A Comparison on Broadcast Encryption Schemes: A New Broadcast Encryption Scheme

Hüseyin BODUR, Resul KARA
Department of Computer Engineering, Engineering Faculty, Düzce University, 81620, Turkey
huseyinbodur@duzce.edu.tr

***Abstract*—In broadcast communication, the schemes in which encryption methods are employed are often used to transmit messages from a source to multiple users. There are three types of scheme categories: central, contributory and hybrid schemes. In this work, three central techniques which are widely used nowadays are discussed: Logical Key Hierarchy (LKH), One-way Function Tree (OFT) and One-way Function Chain (OFC). A new central broadcast scheme has also been proposed. This scheme uses an asymmetric encryption method in the root node and the user nodes and, as opposed to other central schemes, a symmetric encryption method as well. It contains a hash() function in the intermediate node calculations. The proposed scheme was compared with the existing schemes in terms of number of key transmissions, number of operations, number of user keys, size of user keys, and size of key transmission in user adding/removing and batch user adding/removing operations. The results are shown graphically.**

***Index Terms*—cryptography, public key, random number generation, data security, digital signatures.**

I. INTRODUCTION

The many current schemes which aim to transmit messages from one source to multiple users are referred to collectively as broadcast communication. These schemes are usually based on a binary tree. The broadcast center (BC) is located in the root node and the users are in the leaves.

In broadcast communication, encryption methods are generally used to transmit messages to multiple users. A message is encrypted by the BC in the root node and transmitted to users. Forward secrecy means that a user that quits the broadcast scheme should not be able to access future broadcast messages [1-2]. Backward secrecy means that a user that joins the broadcast scheme should not be able to access past broadcast messages [1], [3]. Some key updates in a scheme must be performed after each membership change in order to provide forward and backward secrecy. Otherwise, security problems will occur in the scheme.

The three types of scheme categories include: central, contributory, and hybrid schemes. The most important difference between these schemes is the presence of an entity called the Key Server (KS) in central schemes. The KS is responsible for the generation and distribution of keys and rekeying to ensure group communication. There is no KS in contributory schemes, and operations are distributed to and performed by the users. Hybrid schemes include the KS, but some of its tasks are performed by the users.

Central schemes must be scalable in terms of communication, computation and storage costs. They must provide forward and backward secrecy and collusion independence. In addition, they must handle multiple membership changes. The handling of these issues by various central schemes has been analyzed. The results, in addition to new central schemes that have been proposed, are available in the literature. A review of some central schemes, in particular the LKH, OFT and OFC, is presented here.

In one study, Shanu and Chandrasakaran pointed out that the most important processes in the LKH structure are re-keying to ensure forward and backward secrecy after user addition/removal and using a distribution function to reduce the re-keying operation costs [4]. In another study, Prathap and Vasudevan analyzed various schemes and proposed a new hybrid method by combining the advantageous properties of these schemes for user addition/removal operations [5]. Sakamoto et al. proposed a scheme to reduce the distance from the users to the root node in the LKH tree. They used the Huffman algorithm in their proposed scheme [6].

In their study, Gu et al. proposed an effective key management scheme called Key Tree Reuse (KTR). As a new key management approach, the KTR allows users to register with multiple programs in the broadcast system using the same key value. Although it is LKH-based, its re-keying costs were lower than with the LKH scheme [7]. In another study, Song et al. proposed a new group key management algorithm based on a public key (asymmetric) infrastructure to enable encrypted cloud data sharing with dynamic groups. Data security was provided through the proposed scheme by taking advantage of public-key encryption, even if exposed to attacks by malicious users on the cloud server [8].

Alyani et al. endeavored to apply the Diffie-Hellman key exchange on the LKH structure and by changing the LKH structure, to improve the key management scheme. This change was based on boosting its performance by increasing the number of users in the subsets of the tree [9]. In their study, Liu et al. proposed a new tree structure based on an intuitive search algorithm to reduce the cost of key updates performed after adding / removing users. The study also used a different number of nodes at each level of the LKH tree [10].

Sakamoto, in his study, maintained that the cost of the key update could be reduced if the average number of users added to or removed from a key tree were known [11].

In one study, the Diffie-Hellman key exchange was used to solve the problems encountered during the implementation of the LKH scheme to a Nosql database on

This work was supported by the Tubitak 2211-C Domestic Priority Areas Doctoral Scholarship Program.

Digital Object Identifier 10.4316/AECE.2020.04009

a cloud server [12]. In another study, the security weaknesses of the OFT scheme were examined. Two improved OFT schemes called repeated OFT (ROFT) and node OFT (NOFT) were proposed. Unlike OFT, these ROFT and NOFT schemes did not require extra communication costs in group management [13]. The weakness of the OFT scheme against collusion attacks was addressed in another study, and a new scheme was proposed by adding a method to the OFT scheme which aimed to prevent collusion attacks with a minimum broadcast size after random user-adding/removing operations in the scheme [14].

In their study, Hwang and Sung gave information about micro and macro payment systems and a new micro payment scheme based on the elliptical curve encryption method was proposed. The proposed scheme was based on the OFC [15].

In their study, Lee et al. proposed a scheme called TARD to reduce the cost of the key management occurring with temporary access to guest devices within a network. The TARD scheme utilized the OFC scheme for the secure transmission of a cryptographic token [16]. Benmalek and Challal proposed a new multi-group key management scheme to ensure that the Advanced Metering Infrastructure (AMI), as the main component in smart grids, was resistant to cyber-attacks. They adopted a variation of OFC in their solution [17].

Chen and Tzeng proposed two Group Key Management (GKM) schemes, KeyDer-GKM and ReEnc-GKM, to reduce the cost of computation and storage for a large and highly dynamic group of members where the rekey process was performed frequently [18].

Benmalek et al. proposed four key management schemes to be used for an AMI in the smart grid. These were based on an asymmetric group key agreement such as Elliptic-Curve Diffie-Hellman (ECDH) and support unicast as well as multicast communication [19].

Kumar et al. introduced a Centralized Group Key Distribution (CGKD) protocol, based on a key star structure that reduced the computation and storage cost of KS during key updating. In addition, an extended CGKD protocol based on a clustered tree was proposed for large-sized groups [20].

In another study, Hanatani et al. proposed a technique which has been recently standardized in IEEE 802.21. The technique was based on LKH but used a 'Complete Subtree' for the number of encryption and decryption operations [21].

In their study, Elhoseny et al. proposed a scheme for secure data transmission in the Wireless Sensor Network (WSN). It used Elliptic Curve Cryptography (ECC) and Homomorphic Encryption (HE) methods and improved various aspects of the network performance such as memory requirements, energy consumption, lifetime and communication overhead [22].

Lin et al. proposed a Cluster-based Elliptic Curve Key Management (CECKM) scheme to ensure secure group communications in WSNs. The proposed scheme provided more efficient and rapid group key resynchronization time compared to the Diffie-Hellman and RSA cryptosystems [23].

Islam and Biswas introduced a pairing-free identity-based two-party authenticated key agreement protocol using ECC. The proposed protocol enabled two parties to generate a common secret key between them and was suitable for secure and efficient peer-to-peer communications [24].

In another study Chaudhari et al. selected a group of centralized group key management schemes for a dynamic network of sensors and analyzed them under strong active outsider attacks. They pointed out that each sensor node should have an individual secret key so that existing schemes could be secure and that n secure channels were required for a group of n sensors [25].

In their study, Hur and Lee proposed a w-session Reliable Group Key management (w-RGK) scheme that allowed users to recover the current group key to update messages, even if they had lost the key. The w-RGK scheme utilized session information as a hint message. The session information allowed a w-session stateless user to check its path keys in the current session for the latest w-sessions [26].

In another study, Zhang et al. proposed a Hierarchical Group Key Agreement protocol using Orientable Attribute (HGKA-OA) for cloud computing networks. The proposed scheme utilized group key factors which eliminated computation overhead for group key agreements [27].

Vijayakumar et al. proposed approaches to challenges encountered in key management and key distribution in mobile and cloud network applications. The proposed approaches aimed to improve existing systems by providing more efficient and secure communications [28].

In another study, Kung and Hsiao introduced a two-tier GKM called GROUPIT for dynamic Internet of Things (IoT) devices. Each device was assigned to one of many predefined groups in the proposed scheme. Similar devices were grouped together and key management was implemented within each group to support a large number of IoT devices and reduce key update and computation costs [29].

Lee and Park proposed a selectively secure single revocation encryption scheme, a fully secure single revocation encryption scheme and an identity-based revocation scheme [30]. They measured the security and the performance of the schemes by comparing them with each other.

In another study, Yan et al. explored approaches for achieving secure group-oriented communication [31]. They combined two existing schemes into a new cryptosystem, called dual-mode broadcast encryption.

They concluded that in terms of computational costs, a cryptosystem with two modes was more efficient than one with a single mode.

In their study, Hongyong et al. presented a revocable broadcast encryption scheme suitable for use in the cloud environment [32]. Security was provided under standard assumptions using a dual- system encryption technique with a constant-sized cipher text and a private key. Maiti and Misra proposed a scheme to provide privacy preserving in identity-based broadcast proxy re-encryption [33]. The proposed scheme reduced the decryption time compared to broadcast proxy re-encryption schemes and privacy-preserving schemes.

In another study, Jiang and Guo proposed an encrypted

data sharing scheme for secure cloud storage. Their scheme achieves dynamic sharing that enables adding a user to and removing a user from sharing groups dynamically without the need to change encryption public keys. It is a very important functionality requirement for cloud storage [34].

In this study, in terms of ensuring forward and backward secrecy, three central schemes in the literature and the proposed scheme were examined studied after user-adding/removing and batch user-adding/removing operations. The LKH, OFT, and OFC schemes were discussed.

The recommended scheme was compared with existing schemes in terms of number of key transmissions, number of operations, number of user keys, size of user keys, and key transmission sizes in user-adding/removing and batch user-adding/removing operations.

II. Group Communication Attacks

In this section, we present potential attacks which can impact broadcast communication.

Man in the Middle Attack: In this attack type, an attacker aims to locate the connection between the BC and the users. If this succeeds, the attacker removes the connection and establishes two separate connections: middle man-BC and middle man-user [35-36].

Eavesdropping: In this attack type, an attacker can secretly attend a group communication and attempt to listen to messages that have been transmitted. Messages must be encrypted using a group key before transmission in order to block eavesdropping [37].

Denial of Service (DoS) Attack: This is an attack type that aims to stop or hinder group services. A group user can initiate Dos attacks by transmitting a fake leave request in the name of another group user.

In addition, these attacks can be executed with the help of an outside attacker [38-39].

Replay Attack: An attacker can stop a successful authentication message belonging to a legitimate user and resend this message to get access to the group. This attack can be reduced by adding a random sequence number to the message [37].

Impersonation Attack: An attacker can try to contact a group by imitating the identities of users in the group with the aim of launching other attacks by entering the group without permission [39].

Injecting a false message: A false message is injected into a group by an attacker, causing the group to make a false decision. This attack can be reduced by adding a Message Integrity Code (MIC), which ensures the authentication and message integrity of the original message [39].

Compromise Attack: An attacker can obtain a secret key and all transmission messages by compromising a user in the group. The group controller (GC) must be able to identify compromised nodes as soon as possible and remove them from the group [39].

Collusion Attack: This attack can be seen in schemes where the common secret key is obtained when the symmetric keys of the users are computed with a correct functional dependency towards the root node. For this attack, it is necessary for a user to leave the group at time t_1 and a user to join the group at time t_2, respectively. If the joining and leaving users collude with each other secretly, they can figure out the current common secret key at time interval t_2-t_1. One way of reducing this attack is to keep the position information of the leaving user on the scheme and add the new user to this position [40-41].

III. Central Secure Group Communication

A Secure Group Communication (SGC) scheme contains two main components: GKM and Group Membership Management (GMM).

The GKM provides a common secret key among the SGC group members. In the SGC scheme, the common secret key has a significant importance in terms of message security. In group communication, messages are encrypted using the common secret key.

The GKM is responsible for managing both the common secret key and the other secret keys in the BC and for the users. These keys are often used for signature and authentication operations.

The strength of a broadcast scheme depends on the key management protocol used in the group and the length of the common secret key. The key management protocol specifies how to generate, distribute and update a common secret key. The common secret key must be updated after each membership change to provide forward and backward secrecy.

The GMM defines the joining and leaving processes of a user. Firstly, in the process of joining the group, the authentication process must be carried out. Messages in the group should only be accessed by authenticated users.

An SGC scheme is classified into three categories: central, contributory, and hybrid, which will be addressed in a sequel to this study.

The central group key management schemes include a central secure entity called a Group Controller (GC). A GC performs the generation, distribution and updating of the keys in the group. Symmetric encryption methods are generally used in central GKM schemes.

A. Logical Key Hierarchy

The Logical Key Hierarchy (LKH) scheme was suggested by two independent research groups, Wong et al. [42] and Waller et al. [43], at approximately the same time.

As one of the GKM schemes, its use has become popular in recent technologies. The main idea of the LKH is to create a key tree structure that contains a set of encryption keys to which authorized users are joined. The logical key tree is managed by a GC.

There are two types of nodes in the tree: key nodes and user nodes. User nodes are contained in the leaves of the tree.

A user node has individual keys associated with that user. The broadcast center is contained at the root node. A key called the traffic encryption key (TEK) is associated with the broadcast center.

The key nodes on the way from the broadcast center to the users are called intermediate nodes. Intermediate node keys are called key encryption keys (KEK). They are used by the GC to safely transmit the TEK key to all members in the group.

The intermediate node keys on their way from the user node to the root node must be transmitted via a secure path so that each user can calculate the TEK on the scheme. The keys on the path from the user to the root node must be updated when a user joins or leaves the scheme.

Fig. 1 shows a tree structure with eight users. For example, when $User_8$ joins the tree, the following steps are performed in sequence:

The KEK_8 key is created and transmitted to $User_8$.

The $\{KEK'_{78}\}_{KEK8} \rightarrow KEK'_{78}$ key is encrypted with the KEK_8 key and transmitted to $User_8$.

The $\{KEK'_{78}\}_{KEK7} \rightarrow KEK'_{78}$ key is encrypted with the KEK_7 key and transmitted to $User_7$.

The $\{KEK'_{5678}\}_{KEK8} \rightarrow KEK'_{5678}$ key is encrypted with the KEK_8 key and transmitted to $User_8$.

The $\{KEK'_{5678}\}_{KEK5678} \rightarrow KEK'_{5678}$ key is encrypted with the KEK_{5678} key and transmitted to $User_5$, $User_6$ and $User_7$.

The $\{TEK'\}_{KEK8} \rightarrow TEK'$ key is encrypted with the KEK_8 key and transmitted to $User_8$.

The $\{TEK'\}_{TEK} \rightarrow TEK'$ key is encrypted with the TEK key and transmitted to $User_1$, $User_2$, $User_3$, $User_4$, $User_5$, $User_6$ and $User_7$.

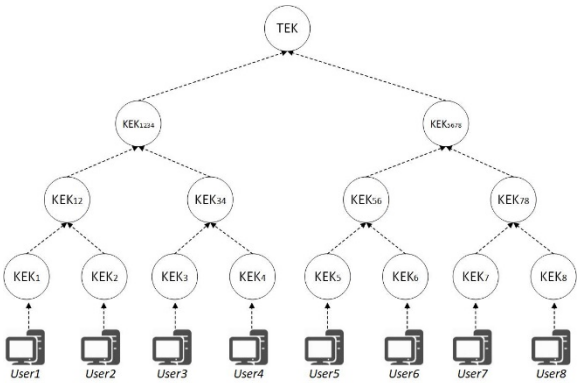

Figure 1. A logical key hierarchy scheme

Updated keys must also be distributed to the users who need them by the GC via a secure path. If there are n users in an LKH scheme, the tree's height is $h=log_2 n$. Each user stores a total of $h+1$ keys, one of which is the user's own individual key.

B. One-Way Function Tree

The One-Way Function Tree (OFT) approach was proposed by Sherman and McGrew [44]. Unlike the LKH, the keys on the tree are computed from top to bottom using a mixing function $f()$ and a one-way function $g()$. Each user node (x) has three cryptographic keys. The n_x is the node secret. Each node key k is obtained by entering the secret key into the $key()$ function: $k_x = key(n_x)$.

n_x' is the blinded node secret, which is calculated using the one-way function $n_x' = g(n_x)$. Due to the property of one-way functions, it is not possible to obtain n_x from the inverse of the function result n_x'.

The $f()$ function is a mixing function (e.g., XOR). Each node secret n_x is obtained by mixing the blind node secrets of the left and right children. Intermediate node calculation is as follows:

nx=f (g(nx_left_child), g(nx_right_child))
=f (nx_left_child', nx_right_child')
n_{00}=f (g(n_{000}), g(n_{001}))
n_{01}=f (g(n_{010}), g(n_{011}))
n_{10}=f (g(n_{100}), g(n_{101}))
n_{11}=f (g(n_{110}), g(n_{111}))
n_0=f (g(n_{00}), g(n_{01}))
n_1=f (g(n_{10}), g(n_{11}))
n_{root}=f (g(n_0), g(n_1))

For calculating node secrets, each user must know the sibling blind node secrets on the path from the user to the root node. For example, in Fig. 2, n_{111}', n_{10}' and n_0' should be transmitted to $User_7$. Otherwise, the user cannot calculate the group key.

C. One-Way Function Chain

The One-Way Function Chain (OFC) was first proposed by Canetti et al. [45] and named by Sherman et al. [44]. Unlike the OFT, the OFC does not have blind node secrets. Each node has two keys; one node is the node secret (r_x) and the other is the node key (k_x). The users are located in the leaves.

There is always a relationship between the node secrets located on the path to the root node of the parent of the user who left the tree. This relationship is called a chain.

The function f, which is used in the OFC and doubles the size of its input, is a one-way pseudo-random generator. This function is used to generate node keys and node secrets. Let us assume that there were eight users in the tree and one user ($User_1$) then left the tree, as shown in Fig. 2.

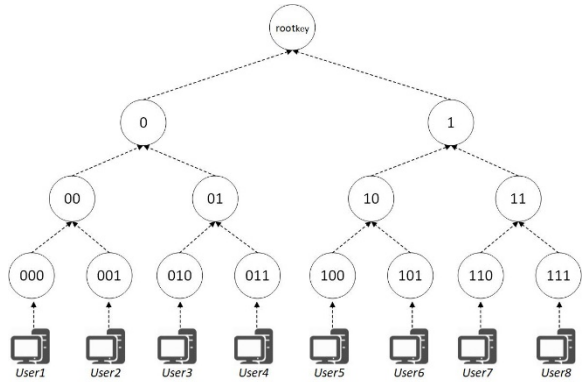

Figure 2. Secret key computation from user nodes to root node

Firstly, the node associated with $User_1$ must be deleted from the tree. Next, the node keys and node secrets located on the path to the root node of the parent of the user who left the tree, must be updated. The updated keys must then be distributed to users who need the current keys.

The GC generates a new random r_{00}, which is encrypted with k_{001} and transmitted to $User_2$, and then r_{00} is entered into an $f()$ function. The left half of this function is the node key of the corresponding node $(k_{00}' = f(r_{00})|_L)$ and the right half is the node secret of the upper node $(r_0 = f(r_{00})|_R)$.

The r_0 is encrypted with k_{01} and transmitted to $User_3$ and $User_4$. The $k_0' = f(r_0)|_L$ is performed and the node key of

this node is calculated. The $r_{root} = f(r_0)|_R$ is performed and the root node secret is calculated. The r_{root} is encrypted with k_1 and transmitted to $User_5$-$User_8$. Finally, the $k_{root}' = f(r_{root})|_L$ is performed and the root node key is created.

Any user can easily calculate the keys that are needed. For example, $User_2$ can get r_{00} by using key k_{001}. The current key of the root node can be obtained by performing the operations $k_{00}' = f(r_{00})|_L$, $k_0' = f(f(r_{00})|_R)|_L$ and $k_{root}' = f(f(f(r_{00})|_R)|_R)|_L$, respectively. As a result, the communication cost is *log n*.

IV. The Proposed Scheme

The proposed scheme is based on a binary tree. The BC is located in the root node and the users are located in the leaves, as with the other schemes in the literature. However, unlike the other schemes, in the proposed scheme both symmetric and asymmetric encryption methods are used. Each user added to the tree has two related keys created with the ECDH; one of the keys is the public key (pu_{user_x}) and the other one is the private key (pr_{user_x}).

Each user must also have a symmetric key to calculate the common secret key. The GC calculates the symmetric key by means of each user's private key. For this purpose, the GC combines the user's private key with random data generated using a random data generator and then put into a *hash()* function. The salting process is then carried out by adding random data to the private key:

$n_{user_x} = hash\ (pr_{user_x} + random\_data)$.

Unlike the method proposed in [46], in the salting process, randomly generated data are used instead of the user's position data.

The GC sends pu_{user_x}, pr_{user_x} and n_{user_x} to the user and loads pu_{user_x} into the predefined public key library (PKL). Similar to the OFT scheme, in order to calculate the common secret keys in the BC, a key calculation must be made from the users to the root node. For the calculation of each intermediate node key n_X, the left half of the left child symmetric key and the right half of the right child symmetric key are combined. This value is then put into a hash() function.

$$n_x = hash(n_{x_{leftHalfOfLeftChild}} + n_{x_{rightHalfOfRightChild}} + random\_data)$$

Similar to the scheme in Fig. 2, the following calculation is used to obtain the calculation of the intermediate node keys and the common secret key from the user nodes to the root node in a tree.

$$n_{00} = hash(n_{000leftHalf} + n_{001rightHalf} + random\_data)$$
$$n_{01} = hash(n_{010leftHalf} + n_{011rightHalf} + random\_data)$$
$$n_{10} = hash(n_{100leftHalf} + n_{101rightHalf} + random\_data)$$
$$n_{11} = hash(n_{110leftHalf} + n_{111rightHalf} + random\_data)$$
$$n_0 = hash(n_{00leftHalf} + n_{01rightHalf} + random\_data)$$
$$n_1 = hash(n_{10leftHalf} + n_{11rightHalf} + random\_data)$$
$$n_{rootkey} = hash(n_{0leftHalf} + n_{1rightHalf} + random\_data)$$

Unlike the OFT scheme, in the proposed scheme only half of the symmetric key is transmitted from the user nodes to the root node. This reduces the total key transmission size in the key updates.

A. System Model

The system consists of a broadcast center (BC), a GC, a dynamic set of group users, a set of users who want to join or leave the group, and the PKL.

The BC is responsible for broadcast message transmission. Broadcast message transmission is carried out using the common secret key, which is the root node key. Before the BC encrypts and sends data, it adds a time stamp to the end of the data. The time stamp is a simple but effective step toward data security. Membership operations are shown in Fig. 3. The GC is responsible for key generation and key distribution.

When a user joins the group, the GC creates a public key (pu_{user_x}), private key (pr_{user_x}) and symmetric key (n_{user_x}) and gives these keys to the user through a unicast channel.

When a user leaves the group, the GC removes that user's node from the tree and along with the user's node keys from the PKL. The leaving user's sibling node relocates to the place of its parent node position. The GC runs a rekey process when a user joins or leaves the group. A rekey message is transmitted in order for the group users to obtain the common secret key.

Broadcast message transmission does not continue during the rekey process. If a user misses the re-key processes or goes offline, the PKL can be accessed to obtain the keys required to calculate the common secret key.

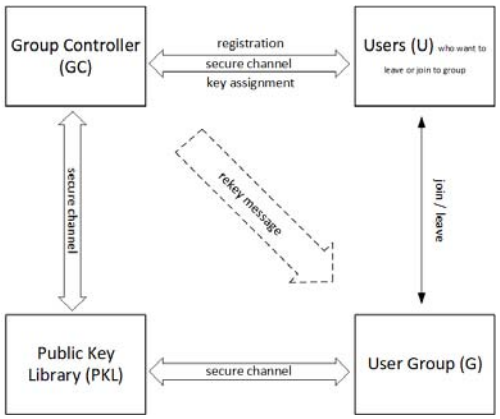

Figure 3. Membership Operations

B. Initialization Process

The GC creates two related keys using the ECDH. One of the keys is the public key ($pu_{root_{key}}$) and the other one is the private key ($pr_{root_{key}}$). It sends the keys to the BC and also loads $pu_{root_{key}}$ into the PKL.

The initial $n_{root_{key}} = hash(pr_{root_{key}} + random\_data)$.

C. User Joining/Leaving

A rekey process is performed when a user either joins or leaves. For each operation, a symmetric key of one of the user nodes of the key tree changes, affecting all of the symmetric keys along the path from this leaf to the root.

The GC securely communicates the changed information along this path to those users who need to know. The GC and all users individually compute the new common secret key.

Fig. 2 shows the tree structure with eight users. Assuming that $User_8$ is granted permission to join the tree, the following steps are performed in sequence:

The GC creates a public key (pu_{user8}) and a private key (pr_{user8}) with the ECDH. It calculates $User_8$'s symmetric key using $User_8$'s private key and randomly generated data:

$n_{user8} = hash(pr_{user8} + random\_data)$.

The GC transmits, pu_{user8}, pr_{user8} and n_{user8} to $User_8$ through a secure channel and also loads pu_{user8} and $n_{111rightHalf}$ to the PKL. The GC encrypts the sibling keys ($n_{110leftHalf}$, $n_{10leftHalf}$, $n_{0leftHalf}$) with pu_{user8} and transmits them to $User_8$ through a secure channel.

$User_8$ decrypts the encrypted sibling keys with pr_{user8} and obtains the common secret key using the following calculation.

$n_{11}' = hash(n_{110leftHalf} + n_{111rightHalf})$

$n_1' = hash(n_{10leftHalf} + n_{11rightHalf})$

$n_{rootkey}' = hash(n_{0leftHalf} + n_{1rightHalf})$

The GC encrypts $n_{111rightHalf}$ with n_{11} and transmits it to $User_7$ through a secure channel. It encrypts $n_{11rightHalf}'$ with n_1 and transmits it to $User_5$ and $User_6$ through a secure channel. It encrypts $n_{1rightHalf}$ with $n_{rootkey}$ and transmits it to $User_1$, $User_2$, $User_3$, and $User_4$ through a secure channel. In this way, all users in the tree update all intermediate node keys and $n_{rootkey}$ on the path from their node to the root.

When $User_8$ leaves the tree, the GC removes the pu_{user8} and $n_{111rightHalf}$ keys from the PKL. The n_{110} node relocates to the n_{11} position and key update operations are performed from the user nodes to the root node in the tree.

The BC stores $pu_{rootkey}$, $pr_{rootkey}$ and $n_{rootkey}$. A group user stores pu_{userx}, pr_{userx}, and half of the sibling's symmetric keys along the path from this leaf to the root and $n_{rootkey}$.

The public keys of all users and the public key of the root node are stored in the PKL. In addition, the left or right halves of each user's symmetric keys, according to the user's position are stored in the PKL.

D. Batch User Joining/Leaving

By using a batch operation that deals with joinings and/or leavings of multiple users rather than repeatedly applying individual joining or leaving operations, the number and size of key transmissions and the number of multiple joining/leaving operations can be substantially reduced. This is because the key update process is only performed after the batch user joining/leaving operations.

E. Security Analysis

In our scheme, as in other GKM schemes, the GC is considered to be a trusted entity for key generation and distribution.

In the user-joining phase, the GC gives the keys to a newly added user through a secure unicast channel that provides data integrity and confidentiality.

The user stores the keys locally. All the communication channels between any two entities are assumed to be authenticated. That is, we assume that the identity of the sender and the integrity of exchanged messages can be verified.

Two keys related to the BC are created using the ECDH: the $pu_{rootkey}$, used to sign a message, and the $pr_{rootkey}$. The $pr_{rootkey}$ is used to sign a message and $pu_{rootkey}$ is used to verify the signature.

When the BC in the root node wants to send data to all users with the common secret key, it first summarizes the data with the *hash()* function. It signs the summary value with $pr_{rootkey}$ and sends the message to all users by adding the summary value to the end of the message.

The users separate the signature in the message and solve the signature with $pu_{rootkey}$. They then compare the two values by summarizing the message. If the values are equal, the message is sent by the BC. Thus, a malicious user cannot act as a BC and perform a message transmission. Therefore, the proposed scheme is not affected by impersonation attacks or injection of false messages. If the BC wants to make a private broadcast to a user, it can obtain the user's public key from the PKL.

The sibling keys in the path from the root node to each user must be given to that user. For the transmission of the broadcast, the common secret key can be used in any encryption method.

Forward and backward secrecy must be ensured after each user change. Thus, the keys on the path from the user to the root node must be updated when a user joins or leaves the scheme.

Adding or removing a user from the scheme is performed in two steps. First, the user must make a request to be added or to leave the scheme. The GC evaluates these requests when broadcast transmission stops.

Broadcast transmission is halted during the t_2-t_1 time and is resumed after the forward and backward secrecy of the proposed scheme has reinforced it against a Collusion Attack.

All broadcast transmissions must be encrypted to block eavesdropping. The proposed scheme uses encryption methods for both key exchange and broadcast transmission, as do the other schemes in the literature. Therefore, the proposed scheme is not affected by eavesdropping.

In the Man in the Middle Attack, an attacker locates the connection between the BC and the user. Even if the attacker listens to the channel, a common secret key is needed to decrypt the encrypted data.

Since the connection between the two entities is encrypted, the attacker must also listen to the *GC-U* channel and obtain the encryption keys. However, the *GC-U* channel is secure, and therefore, the attacker cannot obtain the encryption keys.

A user-leaving request is performed using the position value of the user. However, users do not know their position value in the scheme. Their position value is obtained from the scheme upon their request to leave.

An attacker cannot transmit a fake leave request in the name of another group user. Therefore, the proposed scheme is not affected by the DoS Attack.

The Compromise Attack cannot be easily detected. It is the duty of the GC to block this attack. The GC must be able to identify compromised nodes as soon as possible and remove them from the group. In the proposed scheme, a time stamp is added to the end of the data transmitted to users from the BC. Therefore, the proposed scheme is not affected by the Replay Attack.

V. Applied Work

The proposed scheme utilizes an asymmetric key agreement protocol in the root node with the users located in the leaves. Symmetric keys in the intermediate nodes are

calculated from the user nodes to the root node using a symmetric encryption method.

User addition/removal and batch user addition/removal operations are referred to as membership change. Some keys, especially the common secret key in the root node, must be updated after each membership change in a scheme in order to provide forward and backward secrecy.

Communication, computation and storage costs occur as a result of each re-keying operation.

A scheme should provide data transmission at very low costs without causing security problems. For this purpose, the proposed scheme, as with the LKH, OFT and OFC schemes in the literature, is performed in Java using a similar symmetric encryption method (AES-128). Unlike other schemes, in the proposed scheme the ECC-112 key agreement protocol is also used.

Users are added respectively by adding a user at each step in the scheme 2^n $(0 \leq n \leq 21)$ and then, by removing a user at each step in the scheme 2^n $(0 \leq n \leq 21)$, users are removed respectively.

In addition, batch user addition/removal operations are performed, in which a total of 2^n $(0 \leq n \leq 21)$ users are added to or removed from the scheme at once. The schemes were then compared in the following aspects:

Number of key transmissions: The total number of keys transmitted to the users as a result of the creation of 2^n $(0 \leq n \leq 21)$ users in the tree. This calculation includes the number of key updates performed by adding a user at each step.

Number of operations: The total number of operations obtained as a result of the creation of 2^n $(0 \leq n \leq 21)$ users in the tree. This calculation includes the number of all operations performed by adding a user at each step.

Number and size of user keys: The number and size of keys that must be stored by each user for 2^n $(0 \leq n \leq 21)$ users in the tree.

Key transmission size: The total key size transmitted to users as a result of the creation of 2^n $(0 \leq n \leq 21)$ users in the tree. This calculation includes the total key size as a result of the key updates for each of the 2^n $(0 \leq n \leq 21)$ users.

The results including the number of key transmissions, number of operations and size of key transmissions are shown in Tables I–VI, respectively.

The results including the number and size of user keys are shown in Table VII. MATLAB was used to prepare the data. In the tables, U.C. refers to user count and P.S. to the proposed scheme.

In terms of the number of key transmissions, the proposed scheme placed after the LKH, OFC and OFT schemes in user addition/removal and batch user addition/removal operations, as shown in Fig. 4-7.

The number of key transmissions of the LKH, OFT and OFC schemes were equal because the symmetric keys used in the schemes were calculated from the user nodes to the root node in similar steps.

Fig. 8 shows that, in terms of the number of operations, the proposed scheme was ranked after the LKH and OFC schemes in user adding operations. As a result of user updates, the most operations were performed in the OFT scheme.

Fig.9 shows that, in terms of the number of operations, the proposed scheme gave the best result for batch user-adding operations.

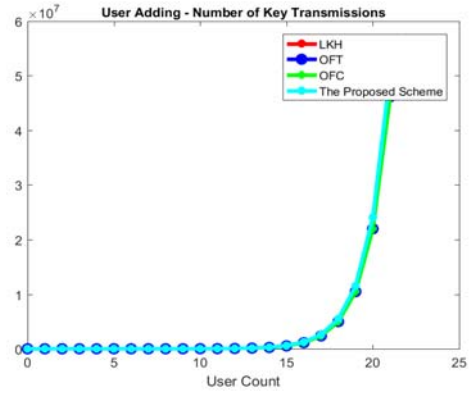

Figure 4. User Adding–Number of Key Transmissions

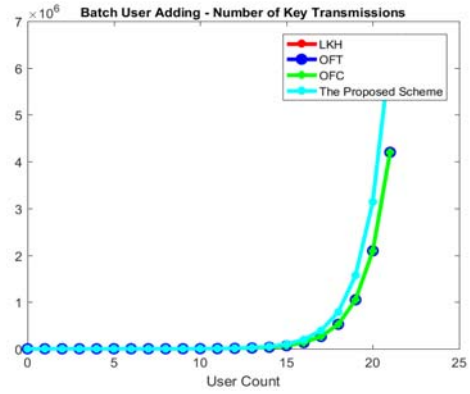

Figure 5. Batch User Adding–Number of Key Transmissions

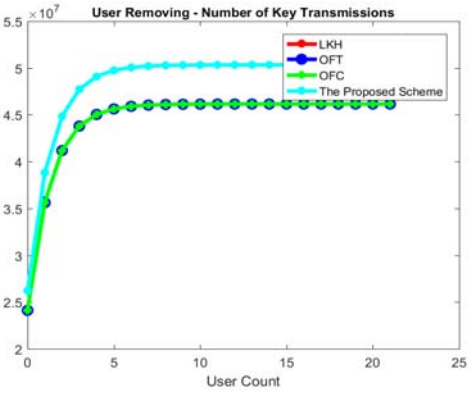

Figure 6. User Removing–Number of Key Transmissions

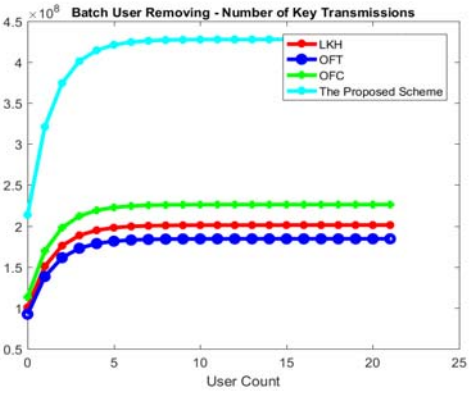

Figure 7. Batch User Removing–Number of Key Transmissions

TABLE I. NUMBER OF KEY TRANSMISSIONS I

	User Adding				Batch User Adding			
U.C	**LKH**	**OFT**	**OFC**	**P.S.**	**LKH**	**OFT**	**OFC**	**P.S.**
2^0	2	2	2	5	2	2	2	5
2^1	5	5	5	12	3	3	3	7
2^2	13	13	13	26	7	7	7	13
2^4	81	81	81	122	31	31	31	49
2^6	449	449	449	590	127	127	127	193
2^8	2305	2305	2305	2834	511	511	511	769
2^{12}	53249	53249	53249	61466	8191	8191	8191	12289
2^{16}	1114113	1114113	1114113	1245218	131071	131071	131071	196609
2^{20}	22020097	22020097	22020097	24117290	2097151	2097151	2097151	3145729
2^{21}	46137345	46137345	46137345	50331692	4194303	4194303	4194303	6291457

TABLE II. NUMBER OF KEY TRANSMISSIONS II

	User Removing				Batch User Removing			
U.C	**LKH**	**OFT**	**OFC**	**P.S.**	**LKH**	**OFT**	**OFC**	**P.S**
2^{21}	24117248	24117248	24117248	26214402	100663272	92274668	113246188	213909690
2^{20}	35651584	35651584	35651584	38797316	150994896	138411992	169869272	320864628
2^{16}	45613056	45613056	45613056	49741836	198180720	181665672	222953352	421135452
2^{12}	46112768	46112768	46112768	50302996	201129744	184368952	226271032	427403076
2^8	46136320	46136320	46136320	50330396	201313968	184537832	226478312	427795500
2^6	46137152	46137152	46137152	50331424	201323136	184546240	226488640	427815456
2^4	46137312	46137312	46137312	50331636	201325392	184548312	226491192	427820724
2^2	46137340	46137340	46137340	50331680	201325920	184548800	226491800	427822320
2^1	46137343	46137343	46137343	50331687	201325992	184548868	226491888	427822710
2^0	46137345	46137345	46137345	50331692	201326040	184548912	226491942	427823008

TABLE III. NUMBER OF OPERATIONS I

	User Adding				Batch User Adding			
U.C	**LKH**	**OFT**	**OFC**	**P.S.**	**LKH**	**OFT**	**OFC**	**P.S.**
2^0	2	6	3	4	2	6	3	4
2^1	4	12	6	8	16	16	17	11
2^2	12	24	16	20	36	36	37	25
2^4	100	156	116	132	156	156	157	109
2^6	644	972	708	772	636	636	637	445
2^8	3588	5388	3844	4100	2556	2556	2557	1789
2^{12}	90116	135180	94212	98308	40956	40956	40957	28669
2^{16}	1966084	2949132	2031620	2097156	655356	655356	655357	458749
2^{20}	39845892	59768844	40894468	41943044	10485756	10485756	10485717	7340029
2^{21}	83886084	125829132	85983236	88080388	20971516	20971516	20971437	14680061

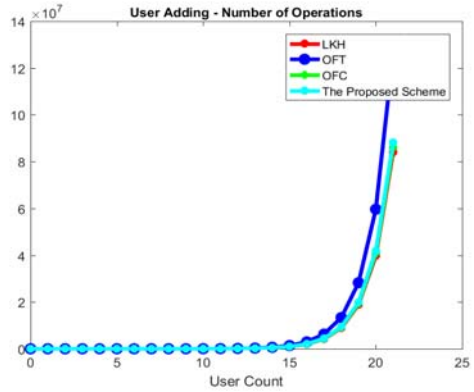

Figure 8. User Adding–Number of Operations

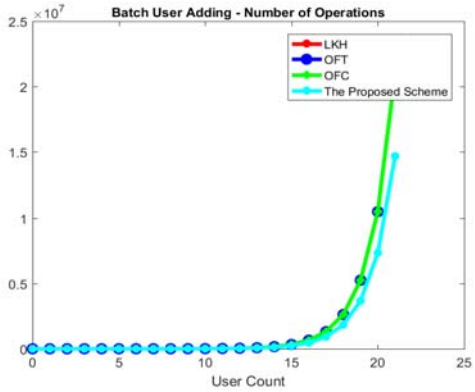

Figure 9. Batch User Adding–Number of Operations

The proposed scheme ranked second-best after the LKH scheme in terms of the number of operations for user removal and batch user removal, followed by the OFT and OFC schemes, as shown in Fig. 10 and Fig. 11, respectively.

In terms of the number of keys stored by the users, the proposed scheme ranked after the LKH, OFT and OFC schemes as shown in Fig. 12. However, in terms of the size of the keys, the total size of the keys decreased as the number of users in the scheme increased. The proposed scheme did not produce the best result until reaching 2^{20} users.

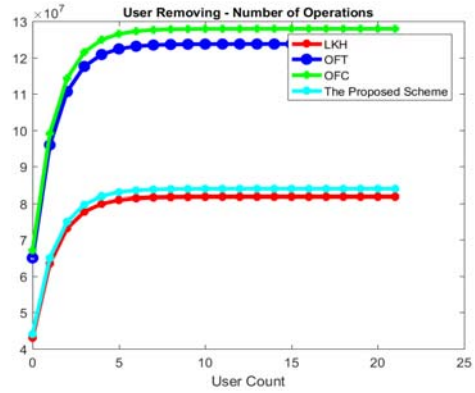

Figure 10. User Removing–Number of Operations

TABLE IV. NUMBER OF OPERATIONS II

	User Removing				Batch User Removing			
U.C	**LKH**	**OFT**	**OFC**	**P.S.**	**LKH**	**OFT**	**OFC**	**P.S.**
2^{21}	42991616	65011712	67109464	44040192	8388604	10485756	12582908	8388605
2^{20}	63438848	95944704	99091032	65011712	12582904	15728632	18874360	12582906
2^{16}	80904192	122388480	126517848	82968576	16515048	20643816	24772584	16515054
2^{12}	81750016	123672576	127863384	83845120	16760792	20951000	25141208	16760802
2^{8}	81787520	123729792	127924440	83884544	16776136	20970184	25164232	16776150
2^{6}	81788704	123731616	127926456	83885824	16776896	20971136	25165376	16776912
2^{4}	81788904	123731928	127926816	83886048	16777080	20971368	25165656	16777098
2^{2}	81788930	123731970	127926870	83886080	16777120	20971420	25165720	16777140
2^{1}	81788930	123731975	127926877	83886084	16777124	20971426	25165728	16777145
2^{0}	81788930	123731980	127926884	83886088	16777124	20971430	25165733	16777148

TABLE V. SIZE OF KEY TRANSMISSIONS (BAYT) I

	User Adding				Batch User Adding			
U.C	**LKH**	**OFT**	**OFC**	**P.S.**	**LKH**	**OFT**	**OFC**	**P.S.**
2^{0}	48	44	54	298	48	44	54	298
2^{1}	120	108	128	606	72	68	88	390
2^{2}	288	256	296	1036	168	156	196	594
2^{4}	1680	1464	1624	2992	744	684	844	1818
2^{6}	9408	8096	8736	10540	3048	2796	3436	6714
2^{8}	49392	42184	44744	43336	12264	11244	13804	26298
2^{12}	1179984	999704	1040664	830080	196584	180204	221164	417978
2^{16}	25166256	21234024	21889384	15863224	3145704	2883564	3538924	6684858
2^{20}	503317008	423625144	434110904	295702768	50331624	46137324	56623084	106954938
2^{21}	1056965160	889192908	910164428	612372926	100663272	92274668	113246188	213909690

TABLE VI. SIZE OF KEY TRANSMISSIONS (BAYT) II

	User Removing				Batch User Removing			
U.C	**LKH**	**OFT**	**OFC**	**P.S.**	**LKH**	**OFT**	**OFC**	**P.S.**
2^{21}	578813976	490733588	511705108	413139150	100663272	92274668	113246188	213909690
2^{20}	855638064	725614632	757071912	614465948	50331624	46137324	56623084	106954938
2^{16}	1094713488	928776312	970063992	794690772	3145704	2883564	3538924	6684858
2^{12}	1106706672	939016392	980918472	804726796	196584	180204	221164	417978
2^{8}	1107272016	939502872	981443352	805278020	12264	11244	13804	26298
2^{6}	1107292032	939520320	981462720	805302496	3048	2796	3436	6714
2^{4}	1107295920	939523752	981466632	805308444	744	684	844	1818
2^{2}	1107296640	939524400	981467400	805310120	168	156	196	594
2^{1}	1107296736	939524488	981467508	805310520	72	68	88	390
2^{0}	1107296784	939524532	981467562	805310818	48	44	54	298

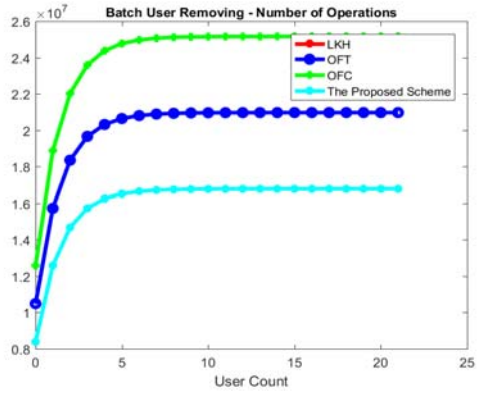

Figure 11. Batch User Removing–Number of Operations

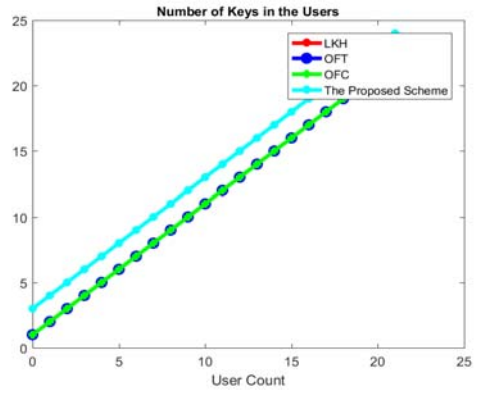

Figure 12. Number of Keys in the Users

Fig. 13 shows that the proposed scheme yielded the best results with 2^{20} or more users.

In terms of the key transmission size, the proposed scheme gave the best results for user addition and removal operations as shown in Fig. 14 and Fig. 15.

This is because the keys were obtained using less calculation for key update operations than the other schemes used. In the calculation operation, each parent node key is obtained from the user nodes to the root node by combining the left half of the left child key and the right half of the right child key and inserting this into a *hash()* function.

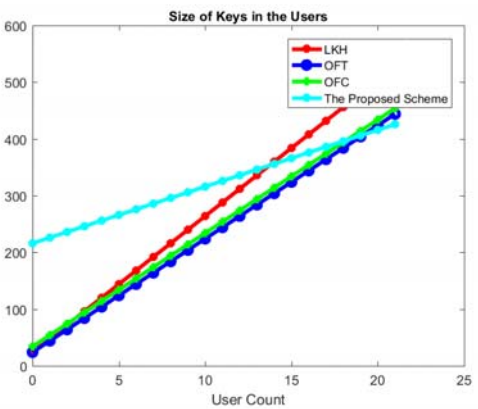

Figure 13. Size of Keys in the Users

TABLE VII. NUMBER AND SIZE OF KEYS IN THE USERS

	Number of Keys in the User				Size of Keys in the User (Bayt)			
U.C	**LKH**	**OFT**	**OFC**	**P.S.**	**LKH**	**OFT**	**OFC**	**P.S.**
2^0	1	1	1	3	24	24	34	216
2^1	2	2	2	4	48	44	54	226
2^2	3	3	3	5	72	64	74	236
2^4	5	5	5	7	120	104	114	256
2^6	7	7	7	9	168	144	154	276
2^8	9	9	9	11	216	184	194	296
2^{12}	13	13	13	15	312	264	274	336
2^{16}	17	17	17	19	408	344	354	376
2^{20}	21	21	21	23	504	424	434	416
2^{21}	22	22	22	24	528	444	454	426

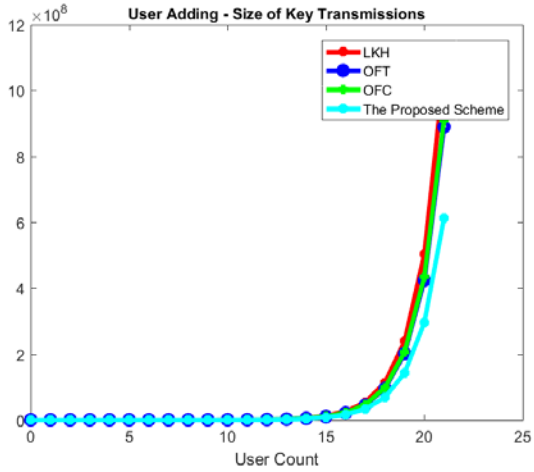

Figure 14. User Adding–Size of Key Transmissions

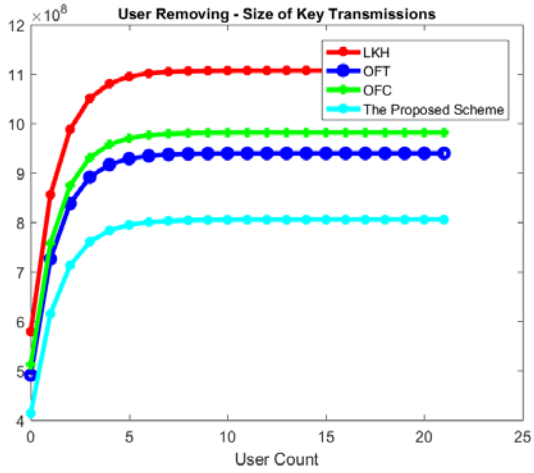

Figure 15. User Removing–Size of Key Transmissions

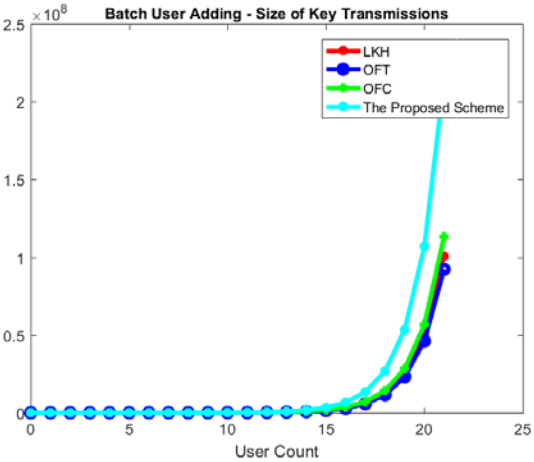

Figure 16. Batch User Adding–Size of Key Transmissions

In terms of key transmission size, the proposed scheme ranked after the LKH, OFT and OFC schemes in batch user addition and removal operations as shown in Fig. 16 and Fig.17.

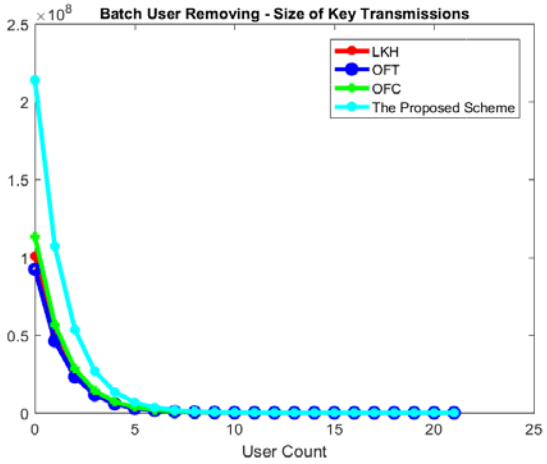

Figure 17. Batch User Removing–Size of Key Transmissions

VI. CONCLUSION

In the study, the LKH, OFT and OFC schemes were discussed after which a new scheme was proposed. The schemes were written in Java and the results were obtained by using a computer with i7-7700HQ CPU, a 2.80GHz processor and 16 GB RAM. The results were graphically expressed using MATLAB.

User addition/removal and batch user addition/removal operations were compared in terms of five criteria including number of key transmissions, number of operations, number of user keys, size of user keys and size of key transmissions.

The proposed scheme yielded the best results in terms of the transmission size in the user-addition/removal operations. It exhibited an advantage in terms of transmission costs for user-addition/removal operations. It also yielded the best results in terms of the number of operations in batch user-addition operations and in user key size when it had 2^{20} or more users.

The proposed scheme also exhibited an advantage in terms of user storage costs when it had 2^{20} or more users. Furthermore, unlike other central GKM schemes, the proposed scheme utilizes public key cryptosystems both in the user nodes and in the root node. In this way, these keys are used both in the symmetric key calculations of the user nodes and in the transmission of signed messages from the BC.

ACKNOWLEDGMENT

This work was supported by the Tubitak 2211-C Domestic Priority Areas Doctoral Scholarship Program.

REFERENCES

[1] B. Daghighi, M. L. M. Kiah, S. Shamshirband, M. H. U. Rehman, "Toward secure group communication in wireless mobile environments: Issues, solutions, and challenges," Journal of Network and Computer Applications, vol. 50, pp. 1-14, 2015. doi:10.1016/j.jnca.2014.11.001

[2] P. Sakarindr, N. Ansari, "Security services in group communications over wireless infrastructure, mobile ad hoc, and wireless sensor networks," IEEE Wireless Communications, vol. 14, no. 5, pp. 8-20, 2007. doi:10.1109/MWC.2007.4396938

[3] X. He, M. Niedermeier, H. De Meer, "Dynamic key management in wireless sensor networks: A survey," Journal of Network and Computer Applications, vol. 36, no. 2, pp. 611-622, 2013. doi:10.1016/j.jnca.2012.12.010

[4] P. K. Shanu, K. Chandrasekaran, "Distribution function based efficient secure group communication using key tree," In 2016 International Conference on Recent Trends in Information Technology (ICRTIT), pp. 1-6, IEEE 2016. doi:10.1109/ICRTIT.2016.7569579

[5] P. M. J. Prathap, V. Vasudevan, "Analysis of the various key management algorithms and new proposal in the secure multicast communications," arXiv preprint arXiv:0906.3956, 2009.

[6] T. Sakamoto, T. Tsuji, Y. Kaji, "Group key rekeying using the LKH technique and the Huffman algorithm," In 2008 International Symposium on Information Theory and Its Applications, pp. 1-6, IEEE, 2008. doi:10.1109/ISITA.2008.4895405

[7] Q. Gu, L. Peng, L. Wang-Chien, C. Chao-Hsien, "KTR: An efficient key management scheme for secure data access control in wireless broadcast services," IEEE Transactions on Dependable and Secure Computing, vol. 6, no. 3, pp. 188-201, 2008. doi:10.1109/TDSC.2008.12

[8] W. Song, H. Zou, H. Liu, J. Chen, "A practical group key management algorithm for cloud data sharing with dynamic group," China Communications, vol. 13, no. 6, pp. 205-216, 2016. doi:10.1109/CC.2016.7513215

[9] N. Alyani, K. Seman, N. M. Nawawi, M. N. S. M. Sayuti, "The improvement of key management based on logical key hierarchy by implementing Diffie Hellman algorithm," Journal of Emerging Trends in Computing and Information Sciences, vol. 3, no. 3, 2012.

[10] H. Liu, J. Li, X. Hao, G. Zou, "A novel LKH key tree structure based on heuristic search algorithm," In 2014 IEEE International Conference on Communiction Problem-solving, pp. 35-38, IEEE, 2014. doi:10.1109/ICCPS.2014.7062210

[11] N. Sakamoto, "An efficient structure for LKH key tree on secure multicast communications," In 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), pp. 1-7, IEEE, 2014. doi:10.1109/SNPD.2014.6888676

[12] H. Bodur, R. Kara, "Implementing Diffie-Hellman key exchange method on logical key hierarchy for secure broadcast transmission," In 2017 9th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 144-147, IEEE, 2017. doi:10.1109/CICN.2017.8319374

[13] Y. Sun, M. Chen, A. Bacchus, X. Lin, "Towards collusion-attack-resilient group key management using one-way function tree," Computer Networks, vol. 104, pp. 16-26, 2016. doi:10.1016/j.comnet.2016.04.014

[14] X. Xu, L. Wang, A. Youssef, B. Zhu, "Preventing collusion attacks on the one-way function tree (OFT) scheme," In International Conference on Applied Cryptography and Network Security, pp. 177-193, Springer, Berlin, Heidelberg, 2007. doi:10.1007/978-3-540-72738-5_12

[15] M. S. Hwang, P. C. Sung, "A study of micropayment based on one-way hash chain," IJ Network Security, vol. 2, no. 2, pp. 81-90, 2006.

[16] J. Lee, J. W. Seo, H. Ko, H. Kim, "TARD: Temporary Access Rights Delegation for guest network devices," Journal of Computer and System Sciences, vol. 86, pp. 59-69, 2017. doi:10.1016/j.jcss.2016.07.002

[17] M. Benmalek, Y. Challal, "MK-AMI: efficient Multi-group Key management scheme for secure communications in AMI systems," In 2016 IEEE Wireless Communications and Networking Conference, pp. 1-6, IEEE, 2016. doi:10.1109/WCNC.2016.7565124

[18] Y. R. Chen, W. G. Tzeng, "Group key management with efficient rekey mechanism: a semi-stateful approach for out-of-synchronized members," Computer Communications, vol. 98, pp. 31-42, 2017. doi:10.1016/j.comcom.2016.08.001

[19] M. Benmalek, Y. Challal, A. Derhab, A.Bouabdallah, "VerSAMI: Versatile and Scalable key management for Smart Grid AMI systems," Computer Networks, vol. 132, pp. 161-179, 2018. doi:10.1016/j.comnet.2018.01.010

[20] V. Kumar, R. Kumar, S. K. Pandey, "A computationally efficient centralized group key distribution protocol for secure multicast communications based upon RSA public key cryptosystem," Journal of King Saud University-Computer and Information Sciences, 2018. doi:10.1016/j.jksuci.2017.12.014

[21] Y. Hanatani, N. Ogura, Y. Ohba, L. Chen, S. Das, "Secure multicast group management and key distribution in IEEE 802.21," In International Conference on Research in Security Standardisation, pp. 227-243, Springer, Cham, 2016. doi:10.1007/978-3-319-49100-4_10

[22] M. Elhoseny, H. Elminir, A. Riad, X. Yuan, "A secure data routing schema for WSN using elliptic curve cryptography and homomorphic encryption," Journal of King Saud University-Computer and Information Sciences, vol. 28, no. 3, pp. 262-275, 2016. doi:10.1016/j.jksuci.2015.11.001

[23] H. Lin, M. Hsieh, K. Li, "The Cluster-based key management mechanism with secure data transmissions scheme in wireless sensor networks," DEStech Transactions on Engineering and Technology Research, amma, 2017. doi:10.12783/dtetr/amma2017/13378

[24] S. K. H. Islam, G. P. Biswas, "A pairing-free identity-based two-party authenticated key agreement protocol for secure and efficient communication," Journal of King Saud University-Computer and Information Sciences, vol. 29, no. 1, pp. 63-73, 2017. doi:10.1016/j.jksuci.2015.01.004

[25] A. Chaudhari, G. Pareek, B. R. Purushothama, "Security analysis of centralized group key management schemes for wireless sensor networks under strong active outsider adversary model," In 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1576-1581, IEEE, 2017. doi:10.1109/ICACCI.2017.8126066

[26] J. Hur, Y. Lee, "A reliable group key management scheme for broadcast encryption," Journal of Communications and Networks, vol. 18(2), pp. 246-260, 2016. doi:10.1109/JCN.2016.000034

[27] Q. Zhang, et al., "A hierarchical group key agreement protocol using orientable attributes for cloud computing," Information Sciences, vol. 480, pp. 55-69, 2019. doi:10.1016/j.ins.2018.12.023

[28] P. Vijayakumar, V. Chang, L. J. Deborah, B. S. R. Kshatriya, "Key management and key distribution for secure group communication in mobile and cloud network," 2018. doi:10.1016/j.future.2018.03.027

[29] Y. H. Kung, H. C. Hsiao, "GroupIt: Lightweight group key management for dynamic IoT environments," IEEE Internet of Things Journal, vol. 5, no. 6, pp. 5155-5165, 2018. doi:10.1109/JIOT.2018.2840321

[30] L. Kwangsu, P. Jong Hwan, "Identity-based revocation from subset difference methods under simple assumptions," vol. 7, pp. 60333–60347, IEEE Access, 2019. doi:10.1109/ACCESS.2019.2915373

[31] Z. Yan, Y. Ruyun, E. Chen, H. Dijiang, "An efficient broadcast encryption supporting designation and revocation mechanisms," Chinese Journal of Electronics, vol. 28(3), pp. 445–456, 2019. doi:10.1049/cje.2019.02.005

[32] J. Hongyong, C. Yue, Y. Kuiwu, Y. Guo, Z. Wang, "Revocable broadcast encryption with constant ciphertext and private key size," Chinese Journal of Electronics, vol. 28(4), pp. 690–697, 2019. doi:10.1049/cje.2019.04.003

[33] M. Sumana, M. Sudip, "P2B: Privacy preserving identity-based broadcast proxy re-encryption," IEEE Transactions on Vehicular Technology, vol. 69(5), pp. 5610–5617, 2020. doi:10.1109/TVT.2020.2982422

[34] L. Jiang, D. Guo, "Dynamic encrypted data sharing scheme based on conditional proxy broadcast re-encryption for cloud storage," IEEE Access, vol. 5, pp. 13336–13345, 2017. doi:10.1109/ACCESS.2017.2726584

[35] Y. Desmedt, "Man-in-the-middle attack," Encyclopedia of cryptography and security, pp. 759-759, 2011. doi:10.1007/978-1-4419-5906-5_324

[36] S. Iqbal, et al., "On cloud security attacks: A taxonomy and intrusion detection and prevention as a service," Journal of Network and Computer Applications, vol. 74, pp. 98-120, Elsevier, 2016. doi:10.1016/j.jnca.2016.08.016

[37] O. Cheikhrouhou, "Secure group communication in wireless sensor networks: a survey," Journal of Network and Computer Applications, vol. 61, pp. 115-132, Elsevier, 2016. doi:10.1016/j.jnca.2015.10.011

[38] F. Sun, Z. Zhao, Z. Fang, L. Du, Z. Xu, D. Chen, "A review of attacks and security protocols for wireless sensor networks," Journal of

Networks, vol. 9, no. 5, pp. 1103, Academy Publisher, 2014. doi:10.4304/jnw.9.5.1103-1113

[39] K. Venkatraman, J. Daniel, G. Murugaboopathi, "Various attacks in wireless sensor network: Survey," International Journal of Soft Computing and Engineering (IJSCE), vol. 3, no. 1, pp. 208-212, Citeseer, 2013.

[40] X. Liu, Y. Zhang, B. Wang, J. Yan, "Mona: Secure multi-owner data sharing for dynamic groups in the cloud," IEEE transactions on parallel and distributed systems, vol. 24, no. 6, pp. 1182-1191, IEEE, 2012. doi:10.1109/TPDS.2012.331

[41] Z. Zhu, Z. Jiang, R. Jiang, "The attack on Mona: Secure multi-owner data sharing for dynamic groups in the cloud," 2013 International Conference on Information Science and Cloud Computing Companion, pp. 213-218, IEEE, 2013. doi:10.1109/ISCC-C.2013.135

[42] C. K. Wong, M. Gouda, S. S. Lam, "Secure group communications using key graphs," IEEE/ACM transactions on networking, vol. 8, no. 1, pp. 16-30, 2000. doi:10.1109/90.836475

[43] D. Wallner, E. Harder, R. Agee, "Key management for multicast: Issues and architectures," RFC 2627, 1999 from: http://www.hjp.at/(de,st_b)/doc/rfc/rfc2627.html

[44] A. T. Sherman, D. A. McGrew, "Key establishment in large dynamic groups using one-way function trees," IEEE transactions on Software Engineering, vol. 29, no. 5, pp. 444-458, 2003. doi:10.1109/TSE.2003.1199073

[45] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, B. Pinkas, "Multicast security: A taxonomy and some efficient constructions," In IEEE INFOCOM' 99. Conference on Computer Communications, vol. 2, pp. 708-716, IEEE, 1999. doi:10.1109/INFCOM.1999.751457

[46] H. Bodur, R. Kara, "Yayın Şifreleme Şemaları Üzerinde Bir Karşılaştırma: Bir Yeni Yayın Şifreleme Şeması," Düzce Üniversitesi Bilim ve Teknoloji Dergisi, vol. 7(1), pp. 861–871, 2019. doi:10.29130/dubited.509102