to authenticate a vote message. To perform a vote against a target node, a participant has to propagate its message across the network to all voting members. If at least $t$ participants of a given target node have voted and if each of these votes has been successfully verified, the target node is marked as “revoked”. To this effect, a revocation message is propagated to all the other network nodes. Any node receiving this message will thereafter remove all the keys corresponding to the target node from its key ring, thus removing it from the network.

2) **CGPM Scheme**: To make an improvement to the proposal in [4], Chan, Gligor, Perrig, and Muralidharan present in [5] another distributed revocation scheme for WSNs that we term “CGPM Scheme”. They add new ideas such as revocation sessions, performing voting and revocation decision only by processing hop-limited local broadcast and propagating a single short message to the entire network to finalize the revocation outcome. Furthermore, they introduce the first distributed revocation protocol that provides rigorous proofs of desired properties namely: completeness, soundness, bounded time revocation completion, unitary revocation and revocation attack resistance.

With the help of the secret-sharing polynomial, the authors define the cryptographic hash of a random polynomial $q(x)$ as $H(q(x)) = H(a_0 \parallel a_1 \parallel a_2 \ldots \parallel a_{t-1})$, where $H$ is a hash function and $a_0, a_1, \ldots, a_{t-1}$ are the coefficients of $q(x)$. In this scheme, during the offline initialization phase, first it is computed for each of the $n$ network nodes $s_{total}$ $t$-degree random polynomials. Afterwards, on each node $j$ for each node $i$ of $j$’s $m$ participants and for each revocation session $s$ against target node $i$, a revocation vote from $j$ against $i$, based on the random polynomial $q_{i,s}$ is loaded. This revocation vote is represented as: $(E_{Mask_{i,s}}(q_{i,s}(x_{jis}),x_{jis}))$, where $Mask_{i,s}$ is the activation mask that $i$ gives to $j$. Note that the activation masks for the current revocation session are exchanged among the participants and the target node during the connection establishment time, this in order to decrypt the relevant votes. As in [4], in this scheme, for each node $j$ in $i$’s participants, for each vote, the $log m$ authenticating hash values are also loaded for the Merkle hash tree [12] together with the corresponding root $R_i$ and $H^2(q_{i,s})$. These $log m$ authenticating values are appended to the message to allow fellow participants of $i$ to verify the authenticity of the vote. The hash of the hash of $i$’s revocation polynomial, $H^2(q_{i,s})$ gives to non-local participants of the target node $i$ a mean to verify the authenticity of a revocation decision against $i$.

In this scheme, when $j$ detects that its neighbor $i$ is compromised, it votes against $i$ by performing an unencrypted hop-limited broadcast in the current revocation session $s$, $(q_{i,s}(x_{jis}),x_{jis})$, along with the Merkle authentication values needed to verify this vote’s authenticity. If $j$ counts at least $t$ revocation votes successfully verified (including its own if any), then it computes the revocation polynomial $q_{i,s}$, of $i$ for this session. Further, it determines the hash of $q_{i,s}, H(q_{i,s})$ and broadcasts it through the entire network. All the participants of $i$ will thereafter verify this pre-image against the value $H^2(q_{i,s})$ stored in their memory. If the verification is successful, all the keys shared with $i$ are deleted and $i$ is marked as “revoked”. The broadcast is then disseminated to the other participants until the entire network is covered.

3) **CYLL Scheme**: To further improve the CGPM scheme in terms of lower storage space requirement, reduced communication overhead and lower computational cost, Chao, Yang, Lin, and Li proposed in [6] a new distributed revocation scheme based on Blom’s $t$-secure property [13].

In their solution each node in every voting session is required to store $t$ elements of one row in a vote matrix and $t$ elements of one column in the public matrix. During the initialization phase, public and private matrix $G$ and $R_{i(s)}$ respectively are generated for each node $i$ in the WSN. The public matrix contains information freely available to all the nodes in the network, while the private matrix, used in each session contains secret information needed for vote verification purposes. In each session, the nodes also encrypt the loaded votes $E_{Mask_j}(V_{(i)})$ from the vote matrix $V_{(i)}$ and the corresponding column information of the public matrix, where the vote matrix $V_{(i)}$ = $(R_{i(s)} \cdot G)^T$ and $T$ denotes the matrix transpose operator. Note that within $V_{(i)}$ there exists one row referred to as the secret-sharing row $V_{(i)}^{j,i}$ which is not preloaded in any node during each session $s$. Each participant of node $i$ also stores $H^2(V_{(i)}^{j,i})$ which represents the hash of the elements in the secret sharing row, $H(\lambda_0(s) || \lambda_1(s) \ldots || \lambda_{t-1}(s))$, where $\lambda_k(s)$ is the $k$-th element of the secret sharing row $V_{(i)}^{j,i}$. The double hash value $H^2(V_{(i)}^{j,i})$ enables the non-local participants of target node $i$ to confirm the validity of the revocation decision for node $i$.

292
To vote against a target node $i$, the participant $j$ locally broadcasts its vote $V_{i,j}^{(s)}$ that is the $j$-th row of the matrix $V_{i,(s)}$ and $g_1^j$ that is the $j$-th column of the public matrix $G_1$. Upon receipt of these messages, the participant nodes perform a verification process. If one of the neighbors of a target node records at least $t$ votes against the target node, it solves the secret-sharing information of the node by resolving a linear system of $t$ equations and $t$ unknown variables. Thereafter, it computes the corresponding hash value $H(V_{i,j}^{(s)})$ which will represents the revocation message to be transmitted by broadcast to the entire network. The non local participants receiving this message verify it using $H^2(V_{i,j}^{(s)})$. If the control succeeds, they remove all the key information relating to the target node $i$ from their memory and add its identity to a revoked list.

4) CCL Scheme: In [7] Chuang, Chang, and Lin suggest another distributed revocation scheme for sensor networks which is not based on symmetric cryptography as the other schemes, but on the features of the one-way hash chain, the certificate revocation list and the public-key cryptography. The authors transform the concept of certificate revocation list (CRL) from centralized revocation to distributed voting approach.

In their scheme, it is assumed that the BS represents the certification authority (CA) that distributes the public key cryptography (PKC) to each node before deployment and each node will authenticate this PKC via the BS after deployment. It is also assumed that all the system clocks are synchronized in the network and the maximum lifetime $T'$ for every PKC will be the same and so will be the update interval $L$ ($T'/L = j$ ($j > 1$)).

After deployment, nodes exchange their certificates to build secure links between each other. A certificate in this scheme includes a one-way hash value which can be used to judge if it is out of time or not.

When a node detects that another network has been compromised, it first broadcasts to all its neighbors a compromised revocation vote (CRV) whose format is: $[i,D,RevocationCID \oplus h_j, Timestamp]$. RevocationCID $\oplus h_j$ informs about the ID of the certificate to be revoked and Timestamp is used to prevent repeated broadcasting of CRV. If a neighbor node receives the CRV packet and finds that $h_j$ of $RevocationCID \oplus h_j$ is correct, it calculates $RevocationCID \oplus h_j \oplus h_j$ to obtain the value of $RevocationCID$. If $RevocationCID \oplus h_j \oplus h_j$ is correct, the neighboring node adds one more revocation vote against the target node. When the number of received revocation votes against the target node exceeds the threshold $t$ defined as $(m >> t > 1)$, the neighbor node revokes the key shared with the compromised node in its key ring and cuts off its link with the target node.

III. SECURITY AND PERFORMANCE ANALYSIS

In this section, we discuss the security of each of the two categories of solutions and evaluate their performance in terms of storage space requirement, communication overhead and computational complexity.

A. Security Analysis

1) Centralized Schemes: In centralized schemes it is assumed that an intrusion detection system [20]-[23] is employed to provide the BS with the ability to detect a compromised sensor prior revoking it. This central authority needs to communicate with sensor nodes to perform the revocation process.

All the centralized revocation protocols [1]-[3] are BS dependant in the sense that this centralized authority only has ability to detect a compromised sensor and is responsible for conducting revocation decisions. With these solutions, once an adversary comes to impersonate the BS, it gains the possibility to start a revocation attack. To deal with this problem, the authors in [2] and [3] propose to use some techniques in order to minimize the risk to impersonate the BS. They suggest using broadcast authentication schemes such as $\mu$TESLA [25], [26] in order to protect securely broadcasted messages from BS to nodes. They reinforce this measure by adopting a session key distribution process based on personal key distribution scheme [11], [26] to unable any compromise node to reveal a new session key. However, with the use of a $\mu$TESLA-like scheme to provide broadcast authentication, the adversaries regain chance to launch DoS attacks [16], [17] due to the delayed message authentication.

The KeyRev scheme has no solution to deal with compromised nodes until the distribution of the next update session key. It means that compromised nodes can be used as surrogates before distributing the next new session key. Consequently, adversaries may use the same pattern of attacks to compromise many other nodes during a session. If these attacks occur in the beginning of establishing MAC and encryption keys, they can cause DoS attacks, black hole attacks or any of the multiple attacks in the network. By avoiding the use of fixed duration of session, DLS proposes a way to reduce the life time of compromised nodes in the network and find a way to get immune against the above mention attacks. Further, it provides possibly to neutralize sniffing attacks by adjusting the session duration according to the cracking time in the same attack.

By the use of only two kinds of keys $K_{mac}$ and $K_{mac}$, KeyRev and DLS guarantee a defense against ciphertext analysis attacks [24], sybil attacks [18], [19], and physical attacks [19]. Indeed, in these schemes, the secret keys are hidden in a message and there is no way to obtain information out of the encrypted data since Message Authentication code (MAC) and encryption keys which are based on a session key and a pre-distributed key, are used. So KeyRev and DLS schemes have advanced secure against ciphertext analysis attacks. Also it is not possible to launch Sybil attacks against these schemes because each node only stores a compromised node’s ID in its node revocation list (NRL) and each node needs to have a MAC key and an encryption key to authenticate its identity. In spite of not

293
removing the pre-distributed key information, KeyRev and DLS are secure from physical attacks. This guarantee is provided due to the fact that even if attackers knew pre-distributed key, they need to update new session key to generate \( K_{enc} \) and \( K_{mac} \).

In summary, it can be said that all the centralized schemes are prone to the single point of failure problem since when a problem occurs with the centralized authority, the functioning of the revocation protocol, its security and the whole network will be affected.

2) Distributed Schemes: For these schemes, the designers also assume that in each node in the wireless sensor networks, an intrusion detection mechanism, which can detect most of the misbehaviors is installed.

Although, distributed revocation schemes solve the single point of failure problem inherent to centralized approaches, they must face other kinds of threats. Indeed, the execution of distributed revocation mechanisms can be done, having in the same time adversaries who exercise complete control of compromised nodes which in turn act as active participants. These malicious nodes can attempt to block revocation, or selectively revoke non-compromised nodes and disrupt the network operation. To deal with these vulnerabilities, it is crucial for a distributed revocation solution to provide fast responses and seal out a detected attack from the network before it can do significant harm.

The CPS scheme requires network-wide broadcast of long messages to revoke a compromised node, which is slow and makes the network prone to DoS attacks. This scheme is also susceptible to replay attacks since the session concept is lacking. Consequently each node must keep a record of which vote have been heard since the beginning of the network’s lifetime.

The solutions in [4]-[6] are based on the random pairwise key pre-distribution protocol and possess the node-to-node authentication property. Thus these schemes give a defense against attacks like node capture attacks, impersonation attacks. Indeed, with the random pairwise key pre-distribution scheme, any node that is captured reveals no information about links that it is not directly involved in; hence this property helps to get a perfect resilience against node capture. The node-to-node authentication property can further provide a protection against impersonation attacks since the nodes are able to verify the identities of nodes which they are communicating with, hence unless a node has already been captured, an adversary is unable to impersonate its identity.

In CGPM, CCL and CYLL, only the participants of the target node are allowed to disseminate vote messages. By this way these scheme can get a protection against DoS attacks.

An important technical challenge when designing a distributed node revocation solution relies on the problem of controlling replication of a single node identity or the generation of multiple Sybil node identities across the network. In [4], [5], and [6], the authors assume that the functionality for addressing this problem is already present. In [5] and [6] it is also assumed that sensor nodes are immobile; hence even if all malicious nodes share arbitrarily the needed information, the risk of these attacks is minimized.

In CCL a sequence number is not needed to verify a given certificate, and it is not easy for an adversary to learn about each hash value on the one way hash chain, thus the CCL scheme is immune from de-synchronization attacks. Also with the CCL scheme it is hard to launch a Sybil attack because the protocol requests to verify in first certificates.

In CGPM, Chan, Gligor, Perrig, and Muralidharan show that it is important to obtain rigorous proofs that could show that the distributed revocation protocols cannot be subverted or abused by compromised nodes. They present in [5] the first precise definition of desired properties namely: Completeness, Soundness, Bounded Time Revocation Completion, Unitary Revocation, and Revocation Attack Resistance that distributed revocation protocols must satisfy to ensure their correct operation and resistance to abuse by a malicious attacker. Furthermore, they provide proof that their proposal fulfils this list of designated properties. Further propositions [6], [7] regarding distributed node revocation in sensor networks demonstrate the usefulness and viability of the CGPM scheme, since they build their solutions upon this framework and employ with the basic properties defined in [5] to prove their correctness.

B. Performance Analysis

1) Centralized Schemes
   a) Space requirement: In the EG scheme, each node preloads a pairwise key \( k_{ij} \) shared with the BS which will be used to decrypt the signature key \( k_e \); hence, this solution does not require any space to store this pairwise key \( k_{ij} \). In KeyRev and DLS, in each session, each node must load the personal secret \( (h_i) \) required to update the session key \( k_S \). Hence, the space required to store the personal secrets will be zero (volatile memory could be simply used for each session).

b) Computational complexity: We consider the complexity for both the revocation message and its verification. With the EG scheme, to revoke a compromised node, the BS has just to broadcast through the entire network a list containing all the keys in the compromised node’s key ring. Since this message does not hide any secret, we can say that the computational complexity related to the revocation message itself is zero. With KeyRev and DLS, to recover the session key, a non-revoked node \( i \) receiving the broadcast message has to calculate some low degree polynomials \( \{p_j(x), q_j(x), g(x), p_i(x), q_i(x)\} \) at point \( i \). Thus to obtain the new session key, node \( i \) has to determine \( k_S = p_i(i) + q_i(i) \). Since the polynomial evaluations are fast, there is no computational complexity associated with the session key recover in KeyRev and DLS.

Regarding the revocation message verification, we have zero computational overhead for the EG scheme due to the fact that each node has only to compute one signature using the received \( k_e \). In KeyRev and DLS, the computational complexity which relies on the verification of the session key update message \( M \), with \( \{M = \{R \cup \{W(x) = g(x).k_e + h(x)\}\}\} \) is equal to zero since this message is not protected.
c) Communication overhead: Here also we take into account the communication overhead generated by the revocation message and the communication overhead involved by its verification.

With the EG scheme each revocation message includes all the key identifiers possessed by a compromised node. If we assume that in each node’s key ring there are \( m \) keys, the revocation message involves a communication overhead which is equal to \( O(m) \). In KeyRev, as well as in DLS, in order to update a session key, the BS broadcasts a message whose format is: \( M = \{R\} \cup \{W(x) = g(x). k_x + h(x)\} \).

Since this message incurs no information which can generate a communication overhead, then we get, for these two schemes, zero communication overhead associated with the revocation message itself.

In the EG scheme, the BS must unicast to each node an encrypted signature to allow the verification of the revocation message. Thus the communication overhead incurred is equal to zero. Concerning the KeyRev and the DLS schemes, as the session key update message does not constitute a secret information, any communication overhead is produced for its verification.

2) Distributed Schemes

a) Space requirement: In CPS, votes are preloaded in the participant nodes together with a pairwise key associated with the target node. Each node must also store for each vote \( \log m \) authenticating hash values in the Merkle hash tree [12] and the root value of the Merkle tree. Hence in CPS, the space required to store all this information at each node is equal to \( O(m. \log m) \). This space complexity becomes \( O(s_{\text{total}}. m. \log m) \) in the CGPM scheme, since this protocol adds the concept of session and the total number of revocation sessions available is \( s_{\text{total}} \). To further reduce this space complexity, Chao, Yang, Lin, and Li introduce a new distributed revocation protocol in which they use the Blom’s \( t \) secure property instead of the Merkle tree. Thus in every voting session, each node preloads \( t \) elements of one row in the vote matrix and \( t \) elements of one column in the public matrix. Hence the storage space required is \( O(s_{\text{total}}. m. t) \). As \( t \) has a small constant value which is independent of the network size, and \( t \ll m \), the space complexity in CYLL is approximately equal to \( O(s_{\text{total}}. m) \). To put the CCL scheme into work, each node preloads its own private key and the \( m \) hash values \( (h_1, h_2, h_3, ..., h_m) \) of the \( m \) nodes in its key ring, thus generating \( O(m) \) space complexity.

b) Computational Complexity: We propose to evaluate for each scheme the computational complexity by determining both the vote calculation and the vote verification complexity. In the CPS scheme, there is no operation to perform while processing to the vote calculation except counting the number of received votes. Consequently the associated computational complexity is equal to zero. In both CGPM and CYLL, it is required to solve a t-degree linear system to obtain the secret information. Thus the complexity of the vote process calculation is equal to \( O(t^2) \). The CCL scheme needs to compute one XOR operation to obtain the secret information that is Revocation\( \text{CID} \), hence the complexity related to the vote calculation is equal to zero.

Regarding the vote verification process, CPS and CGPM execute the hash operation a total of \( \log m + 1 \) times since every node which receives the vote message verifies it by comparing the root of the Merkle hash tree calculated from the received vote information with the root value held in storage. Thus each vote verification process in these schemes generates a computational complexity equal to \( O(\log m) \). With the CYLL scheme, the vote verification process requires only the execution of a dot product operation involving \( t \) multiplications and \( t - 1 \) additions. As a result, the vote verification mechanism leads to a computational complexity which is just \( O(1) \). With the CCL scheme a node needs to compute one XOR operation to authenticate the Revocation\( \text{CID} \oplus h_i \). Since a single operation is involved, the vote verification in CCL does not produce any computational complexity.

c) Communication Overhead: Here also we consider for each scheme, the communication overhead that relies on the voting process itself and the communication overhead involved by the vote verification process.

In CPS as well as in CCL the voting process generates no communication overhead because the vote messages in these schemes do not include any information which can lead to such overhead. In CGPM, the vote process incurs a communication overhead which is equal to \( O(\log m) \) since each voting message includes the unencrypted vote and the \( \log m \) hash values of the Merkle hash tree that can verify its correctness. With CYLL, this communication overhead decreases substantially and becomes \( O(1) \), due to the fact that voting message carries only \( t \) elements of one row in the vote matrix and \( t \) elements of one column in the public matrix.

In CPS, CGPM, and CYLL, a mask exchange process is required to decrypt the votes against a target node. As a result, the communication overhead related to these solutions for the vote verification process is \( O(m) \). The CCL scheme makes to obtain the zero communication overhead because with this solution, all the elements \( (h_1, h_2, h_3, ..., h_m) \) required to verify the votes are preloaded in nodes and no communication is needed to exchange these data.

To allow an easy comparison and evaluation of the proposed revocation schemes, we present in Table II and Table III, a summary of the results which come from the security and performance study.

IV. CONCLUSION

Key revocation is a critical issue that should receive much more attention to ensure security and robustness of WSNs. Several schemes have been proposed in various papers to highlight the related problems in WSNs and give solutions. Through this survey, we provide an overview of these techniques. We further present a security and performance analysis which shows that each of the proposed schemes offers different advantages and disadvantages. A balance between the requirements and resources of a WSN would help determine which scheme can be employed.
### TABLE II. COMPARISON AND EVALUATION OF CENTRALIZED SCHEMES

<table>
<thead>
<tr>
<th>Centralized Schemes</th>
<th>EG Scheme</th>
<th>KeyRev Scheme</th>
<th>DLS Scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SECURITY ANALYSIS</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attacks resistance</td>
<td>✓ Revocation attack</td>
<td>✓ Revocation attack</td>
<td>✓ Revocation attack</td>
</tr>
<tr>
<td></td>
<td>✓ BS impersonation</td>
<td>✓ BS impersonation</td>
<td>✓ BS impersonation</td>
</tr>
<tr>
<td></td>
<td>✓ Ciphertext analysis attack</td>
<td>✓ Ciphertext analysis attack</td>
<td>✓ Ciphertext analysis attack</td>
</tr>
<tr>
<td></td>
<td>✓ Sybil attack</td>
<td>✓ Sybil attack</td>
<td>✓ Sybil attack</td>
</tr>
<tr>
<td></td>
<td>✓ Node capture attack</td>
<td>✓ Node capture attack</td>
<td>✓ Node capture attack</td>
</tr>
<tr>
<td></td>
<td>✓ Same pattern of attack in network</td>
<td>✓ Same pattern of attack in network</td>
<td>✓ Same pattern of attack in network</td>
</tr>
<tr>
<td>Vulnerabilities</td>
<td>✓ High risk to have the BS impersonate by an adversary (use of a single signature key)</td>
<td>✓ DoS attack</td>
<td>✓ DoS attack</td>
</tr>
<tr>
<td></td>
<td>✓ BS single point of failure</td>
<td>✓ Black hole attack</td>
<td>✓ BS single point of failure</td>
</tr>
<tr>
<td></td>
<td>✓ BS single point of failure</td>
<td>✓ Same pattern of attack in network</td>
<td></td>
</tr>
<tr>
<td>Space requirement</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Computational complexity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>per revocation message</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Per message verification</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Communication overhead</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Per revocation message</td>
<td>(O(m))</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>per message verification</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### TABLE III. COMPARISON AND EVALUATION OF DISTRIBUTED SCHEMES

<table>
<thead>
<tr>
<th>Distributed Schemes</th>
<th>CPS Scheme</th>
<th>CGPM Scheme</th>
<th>CCL Scheme</th>
<th>CYLL Scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SECURITY ANALYSIS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attacks resistance</td>
<td>✓ Impersonation attack</td>
<td>✓ Impersonation attack</td>
<td>✓ Impersonation attack</td>
<td>✓ Impersonation attack</td>
</tr>
<tr>
<td></td>
<td>✓ Node capture attack</td>
<td>✓ Node replication attack</td>
<td>✓ Node capture attack</td>
<td>✓ Node replication attack</td>
</tr>
<tr>
<td></td>
<td>✓ Node capture attack</td>
<td>✓ Node replication attack</td>
<td>✓ Sybil attack</td>
<td>✓ Node replication attack</td>
</tr>
<tr>
<td></td>
<td>✓ DoS attack</td>
<td>✓ De-synchronization attack</td>
<td>✓ Sybil attack</td>
<td>✓ Node replication attack</td>
</tr>
<tr>
<td></td>
<td>✓ DoS attack</td>
<td>✓ De-synchronization attack</td>
<td>✓ DoS attack</td>
<td>✓ Node replication attack</td>
</tr>
<tr>
<td>Vulnerabilities</td>
<td>✓ DoS attack</td>
<td>✓ De-synchronization attack</td>
<td>✓ DoS attack</td>
<td>✓ De-synchronization attack</td>
</tr>
<tr>
<td></td>
<td>✓ Replay attack</td>
<td>✓ De-synchronization attack</td>
<td>✓ DoS attack</td>
<td>✓ De-synchronization attack</td>
</tr>
<tr>
<td></td>
<td>✓ is built on some simplifying assumptions:</td>
<td>✓ is built on some simplifying assumptions:</td>
<td>✓ is built on some simplifying assumptions:</td>
<td>✓ is built on some simplifying assumptions:</td>
</tr>
<tr>
<td></td>
<td>prior knowledge of post deployment position;</td>
<td>prior knowledge of post deployment position;</td>
<td>prior knowledge of post deployment position;</td>
<td>prior knowledge of post deployment position;</td>
</tr>
<tr>
<td></td>
<td>nodes are immobile;</td>
<td>nodes are immobile;</td>
<td>nodes are immobile;</td>
<td>nodes are immobile;</td>
</tr>
<tr>
<td></td>
<td>- presence of functionality to address nodes replication and Sybil</td>
<td>- presence of functionality to address nodes replication and Sybil</td>
<td>- all system clocks are synchronized</td>
<td>- all system clocks are synchronized</td>
</tr>
<tr>
<td>Space requirement</td>
<td>(O(m, \log m))</td>
<td>(O(t^*))</td>
<td>(O(m))</td>
<td>(O(s_{total}, m))</td>
</tr>
<tr>
<td>Computational complexity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>per vote</td>
<td>0</td>
<td>(O(t^*))</td>
<td>0</td>
<td>(O(t^*))</td>
</tr>
<tr>
<td>Per vote verification</td>
<td>(O(\log m))</td>
<td>(O(\log m))</td>
<td>0</td>
<td>(O(1))</td>
</tr>
<tr>
<td>Communication overhead</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>per vote</td>
<td>0</td>
<td>(O(\log m))</td>
<td>0</td>
<td>(O(1))</td>
</tr>
<tr>
<td>per vote verification</td>
<td>(O(m))</td>
<td>(O(m))</td>
<td>0</td>
<td>(O(m))</td>
</tr>
</tbody>
</table>

\(^1\) completeness, soundness, bounded time revocation completion, unitary revocation, and revocation attack resistance
In spite of receiving much attention in recent years, there are still many issues to be addressed in WSNs such as making efficient use of sensor nodes’ limited resources, efficiently detecting and identifying compromised nodes, making nodes tamperproof without much overhead and so on. Thus, the study of key revocation in WSNs still presents a lot of research opportunities in future and more schemes should be designed. Future research works should especially seek techniques for compromised node discovery, because this issue has significant influence on the accuracy of existing key revocation schemes.

ACKNOWLEDGMENT

This work was supported partially by NDC Lab, KICT, IIUM.

REFERENCES


