Pairing Based Mutual Healing in Wireless Sensor Networks

Sarita Agrawal*[1], Jay Patel†[2], and Manik Lal Das‡[3]

[1]*DAIICT, Gandhinagar, India*
[2]*Chaudhary Technical Institute, Gandhinagar, India*
[3]*DAIICT, Gandhinagar, India*

Dated: June 2, 2015

Abstract

In Wireless Sensor Networks(WSNs), a group of users communicating on an unreliable wireless channel can use a group secret. For each session, group manager broadcasts a message containing some keying material, from which only the group members authorized in that session can extract the session key. If a member misses a broadcast message for key, it uses self healing to recover missing session key using most recent broadcast message. However, only self healing does not help if node needs to get most recent session key and have missed the corresponding broadcast. Through mutual healing, a node can request recent broadcast information from a neighboring node and then recover the required key using self-healing. In this paper, we propose a bi-linear pairing based self-healing scheme that reduces communication, storage and computation overhead in comparison to existing bi-linear pairing based self-healing schemes. Then, we discuss the mutual healing scheme that provides mutual authentication and key confirmation without disclosing the node locations to the adversary. The analysis with respect to active adversary shows a significant performance improvement for resource constrained sensor nodes along with the security features such as forward and backward secrecy, resilience against node collusion, node revocation and resistance to impersonation.

Keywords: Wireless sensor networks; Mutual healing; Bi-linear Pairing; Authentication

*Email: sarita_agrawal@daiict.ac.in; Corresponding author
†Email: jay_sp_mca@yahoo.co.in
‡Email: maniklal_das@daiict.ac.in

1 Introduction

Wireless Sensor Network (WSN) is a network of autonomous sensor nodes communicating through wireless medium. The nodes in WSN cooperatively monitor the physical and environmental phenomena and give the information as and when needed to a central authority, known as Base station, which in turn passes on the information to the external users. The main constraint in WSN is that sensor nodes have limited memory, power, computation and communication capability. The WSNs are mostly deployed in unattended hostile environments such as military battle fields, forests, in earth quake prone areas and so on. It is therefore infeasible to look after them after deployment and nodes are exposed to attacks such as node subversion, node capture to read/alter the information contained and even physical destruction. Furthermore, the nodes in WSN communicate via wireless medium, so they are also susceptible to routing attacks such as worm hole and sink hole.

Many WSN applications, for example, health care and military battle field surveillance, need the information within the network to be secured, i.e., only the authorized nodes within the network should be able to send/receive the information. The receiving nodes should be assured of the authentication of the information being received and this authenticated information must be securely communicated within/outside the network as per the application requirement. The secure communication within the WSN may take place on node-to-node basis where a secret is shared between only a pair of nodes, or it can be a secret shared between a group of nodes, which may be done during the deployment or during the network operations. One of the possible approaches of sharing the secret during operations is broadcasting the secret to a group of nodes by a central authority. However, in the broadcast approach, if a node misses out on one or more broadcast then there should be some way to get the information, so that the required key can be extracted. This requirement motivated the researchers for incorporating the idea of self-healing and subsequently mutual healing.

Self-healing allows a node to recover the key of some previous session(s) using the broadcast message received in a subsequent session, without requesting for the key information explicitly from the central authority. Since WSN is a distributed network and the sensor nodes have limited resources wherein the cost of communication is much higher than the cost of computation, sending explicit request to the central authority to obtain the session key information would be an overhead. With the self-healing capability, a node can recover the missing session key itself, thus saving on the communication cost. Mutual healing comes in picture, when a node is not able to recover the key with self-healing because it has lost the last session's broadcast or when it has lost more than one broadcast and does not want to wait for the subsequent broadcasts from the central authority. In such scenario, a node may request its neighboring nodes to provide the missing session key information. However, for mutual healing, the requesting as well as responding nodes must authenticate each other.

In this paper, we review some recent works on mutual healing and self-healing in WSNs. Recently, Tian *et al* [11] proposed the self-healing and mutual healing based on bi-linear pairing. The scheme requires the nodes to perform multiple elliptic curve scalar multiplications and solving algebraic system of equations in order to extract the secret session key. For mutual healing node needs to perform bi-linear pairing operations and discloses its location in mutual healing request/response. We propose an improved scheme over [11]. Our scheme has following characteristics:

1. Self-healing to eliminate the need of re-broadcasting of keying material
2. Mutual healing
3. Security against active adversary
4. Improved performance and additional security over Tian el al's scheme [11].

The remainder of the paper is organized as follows. A brief overview of existing self-healing work in WSN is given in section 2. In section 3, we present the preliminary concepts used in the paper. Section 4 describes the proposed protocol in detail. Security and performance analysis is presented in section 5. We conclude the work in section 6.

2 Background and Related Work

Self-healing group key distribution concept was introduced by Staddon *et al* [1] to address secure group communication in unreliable networks such as in Military applications. The scheme is based on exponential arithmetic with an idea to do interpolation in the exponents. Later Blundo *et al* [3] suggested the simpler and enhanced version of this scheme. The security of these schemes depends on the difficulty of solving Distributed Diffie Hellman (DDH) problem. However, these schemes based on exponential arithmetic suffers from high computation costs and expensive maintenance costs. The technique in Staddon *et al* [1] uses secret sharing based on two dimensional polynomials to distribute group keys, enabling group members to recover lost session group keys as long as they have received one broadcast re-key message before and one after the above session. Liu *et al* [2] generalized the definitions in [1] and developed a novel personal key distribution technique incurring less storage and communication overhead and gave some constructions built upon the technique. Liu *et al* [2] proposed an improved scheme in terms of both memory storage and communication complexity. An advantage of both Staddon *et al* [1] and Liu *et al* [2] techniques is that the computation, communication, and storage overheads required to revoke group members and achieve self-healing capability are independent of the group size, and thus are suitable for very large groups. Moreover, the techniques presented in [2] also deals with coalition of more evicted group members, which is an improvement over the techniques given in [1].

It is important to consider all types of collusion attacks when designing key distribution schemes for WSNs which are subject to node collusion. Yuan *et al* scheme in [4] proposed a self-healing key distribution scheme under a security

model, which provides revocation and participation of at most t nodes, respectively and also provides resilience to collusion of at most $2.t$ nodes. Since the parameter t is considered to be much smaller than the maximum number of sessions (m), Yuan *et al* [4] scheme has less storage and communication overhead than the first two schemes in [1] [2]. In 2005, Hong [5] proposed a self-healing scheme which enabled a user to recover all the keys associated with sessions, in which the user was a member of group, from single broadcast message. Yuan *et al*'s scheme[4] has a bit more communication overhead when compared to Hong's scheme [5], but the same communication complexity (i.e., O(t $\leq$ j $\leq$ log$q$) for j^{th} session broadcast) with a constant factor of 2, as it uses $2.t$-degree polynomials which gives it $2.t$-collusion resistance capability, as opposed to t-collusion resistance capability in [5].

. Yuan *et al* [6] also proposed a self healing scheme that has the unique property of limited group membership. It talks about a fixed number of (t-1) users and thus is bound to (t-1) membership, but, this way, it saves the communication bandwidth by maintaining t users throughout the communication resulting in enhanced quality of service. Furthermore, the scheme [6] eliminates the limitations of revoking at most t members, as it is able to revoke any number of members during the lifetime of the secure group communication. However, their scheme requires twice the memory space and about half the broadcast message size as compared to [5].

Subsequently, Tian *et al* [7] presented a self-healing scheme using access polynomial to reduce communication overhead and computation. However, Wang [9] discovered that the scheme in [7] does not provide forward secrecy and revocation capability. Access polynomial based scheme was also presented by Dutta [10]. However, it is noted that selective key distribution is not performed by the given algorithm and the effect is same as a simple broadcast of session key. Tian *et al* [8] proposed self-healing key distribution scheme based on bi-linear pairing [15], in which the integrity and correctness of the cipher text could be verified by the user before key recovery operations are carried out. The scheme in [8] is free from any collusion of non-authorized users. Unless the private key of any user is disclosed/compromised, it is not affected by whatever number of other users are revoked and the storage overhead for each user is constant. Later, Tian *et al* also proposed the notion of mutual-healing using neighboring nodes in WSN [11] using bi-linear pairing. In [11], for each session, a subset of users is formed as a communication user group by group manager (GM) who broadcasts keying material. From this broadcast message, an authorized session group user can recover the session key of current as well as any previous session. For the construction of a broadcast message, the GM defines a $|G_{j-1}|$ X $|G_j|$ matrix (where $|G_j|$ is the size of the communication group in j^{th} session). During the key recovery process, a node has to compute the matrix inverse operation to obtain a vector of size $|G_j|$ and perform some bilinear pairing operations. Mutual healing is assumed only between one-hop neighbors. Location based secret LK_x is set up for each node u_x using range-based secure localization process with the help of mobile robots, which is used in mutual healing process. A node

(say u_i) upon missing a broadcast message in some session, t, locally broadcasts an authentication request message with its own identity, its location and the session number t of the requested broadcast message B_t, all in plain:
$u_i \rightarrow *$: ID_i, l_i, t

A neighboring node (say u_j), upon receiving such request, verifies the claimed location l_i to be within its one-hop communication range and if so, calculates a shared key $K_{ji} = \mathrm{e}(LK_j, \mathrm{H}(ID_i \| l_i))$ and sends a uni-cast response to u_i's request:
$u_j \rightarrow u_i$: ID_j, l_j, $(B_t)_{K_{ij}}$

On receiving response, u_i performs location verification of neighbor u_j and then u_i calculates the key $K_{ij} = \mathrm{e}(LK_i, \mathrm{H}(ID_j \| l_j))$ shared with u_j and extracts the broadcast message (B_t). This scheme is collusion-free for any coalition of non-authorized nodes, including the revoked nodes and new joined nodes.

Recently, Rams *et al* [12] presented a detailed survey of group-key distribution schemes with self-healing property. In this survey, the authors have classified, analyzed and compared most of the significant group key distribution schemes. According to [12], polynomial based algorithms [10] [4] [6] are simple and efficient, but they disclose some information about pre-distributed user data and also pre-distributed data can not be re-used. Exponent based algorithms such as [1] [3] overcome this weakness, however, they do not provide backward secrecy and also computationally heavy as compared to polynomial based schemes. One-way hash chain based schemes are the most efficient self-healing schemes, but lack the collusion-resistance property. Bi-linear pairing based schemes provide all the three security features required in self-healing schemes, namely, forward secrecy, backward secrecy and collision resistance at the cost of computation overhead in bi-linear pairing calculations. Recently Rams and Pacyna [13] proposed self-healing group-key distribution with extended revocation capability. The scheme requires the nodes to update the personal keys using the update polynomial distributed by the GM. Also, the users need to perform exponential computation for session key computation and need to evaluate polynomial in order to recover the update polynomial broad-casted by GM.

Our Observations. We observed that the bi-linear pairing based self-healing and mutual healing schemes provide the security features needed for group-key broadcast. In the bi-linear pairing based scheme proposed by Tian *et al* [11], the generation of broadcast message and key recovery process are computationally heavy, since the matrix computations and scalar multiplications at node end are involved. A sensor node needs to store two vectors of size $|G_j|$ (the group size in session j) and the public keys of all the nodes in the group. A member node needs to perform a matrix inversion operation and $|G_j|$ scalar multiplications in order to recover the key. During the mutual healing request and response communication, the node location is communicated in plain, which may give way to known-location attacks. As there is no means to ensure the integrity of the location provided by the requesting node, an adversary may update the location or session number shared by the requesting node, resulting in location

verification failure. While the protocol in [11] suggests sending the broadcast message in plain during normal course of operation, it computes a shared key using bi-linear pairing for encrypting the broadcast message only to authenticate the response from the neighbor node. This is a computation overhead both for requesting and responding nodes.

3 The Proposed Protocol

The proposed protocol uses some primitives based on group theory, namely Bilinear Pairing. We first define it and some standard complexity assumptions that we have used in the proposed protocol.

3.1 Bilinear Pairings

Let q be a large prime number. We consider two cyclic groups G_1 (additive) and G_2 (multiplicative) defined over q. A mapping $e{:}G_1 \star G_1 \rightarrow G_2$ satisfies the below mentioned conditions:

1. Bilinearity: $\forall$ P, $Q \in G_1$ and $\forall$ $a,b \in Z_q^*$, $e(aP, bQ) = e(P,Q)^{ab}$
2. Non-Degeneracy: For any point P, $Q \in G_1$, $e(P, Q) = 1$ iff $P = \mathrm{O}$ (point of infinity). Or, in other words, $\exists$ P $\in G_1$ and $Q \in G_1$ such that $e(P, Q) \neq 1$
3. Computability: An efficient algorithm exists to compute $e(P, Q)$ for any P, $Q \in G_1$

Elliptic Curve Discrete Logarithm Problem (ECDLP): Given an elliptic curve point $P \in G_1$ (or G_2) and point $Q = s.P$ for any $s \in Z_q^*$, it is difficult to find the value of s.

3.2 Network Model, Adversary Model and Assumptions

We consider a hierarchical WSN with a base station and a number of cluster heads that serve as group managers, managing a group of nodes. The cluster nodes can communicate within the group through a common cluster key. A new session key is broad-casted by the group manager for each communication session. Nodes may leave a group or join a group or may change the group. Due to this, for each session, the group manager may have a different set of authorized nodes in its group. Each member node gathers the data from the environment according to the application requirement and sends it to its current cluster head. A cluster head aggregates the data received from its cluster nodes and forwards the same to the base station.

We assume the presence of an active adversary, who can passively eavesdrop on the wireless channel and can read and intercept the messages being exchanged. The adversary can also delete, modify or insert the the packets into the network as per his/her attack goal.

Each cluster has a Group Manager (GM) and nodes(users) with unique IDs. There is a finite set of users and, in any given session, a subset of users will be included in the communication group as decided by the GM. The GM would

broadcast a message to this communication group and each member in the group would recover the session key from the received broadcast message. A user can recover a missed communication from a session, if and only if it is an authorized member of that group in the session for which it seeks to obtain the communicated message. We focus on sharing of session key within a group of nodes in order to facilitate self-healing and mutual healing in a secure and cost-effective manner.

3.3 Notations used

The notations used in the scheme are as given in Table 1 below.

Notation	Description
U= $\{u_1, u_2, \ldots, u_n\}$	Finite number of nodes in the network
ID_x	A unique identifier for node x.
GM	Group Manager, sets up and manages a communication group (a dynamic subset of U by adding/revoking the nodes)
m	Number of sessions.
$G_j \subseteq$ U	Communication group formed by GM for j^{th} session G_1 = U, by definition.
$R_j \subseteq G_{j-1}$	Set of nodes revoked from group in j^{th} session.
$J_j \subseteq$ U $\setminus G_{j-1}$	Set of nodes joining the group in j^{th} session s.t. $R_j \bigcap J_j = \emptyset$ and therefore, $G_j = (G_{j-1} \bigcup J_j) \setminus R_j$ for j ≥ 2
K_j	Session key for j^{th} session, selected by GM through random uniform distribution for $j = \{1, 2, \ldots, m\}$
B_j	Broadcast message sent by GM to all member nodes in j^{th} session (B_j and the public-private key pair (Q_x, S_x) are used by a node u_x to recover the session key K_j)
d	$\lvert G_j \rvert$, the size of the group in session j

Table 1: Symbols and Notations used in the Protocol

3.4 The Protocol

The proposed scheme consists of three phases. We first give the system set up and then present the step-wise description of the proposed protocol.

3.4.1 System Set-up phase

Group manager (GM) randomly selects a generator $P \in G_1$ and defines two cryptographic hash functions H_1 and H_2 as follows:

1. H_1: $\{0,1\}^* \rightarrow G_1$
2. H_2: $G_2 \rightarrow \{0,1\}^n$

For public-private key setup, GM selects a random number $s \in Z_q^*$, which is known only to GM and serves as its private key. GM sets its own public key as $P_{Pub} = s.P$ (denotes scalar multiplication of integer s with the elliptic curve point P). The system parameters consist of *params* = (G_1, G_2, q, P, P_{Pub}, H_1, H_2). Each node in the network submits its identity to GM, using which, GM computes node's public key $Q_{ID} = H_1$(ID) and $S_{ID} = s.Q_{ID}$ as private key for the node and sends it to the node securely. The public keys are used by GM in the construction of broadcast message (as explained in subsequent paragraph).

GM selects independent session keys K_1, K_2, ..., K_m for m sessions using a uniform distribution.

3.4.2 Group Key Broadcast

For a session j, GM randomly chooses session key $K_j \in {Z_q}^*$. This session key can be used to communicate within the group for session j. GM now constructs a broadcast message B_j, which is used by the nodes in the communication group to extract the key K_j. Here, GM chooses a random number $r_j \in Z_q$ and construct a broadcast message as follows:

$Q = Q_1 + Q_2 + \ldots + Q_d$
$U = r_j.P_{Pub}$
$U_i = r_j(Q - s.Q_i)$, for $i = 1$ to d
$V_j = K_j \oplus H_2(\mathrm{e}(P_{Pub}, r_j.Q))$
$z_j = (\mathrm{U}, U_i \text{ (for } i = 1 \text{ to } d), V_j)$

GM, then, broadcasts the set $B_j = (z_1, z_2, \ldots, z_j)$ for all the nodes which are the members of the group in the j^{th} session.

3.4.3 Key Extraction

Upon receiving the broadcast message B_j, a node u_x can extract the key by computing the following:

$\mathrm{e}(P_{Pub}, U_x).\mathrm{e}(\mathrm{U}, S_x)$
$= \mathrm{e}(P_{Pub}, U_x).\mathrm{e}(r_j P_{Pub}, \mathrm{s}.Q_x)$
$= \mathrm{e}(P_{Pub}, U_x).\mathrm{e}(P_{Pub}, r_j.s.Q_x)$
$= \mathrm{e}(P_{Pub}, U_x + r_j.s.Q_x)$
$= \mathrm{e}(P_{Pub}, r_j.Q)$

Here (Q_x, S_x) is the public-private key pair of node u_x and P_{Pub} is the public key of GM. Once $\mathrm{e}(P_{Pub}, r_j.Q)$ is computed, node u_x can retrieve the key $K_j = V_j \oplus H_2(\mathrm{e}(P_{Pub}, r_j.Q))$

3.4.4 Self Healing

In the group broadcast scenario, if a node misses any broadcast, it will have to request for that broadcast message again from GM. It is noted that nodes in WSN are energy-constrained and communication is costlier as compared to computation. Self-healing is the process through which a node can extract the key without the GM required to uni-cast the missing broadcast to that node, which allows saving node's energy. In this process of self-healing, if a node u_x misses the broadcast message B_i from session $i < j$ $(1 \leq \mathrm{j} \leq \mathrm{m})$ and it was member of the communication group in session i, then it can use the broadcast message B_j from session $j > i$, pick up the component z_i from B_j (as B_j contains z_i, $1 \leq \mathrm{i} \leq \mathrm{j}$) and use the key extraction process (as explained in previous session) to obtain the missing session key for i^{th} session.

3.4.5 Mutual Healing

Mutual healing is needed when a node misses more than one broadcasts. In that case, it needs to wait for the latest broadcast to retrieve the keys of the sessions for which it was a member. Mutual healing becomes inevitable when the node misses the current broadcast and can not recover the current key. In this approach, similar to Tian *et al* [11] scheme, we assume that the sensor nodes are stationary after deployment and mobile robots compute the location l_x for each node u_x in the network, using a secure localization process such as in [16] and securely communicates the same to the nodes. Once the localization process is completed, the mobile robots leave the network. This location information will be used in the neighbor authentication process during mutual healing. The mutual healing phase has two sub-phases, which work as follows:

Mutual Healing Request. When a node u_x needs the broadcast message B_t, it broadcasts an authenticated request to the nodes in its local neighborhood:

$u_x \rightarrow *$: ID_x, $\{l_x\}_{K_c}$, t, c,
$\mathrm{PRF}(K_c, ID_x \| l_x \| t \| c)$

Here, ID_x - ID of requesting node u_x
l_x - location of u_x
K_c - Group key in some previous session c,
of which u_x was member
t - the session for which broadcast message is
being requested
PRF() - Pseudo Random function

Mutual Healing Response. A node u_y, which receives this Mutual healing request, first confirms that the request is from an authenticated node which is in its one-hop communication range. Node u_y verifies $|l_x - l_y| \leq R$ (Here, R is assumed to be one-hop communication distance between two nodes in the sensor network, based on the communication range of the nodes). If the location verification fails, node u_y simply discards the request else it uni-casts a reply to node u_x as follows:

$u_y \rightarrow u_x$: ID_y, $\{l_y\}_{K_c}$, z_t, N_y, $\mathrm{PRF}(K_c, ID_x \| ID_y \| l_x \| l_y \| z_t \| N_y \| t \| \mathrm{c})$

$u_x \rightarrow u_y$: ID_x, $\{N_y + 1\}_{K_t}$, $\mathrm{PRF}(K_t, ID_x \| ID_y \| (N_y + 1))$

In this protocol, the responding node u_y sends the t^{th} component z_t of the broadcast message B_t meant for the requested session t, along with its own ID and encrypted location. The node need not to send the entire broadcast message B_t, since the requesting node has specified which broadcast message it has missed, so it suffices to send the keying material specific to that session i.e. z_t. It also sends a nonce N_y which is used by the requesting node u_x to confirm the reception of the message and recovery of the required key K_t. The nodes must authenticate each other to ensure that the keying material is being requested/sent by a valid user. For authentication purpose, the protocol uses a keyed Pseudo Random Function (PRF) such as HMAC [14] of all the transmit-

ted values using a key shared by the group in some previous session c. Node u_x shares a previous session ID whose key is used for generating the PRF() for the request message. When node u_y receives the message, it computes the PRF() using the corresponding key K_c and is assured of the authenticity of node u_x. Similarly, u_x authenticates u_y by calculating the PRF() of the response message sent by node u_y. Here, we assume that the requesting and responding nodes share at least one session key from some previous session to complete this protocol.

Once, requesting node u_x receives the broadcast message component z_t, it recovers the key for session t, using the key extraction process. Then, it sends $(N_y + 1)$ encrypted by the newly recovered key K_t back to responding node u_y to confirm the receipt of the the correct message and key.

3.4.6 Adding and Revoking Node

Suppose a node u_{new} willing to join session j requests GM for including it in the communication group. It sends its ID, ID_{new}, and the session ID, j, the session from which it is interested to join the group. GM validates the identity of u_{new} by looking at the list of non-revoked nodes and if finds node u_{new} to be an authorized node, GM includes the new node's public key in the computation of Q and also computes U_i for this new node while preparing for the broadcast message i.e.

$$Q = Q_1 + Q_2 + \ldots + Q_{new} + \ldots + Q_d$$
$$U = r_j.P_{Pub}$$
$$U_i = r_j.(Q - s.Q_i), \text{ for } i = 1 \text{ to } d$$
$$\text{(including } i = new\text{)}$$
$$\vdots$$

When a node, say u_{rov}, is revoked from the j^{th} session, GM excludes its public key Q_{rov} in computation of Q and, U_i value corresponding to this revoked node u_{rov} will not be included in the broadcast message i.e.

$$Q = Q_1 + Q_2 + \ldots + Q_d,$$
$$\text{(excluding } Q_{rov}\text{)}$$
$$U = r_j.P_{Pub}$$
$$U_i = r_j.(Q - s.Q_i), \text{ for } i = 1 \text{ to } d$$
$$\text{(excluding } i = rov\text{)}$$
$$\vdots$$

Neither node u_{rov}'s public key is used in the computation of Q in the subsequent sessions, nor is U_i computed for $i = rov$. Therefore, even if u_{rov} receives the broadcast message B_t for a session $t \geq j$, it would not be able to extract the key K_t for session t.

The newly joined node can participate in the j^{th} session and subsequent sessions (if he is chosen as the valid member in the group in a particular session). A revoked user would not be able to participate in the j^{th} session and any subsequent sessions (unless it is joined again by GM)

4 Analysis

4.1 Assumptions and Adversarial Model

GM is a trusted entity and the public-private key pairs are assigned by GM to all the nodes in the network through a secure channel before deployment. Each node needs to keep only its own public-private key pair. A node is not given access to the public keys of other nodes in the network. We assume that each node in the network knows its location once the secure localization protocol is carried out immediately after the deployment. During the network operations, the nodes do not disclose their locations to each other unless mutual healing request-response protocol is executed. The nodes delete the information shared during the mutual healing. Adversary has access to the public wireless communication channel and can intercept, inject or modify messages.

4.2 Security Analysis

4.2.1 Forward Secrecy and Backward Secrecy

Forward secrecy ensures that a session key derived from a set of long-term keys will not be compromised if one of the long-term keys is compromised in the future. Backward secrecy ensures that compromise of a session key does not reveal the past session keys or long-term keys. In the context of group key distribution [12], forward secrecy ensures that any number of users revoked before or in session j colluding together can not recover any of the future session keys K_j, K_{j+1}, ..., K_m. Similarly, backward secrecy implies that any number of users joining after session j colluding together can not recover any of the past session keys K_2, K_3, ..., K_{j-1}.

Theorem 1: The proposed scheme provides forward and backward secrecy.

Proof: In order to prove forward secrecy, we need to show that a user revoked in a session j can not compute the future session keys, which means a user $u_x \in R_j$ can not compute a key K_w(j ≤ w ≤ m). Let us look at the process of generating the broadcast message by GM for the session w. GM takes the public keys of only those users who are authorized members of w and computes Q as:

$Q = \sum_{i=1}^{d} Q_i$, for $u_i \in G_w$

GM, chooses the session specific random number r_w and computes

$U = r_w.P_{Pub}$, for $r_w \in_R Z_q^*$

$U_i = r_j(Q - sQ_i)$, for $i = 1$ to d,

where $s \in_R Z_q^*$ - private key of GM

GM, then picks a random key K_w for session w and computes

$V_w = K_w \oplus H_2(\mathrm{e}(P_{Pub}, r_w.Q))$

$z_w = (\mathrm{U}, U_i, V_w)$, for $i = 1$ to d

To obtain the session key for session w, a node needs to have access to the broadcast message B_w or the component z_w from some broadcast B_t ($w \leq t \leq m$). We note that, broadcast message is available to any user who is within the communication range of GM during that broadcast.

With the availability of component z_w, the values V_w, U, U_i for (i=1 to d, $u_i \in G_w$) are available to any user having access to component z_w. The session key K_w can be extracted as follows:

$K_w = V_w \oplus H_2(e(P_{pub}, r_w.Q))$

However, for obtaining the expression $e(P_{pub}, r_w.Q))$, the user needs to know r_w, which is ephemeral secret for session w known only to GM. ... (1)

Furthermore, for a user u_i, authorized for session w, its public key component Q_i is used in the generation of z_w, so u_i can compute the expression $e(P_{pub}, r_w.Q)$ as

$e(P_{pub}, r_w.Q)$
$= \mathrm{e}(P_{Pub}, U_i).\mathrm{e}(\mathrm{U}, S_i)$
(S_i is private key of u_i)
$= \mathrm{e}(P_{Pub}, U_i).\mathrm{e}(r_w.P_{Pub}, \mathrm{s}.Q_i)$
$= \mathrm{e}(P_{Pub}, U_i).\mathrm{e}(P_{Pub}, r_w.sQ_i)$
$= \mathrm{e}(P_{Pub}, U_i + r_w.sQ_i)$
$= \mathrm{e}(P_{Pub}, r_w.Q)$ ... (2)

An unauthorized user u_x can not compute $e(P_{pub}, r_w.Q)$ in this manner, because, its public key is not used in the computation of Q. U_i computed using private key of u_x is also not part of the broadcast message for this session w.

Another possibility for an unauthorized user u_x is to obtain r_w in order to solve the expression $\mathrm{e}(P_{Pub}, r_w.Q)$. Since, U and P_{pub} are known publicly, user u_x may attempt to extract r_w using $U = r_w.P_{Pub}$. However, due to the hardness of Elliptic Curve Discrete Logarithm Problem (**ECDLP**), given U ($= r_w.P_{pub}$) and P_{pub}, it is not possible to obtain r_w. ... (3)

Since a user $u_x \in R_j$ can neither obtain U_x for a session w nor can it directly compute $\mathrm{e}(P_{pub}, r_w.Q)$, the expression $H_2(e(P_{pub}, r_w.Q))$ could not be solved and hence extraction of K_w is not possible for any user $u_x \in R_j$.

For backward secrecy, a user joining the group after session j should not be able to compute any of the past session keys, that is, a user $u_x \in J_{j+1}$ can not compute a key $K_w (2 \leq w \leq j)$. The proof is on similar lines to that for forward secrecy.

Hence, the proposed scheme provides both forward and backward secrecy. □

Example: Let U = u_1, u_2, ..., u_{10} is the universe of users in the given WSN. $G_j = u_1, u_2, u_4, u_6$ is the set of 4 users who form the communication group in j^{th} session i.e. $|G_j| = d = 4$. Thus for j^{th} session: $\mathrm{Q} = Q_1 + Q_2 + Q_4 + Q_6$. U_is (for i = 1, 2, 4 and 6) will be $r_j.(Q - s.Q_1)$, $r_j.(Q - s.Q_2)$, $r_j.(Q - s.Q_4)$ and $r_j.(Q - s.Q_6)$. Now suppose node u_3, who is not part of the communication group in this session j, attempts to extract the key K_j from the broadcast message B_j. Node u_3 proceeds to compute $\mathrm{e}(P_{Pub}, r_j.\mathrm{Q}) = \mathrm{e}(P_{Pub}, U_x).\ \mathrm{e}(\mathrm{U}, S_x)$. However, $U_x = r_j.(Q - s.Q_3)$ corresponding to node u_3 is not available in the broadcast message for session j. Therefore, node u_3 will not be able to extract key K_j. Furthermore, suppose node u_3 was part of some i^{th} session (for $i \leq j$), so, $U_x = r_i.(Q - s.Q_3)$ is available to u_3, but since $r_i \neq r_j$ (for any session id

pair i, j), u_3 still can not compute e(P_{Pub}, r_j.Q) = e(P_{Pub}, U_x). e(U, S_x) and extract K_j. The same argument can be given for any session t after j^{th} session. We can thus conclude that the proposed protocol provides both forward and backward secrecy.

4.2.2 Collusion Resistance

Any number of users revoked before a session j and joined after session t ($j \leq t$) colluding together can not recover session keys K_j, K_{j+1}, ..., K_t.

Theorem 2: The proposed scheme is resistant to collusion of nodes revoked from session j and nodes joined after a session t.

Proof: We need to show that users $u_x \in R_j$ and $u_y \in J_{t+1}$ colluding together can not compute the key K_w(j ≤ w ≤ t).

According to Theorem 1 above, the revoked users can not compute the future session keys and and the newly joined users can not compute the past session keys. The key for the session w can be extracted only by a user which is an authorized member in G_w. No user $u_x \in R_j \cup J_{t+1}$ is a valid member in any group G_w(j ≤ w ≤ t). So, even if all of the members in $R_j \cup J_{t+1}$ collide together, they still don't have a valid input to resolve the expression $H_2(e(P_{pub}, r_w.Q))$ to extract the required session key K_w. □

4.2.3 Resistance to Impersonation

Impersonation attack implies that an unauthorized user or adversary claims to be a valid member of the group.

Theorem 3: The proposed protocol is resistant to impersonation attack.

Proof: An adversary can claim to be a valid member of a session w, if he can get hold of the session key K_w. The session key is distributed by GM through a broadcast message B_w which has the component z_w constructed using the public and private keys of only the authorized members of the session w and therefore, only those members can get hold of the key K_w using defined key extraction process. An adversary may attempt to solve the expression $H_2(e(P_{pub}, r_w.Q))$ for which it needs to know the random secret r_w. Using the available information $U = r_w.P_{Pub}$ and P_{Pub} adversary can try to extract r_w. However, this corresponds to solving ECDLP which is a hard problem. Thus, an unauthorized user/adversary can not solve it and obtain key K_w.
Hence, an unauthorized user/adversary can not claim to be a valid member of the group. □

4.2.4 Node Revocation Capability

Node revocation implies that a revoked node is not able to obtain future session keys from any future broadcast message.

Theorem 4: The proposed protocol provides node revocation capability.

Proof: A node u_{rov} revoked in j_{th} session may attempt to obtain the session for any session $t \geq j$. Suppose the node u_{rov} receives the broadcast message for a session t. Thus, u_{rov} has the component z_t from which it can extract session key K_t. To extract key K_t, the node u_{rov} needs to compute (Ref subsection: Key Extraction)

$$\mathrm{e}(P_{Pub}, U_{rov}).\mathrm{e}(U, S_{rov})$$

Node u_{rov} has P_{Pub}, the public key of GM, U from z_t, and S_{rov} as its own private key. However, U_{rov} is not available to u_{rov}, because during the construction of broadcast message B_t, neither Q_{rov} (the public key of u_{rov}) is included in computation of Q, nor the component U_{rov} is added. So, node u_{rov} can not extract K_t.
Hence proved that revoked node does not have access to future session keys. □

4.2.5 Secured Node Location

In the mutual healing process given by Tian *et al.* [11], a node sends out its location ID in plain while requesting a missing broadcast message from its one-hop neighbors. This may lead to known location attacks, as the node location is easily available to the adversary. In our proposal, unlike the Tian *et al* [11] scheme, the node location is never transmitted in plain and the knowledge does not remain with any other node once the mutual healing process is completed. The proposed scheme does not disclose a node's location to an outside adversary. We assume all nodes in the network know their respective location through some secure localization scheme [16]. In the mutual healing process, a node who needs a missing broadcast message, broadcasts a request for the same to its one hop neighbors:

$$u_x \rightarrow *\text{: } ID_x, \{l_x\}_{K_c}, t, c, \mathrm{PRF}(K_c, ID_x \| l_x \| t \| \ c)$$

Since the location of the requesting node u_x is encrypted with session key K_c of some previous session c, only a one-hop neighbor of u_x who was a member in session c will be able to get the location information of node u_x.
A node having the requested broadcast message uni-casts the message to the requesting node:

$$u_y \rightarrow u_x : ID_y, \{l_y\}_{K_c}, z_t, N_y, \mathrm{PRF}(K_c, ID_x \| ID_y \| l_x \| l_y \| z_t \| N_y \| t \| \ c)$$

Again, due to the encryption of the location of node u_y, location u_y is not available to any outside adversary during this communication. Once, the mutual healing process is completed, nodes delete the locations of each other. Hence, the location information of participating nodes will never be available to any outside adversary.

4.2.6 Mutual Authentication

Two nodes participating in a mutual healing session can mutually authenticate each other, since the communication between two parties contain code generated from a common secret using PRF().

In the proposed protocol, the missing broadcast request message and the unicast response message are as follows:

$u_x \rightarrow *$: ID_x, $\{l_x\}_{K_c}$, t, c, PRF$(K_c, ID_x \| l_x \| t \|\ c)$

$u_y \rightarrow u_x$: ID_y, $\{l_y\}_{K_c}$, z_t, N_y, PRF$(K_c, ID_x \| ID_y \| l_x \| l_y \| z_t \| N_y \| t \|\ c)$

Both the messages do contain PRF code of the transmitted information that is generated using the session key of some previous session in which both requesting node u_x and responding node u_y were part of the communication group. When a node receives the PRF code, it generates its own PRF using secret K_c and verifies the code. Since the session key K_c is not available to an adversary, the node is assured that the message has been sent by some node who is valid group member in session c. Thus, apart from verifying the one-hop distance as in [11], the nodes are able to mutually authenticate each other.

4.2.7 Key Confirmation

In a mutual healing session, the responding node can confirm if the valid key is received at the requesting end. Requesting node u_x receives the component z_t of missing broadcast message B_t, recovers the required session key K_t and sends the incremented nonce encrypted with the key K_t, it had recovered for the requested session t, back to the responder node u_y:

$u_x \rightarrow u_y$: ID_x, $\{N_y + 1\}_{K_t}$, PRF$(K_t, ID_x \| ID_y \| (N_y + 1))$

When u_y gets back the correct incremented nonce after decryption using the same key K_t, it is assured that the correct message has been received by the requesting node u_x. This ensures the key confirmation.

Table 2 below gives the comparison of major security features in Tian *et al*'s scheme [11] and the proposed scheme.

Attributes → Schemes ↓	Forward and Backward Secrecy	Mutual Authentication	Key Confirmation (In Mutual Healing)	Secure Node Location
Tian *et al* [11]	Yes	Yes	No	No
Proposed Scheme	Yes	Yes	Yes	Yes

Table 2: Comparison of Security Features with Tian's scheme [11]

4.3 Performance Analysis

4.3.1 Computation cost

A node after receiving the broadcast message, takes the j^{th} component and for computing e(P_{Pub}, r_t.Q), it needs to perform two bi-linear pairing operations, one for e(P_{Pub}, U_x) and other for e(U, S_x). Then it computes the hash

$H_2(\mathrm{e}(P_{Pub},\ r_t.\mathrm{Q}))$ and finally extracts key K_j using an XOR operation. The hash and XOR computation cost is negligible in comparison to the bi-linear pairing computation. So, if T_p is the cost of bi-linear pairing computation, then self-healing and key-recovery operation takes only 2 * T_p. In Tian *et al*'s [11] proposal, GM needs to define a $|G_{j-1}|$ X $|G_j|$ matrix and compute $|G_{j-1}|$ additional ECC points using public keys of the members of current communication group, in order to construct the broadcast message. A node in the group needs to perform matrix inversion operation. Secondly, during the mutual healing process, the responding node encrypts the requested broadcast message with a key generated from the location based key using bi-linear pairing operation. The requesting node needs to calculate the same key again using bi-linear pairing operation. In our proposal, we avoid the need of any matrix, additional ECC points computations by GM, matrix inversion operation by node and bi-linear pairing operations for authentication during mutual healing. For mutual healing, one symmetric key encryption, and one PRF() computation for request message, one symmetric key decryption, one simple comparison to check euclidean distance and one PRF() to verify response, and one symmetric key encryption for key confirmation. In case node is responder, then it needs one symmetric key decryption, one simple comparison to check euclidean distance and one PRF() to verify request, then for response one symmetric key encryption, and one PRF() computation. When compared with Tian's scheme [11], we find that computation cost is greatly reduced, as now a node does not require to solve system of linear equations and also it could avoid doing scalar multiplication with respect to all other nodes in the group in order to recover a key.

4.3.2 Communication cost

The group manager broadcasts one single message in each session. So, for m sessions, there will be total m broadcast messages. In each session j, $j(2d+4)\log q$ bits are transmitted as B_j, which is similar to Tian's scheme [11]. At node end, node receives a broadcast message in each session and in usual operation, it does not need to transmit anything for obtaining the group secret key. When a node needs to obtain the group key information through mutual healing, it needs to send out two messages, one broadcast message as mutual healing request and another key confirmation message to the responding node. The overall bits communicated in the process is much less as compared to Tian's scheme, with an additional security feature of key confirmation. Since the requesting node is specifying the session number t for which it has missed the broadcast message, it implies that the requesting node had received all the previous messages, so the responding node need not to send the whole broadcast message, but can only send the t_{th} component i.e. z_t from the broadcast message B_t.

4.3.3 Storage cost

Each node stores its own public-private key pair and the group session keys for maximum m sessions. It also needs to temporarily store the broadcast message, which contains j ($1 \leq j \leq m$) entities. We are able to save on the cost of storing a $\|G_j|$ x $\|G_j|$ matrix and also a node does not need to store the public keys of all the nodes in the group to recover a session key, unlike in [11].

The comparison of performance parameters (overhead in bits) is presented in Table 3 below. In this table,

j: session number ($1 \leq j \leq m$)
d: No. of users in the group in session j
T_p, T_s: Time for Pairing and Scalar multiplication respectively
T_h, T_m: Time for Hash and PRF respectively
T_e, T_d: Time for Symmetric Encryption and Decryption respectively

Since the pairing operation is costliest in comparison to other operations, we

Schemes →	Tian *et al.* [11]		Proposed Scheme	
Attributes ↓	Self Healing	Mutual Healing	Self Healing	Mutual Healing
Storage Overhead	$(2jd + 2j + 5)\log q + 2d\log q + 16(d^2+d) + 16$	$2\log q$ (for location based key)	$(2jd + 2j + 5)\log q + 16$	0
Communication Overhead	0	Requesting Node: $3\log q$	0	Requesting Node: $8\log q$
		Responding Node: $((2d+3)j+2)\log q$		Responding Node: $((2d + 3) + 4)\log q$
Computation Overhead	$2T_p + T_h + dT_s$	Requesting Node: $T_p + T_d + T_h$	$2T_p + T_h$	Requesting Node: $2T_e + 3T_m + T_d$
		Responding Node: $T_p + T_e + T_h$		Responding Node: $T_e + 3T_m + 2T_d$

Table 3: Comparison of Performance with Tian *et al*'s scheme [11]

have eliminated the use of pairing during mutual healing. Instead, we are using only encryption, decryption and PRF computation operations which take few microseconds to perform [18]. Also, for self-healing, we have reduced the computation time by eliminating the need to perform d scalar multiplications. This reduces the self-healing computation overhead significantly. For example, if we consider an average group size of 10 nodes and the TinyPairing [17] implementation of bilinear pairing and related operations for WSN with finite field GF(3^{97}), in which, the pairing operation takes 5.32 seconds and scalar multiplication takes 2.45 seconds, the proposed scheme reduces self-healing computation time by 2.45*10 (=24.5)seconds.

5 Conclusion

In order to secure the communication for information exchange in WSN, base station and nodes in WSN need to share some secret key. Broadcasting the keying material containing secret to a group of nodes by a central authority is one of the approach used for this purpose. However, in this process, there is a possibility that a node misses out on one or more broadcast. The system needs to have some way to get such missing information in order to extract

the required key. Self-healing and subsequently mutual healing is incorporated to achieve this purpose. In this work, we have proposed a simplified approach for generating the broadcast information and key extraction from the keying broad-casted material. In this approach, the central authority, group manager, generates a message involving the session key and the public keys of the member nodes in current session. It uses bi-linear pairing operation for generation of keying material and later extraction of key from the same in an efficient manner. During mutual healing, the scheme ensures secrecy of the node locations by keeping the location information out of reach of the adversary, while sharing the missing broadcasts for mutual healing. Our protocol not only provides the forward and backward secrecy, node revocation capability and resilience to node collusion, but gives additional security features such as resistance to impersonation attack and secrecy of node location and message confirmation with authentication in mutual healing. These additional features are provided along with reduced storage, computation and communication overhead, which is a significant achievement in WSN environment.

References

[1] J. Staddon, S. Miner, M. Franklin, D. Balfanz, M. Malkin, and D. Dean. Self-healing key distribution with revocation. In: IEEE Symposium on Security and Privacy, Vol 0, Pages 241 - 257, 2002.

[2] D. Liu, P. Ning, and K. Sun. Efficient self-healing group key distribution with revocation capability. In Proceedings of 10th ACM conference on Computer and communications security, Pages 231 240, 2003

[3] C. Blundo, P. Darco, A. De Santis, and M. Listo. Design of self-healing key distribution schemes. Des. Codes Cryptography, Vol. 32, Pages 15 44, 2004.

[4] T. Yuan, J. Ma, Y. Zhong, and S. Zhang. Self-healing key distribution with revocation and collusion resistance for wireless sensor networks. In Proceedings of International Multi-symposiums on Computer and Computational Sciences, Pages 83 90, 2008.

[5] D. Hong and J. S. Kang. An efficient key distribution scheme with self-healing property. In: IEEE Communication Letter, Vol 9, No 8, Pages 759 761, 2005.

[6] T. Yuan, M. Jianqing, Y. Zhong and S. Zhang. Self-healing key distribution with limited group membership property. In Proceedings of First International Conference on Intelligent Networks and Intelligent Systems, Pages 309 312, 2008.

[7] B. Tian, S. Han and T. Dillon. An Efficient Self-Healing Key Distribution Scheme. In Proceedings of International Conference on New Technologies, Mobility and Security, Pages 1 - 5, 2008.

[8] B. Tian, E. Chang, T. Dillon, S. Han and F. Hussain. An authenticated self-healing key distribution scheme based on bi-linear pairings. In Proceedings of the 6th IEEE Conference on Consumer Communications and Networking Conference, Pages 1061 - 1065, 2009.

[9] H. Wang. On the Security of Some Self-healing Key Distribution Schemes. In Proceedings of the International Conference on Multimedia Information Networking and Security, Pages 777 - 780, 2010.

[10] R. Dutta, S. Mukhopadhyay, and T. Dowling. Enhanced access polynomial based self-healing key distribution. In Security in Emerging Wireless Communication and Networking Systems, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol.42, Pages 13 24, 2010.

[11] B. Tian, S. Han, J. Hu and T. Dillon. A mutual-healing key distribution scheme in wireless sensor networks. In: Journal of Network and Computer Applications, Vol. 34 Issue 1, Pages 80 - 88, 2011.

[12] T. Rams and P. Pacyna. A Survey of Group Key Distribution Schemes With Self-Healing Property. In IEEE Communications Surveys and Tutorials, Vol.15, No.2, Pages 820-842, 2013.

[13] T. Rams and P. Pacyna. Self-healing Group Key Distribution with Extended Revocation Capability. In the proceedings of International Conference on Computing, Networking and Communications, Communications and Information Security Symposium. Pages 347 - 353, 2013.

[14] H. Krawczyk, M. Bellare and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. Request for Comments: 2104, 1997. (http://www.ietf.org/rfc/rfc2104.txt)

[15] Alfred Menezes. An introduction to pairing-based cryptography. In Recent Trends in Cryptography, Vol. 477 of Contemporary Mathematics, AMS-RSME, Pages 47 - 65, 2009.

[16] L. Lazos and R. Poovendran. Serloc: secure range-independent localization for wireless sensor networks. In the proceedings of the 3rd ACM Workshop on Wireless Security (WiSe), Pages 21 30, 2004.

[17] X. Xiong, D. Wong and X. Deng. TinyPairing: A Fast and Lightweight Pairing-based Cryptographic Library for Wireless Sensor Networks. In the proceedings of IEEE Wireless Communications and Networking Conference (IEEE WCNC10), Pages 1-6, 2010.

[18] C. Karlof, N. Sastry and D. Wagner. TinySec: a link layer security architecture for wireless sensor networks. In the proceedings of the Sensys, 2004.