User Multi Group Key Distribution Using Secret Sharing with Circulate Matrices Based on Structures Pythagors Equation and Ecdh Key Exchange Protocol

Eman Talib [1,2,] and Valeriy Osipyan [1]
[1]Kuban State University, Krasnodar 350040, Russia
[2]Basra University, Computer Science and Information Technology collage, Basra, Iraq
E-mail: emanalghareeb38@gmail.com, v.osippyan@gmail.com

Keywords: ECDH exchange protocol, Pythagoras Equation, Diophantine equations in second degree, circulate matrices, secret sharing.

Received: February 21, 2023

The majority of currently used conventional group key distribution protocols are primarily created for a single group. But group communications are becoming more and more popular as networks improve quickly. So all participating users must share or exchange a secure group key beforehand in order to protect communication and multi-group key installations are necessary for many group-oriented applications at the moment. This allows users to join numerous groups at once. A novel type of user-oriented multi-group key setups employing secret sharing was recently provided by C.F. Hsu et al. in 2018 (UMKESS). This protocol, like many other group key establishment systems, is polynomial-based, requiring both the key generation center (KGC) and each group member to resolve t-degree *approximating polynomials in order to distribute and retrieve the secret group key. N. Shruti et al in 2018 suggested a user-friendly group key distribution mechanism uses secret sharing with circulate matrices.in this article we have improved performance security of previous protocol by using two techniques, ECDH exchange protocol to generate sharing secret key with using it as key in term of Diophantine equations in second degree. Security analysis is displayed that our suggested technique more effective, secure, robust and achieves the key security, provides forward and back-ward secrecy, prevents insider and out sider attacks.*

Povzetek: Predstavljena je nova metoda distribucije ključev v skupini z uporabo matrik, pitagorovih enačb in Ecdh protokola.

1 Introduction

Today's communication methods go further than one-to-one or one-to-many (i.e., server/client) interactions and it became between M and M that occur increasingly frequently. So when offering secure communication, the two security features Message confidentiality and Message authentication are often taken into consideration. According to [1], when a secure communication includes more entities, all groups members will need group key known group key management which can classified into two categories: centralized group key management protocols and distributed group key management protocols. Numerous academic publications have examined group key establishment processes (distributed group key management protocols). These protocols are divided into two groups, group key agreement (or exchange) protocols and group key distribution (or transfer) protocols, depending on whether a trustworthy key generation centre (KGC) is present or not [2].

Key distribution protocols rely on a mutually trusted key generation centre (KGC) to select session keys and then transport session keys to all communication entities secretly. Most often, KGC encrypts session keys under another secret key shared with each entity during registration. While key agreement protocols; all communication entities are involved to determine session keys. The most commonly used key agreement protocol is Diffie-Hellman (DH) key agreement protocol [3].

DH key agreement protocol was overly broad, for instance, Ingemarsson et al. [4], Steer et al. [5], Burmester and Desmedt [6], and Steiner et al. [7] used this strategy. In 1996, Steiner et al. proposed a logical expansion of DH and gave the project the term DH key exchange. Authentication services were added later in 2001, and it has since proven to be secure [6]. In 2006, Bohli [8] established a framework for strong group key agreement that offers protection in an unauthenticated point-to-point network against active adversaries and malevolent insiders. Then, in 2007, Bresson et al. [9] developed a general authenticated group DH Key exchange, and the algorithm is proven to be secure. Also, in 2007, Katz and Yung [10] created the first group DH protocol that is provably safe in the standard model (that is, without relying on "random oracles") and is constant-round and fully scalable. Establishing a hidden group key among all group members without relying on a KGC that is both mutually trusted is the basic characteristic of the group DH key exchange.

In contrast to group key agreement techniques, group key distribution protocols have a KGC that is authorized by all users. So that group key generation can be done more quickly and most widely used between two sorts of group key distribution protocols [2].

Distributed group key management protocol is based on non-DH key agreement approach. Tzeng [11] In recent years, a discrete logarithm (DL) assumption-based conference key agreement system with fault tolerance was suggested. Even if there are numerous malevolent individuals among the conference participants, the protocol can still construct a conference key. The protocol, however, imposes a significant efficiency barrier by requiring each participant to produce nn-power polynomials, where n is the number of participants. In 2008, Cheng and Laih [12] updated Tseng's bilinear pairing-based conference key agreement protocol. In 2009, Huang et al. [13] To increase the effectiveness of Tseng's protocol, a no interactive protocol based on the DL assumption was presented. One major issue with key agreement protocols is that because all communication entities must participate in deciding session keys, setting up the group key may take an excessively lengthy period, especially when there are a lot of group members. However, due to the following issues, distributing a group key is challenging [14]:

1.Since a group key is sent to numerous users, it is simpler to intercept it as it is being distributed.

2.Group members may occasionally switch. The group key must be updated whenever a user leaves or enters a group so that users outside the group are unaware of the new group key.

3.A session key must be updated after a while, even if a group does not change for a long time. Otherwise, enemies might be able to break it.

The aforementioned issues may be solved through secret sharing, a group-oriented cryptographic technology. Consequently, using Shamir's (t, n) threshold secret sharing (SS) scheme, a significant amount of group key distributed protocol research has been published in the scientific literature over the last few years. The most common group key distributed methods are based on the secret sharing scheme (SSS), which was separately developed in 1979 by Blakley and Shamir [3]. Then, in 1989, Laih et al. [15] Presented the first group key transmission protocol employing the secret sharing scheme (SSS). Later, numerous alternative group key transmission protocols [15] [16][17] were suggested, all of which used. 2010 saw the proposal of a first authenticated GKT focus mainly on SSS by Harn et al [17]. The innovative GKT protocol's confidentiality and authentication are information theoretically safe. However, under this protocol, KGC and each group member must create a t-degree interpolating polynomial in order to distribute and retrieve the secret group key. The authenticated SSS protocol by Harn et al. [17] with the construction of a t-degree interpolating polynomial has also been the subject of numerous studies [18][19][20]. In order to get around this problem, Hsu et al. [14] provided a reliable GKT protocol. They used a linear secret sharing strategy on the vandermonde matrix to distribute the group keys effectively, which lowers the computational burden on each group member. In their technique, the information relating to the group keys was hidden by the vandermonde matrix. An authenticated and secure GKT protocol based on a secret sharing system with circulant matrices was recently presented by S. Nathani et al. in 2018[21]. However, all of the aforementioned conventional GKT protocols can only create one group key at a time, or create one group key per group. The demand for multi-group communications, where users can join numerous groups at once, is increasing because of the rapid growth of group-oriented services like as commercial conference systems, body area wireless networks, programmable routey communications, and file sharing tools, etc. C.F. Hsu et al. [22] Recently presented a novel kind of user-oriented multi-group key setups utilizing secret sharing (UMKESS) in 2018. Additionally based on polynomials is this multi group key setup approach. Again, this meaning that in order to distribute and recover the secret group key, KGC and each member of the group must solve a polynomial of t degrees.

As a result, it is expanded the standard GKT protocol which inspired by Nathani et al. in 2018[21]into a multi-group key transfer protocol on SSS with circulant matrices, which is inspired by C.F. Hsu et al.'s [22] UMKESS protocol. In this research, we offer a novel SSS user multi group key distribution technique using circulant matrices with employment of two technique, first is ECDH key exchange in generating initial values, while the second is using terms' Pythagoras equations in generating and distribution group key in different ways in each time .where we improved last schema in terms of achievement confidentiality and authentication as well as high achievement compared to previous work.

This paper is organized as follows: In section 2 we give a brief of the underlying mathematic and preliminaries. In section 3 we describe our cryptosystem including and give a method a Diophantine equation of second degree with a given solution. In section 4 security analysis 5 performance evaluation 6 conclusions.

2 The underlying mathematic and preliminaries

2.1 Diophantine equation

In this section, we will explain the equations that we adopt in this paper. This is Diophantine equation. It is one of the earliest topics in number theory which had been first studied by the Greek mathematical Diophantus of Alexandria during the 3th century. By definition, a Diophantine equation is a polynomial equation of the form [23] [24]

$$f(x_1,x_2,\ldots,x_n)=0 \qquad (1)$$

Definition1. A Diophantine equation is a polynomial equation where the coefficients are integers, and the solutions are integers or n prove the impossibility of that. The most basic Diophantine equation is of the following form[25]:Historically, this equation is:

$$x^2 + y^2 = z^2 \qquad (2)$$

One of the first Diophantine equations which it is derived from the problem of existing all the rectangular triangles whose sides have integer lengths. Such triples (x,y,z) are called Pythagorean triples.

Definition2. For any right-angled triangle, the square of the hypotenuse c equals the sum of squares of on the two (shorter) legs lengths a and b, which is written as $x^2 + y^2 = z^2$.

Where Pythagorean triple is based on a set of Diophantine equations which has a general two-parameter solution (a and b - parameters). There is more than one formula for solving the Diophantine equation of the second degree but we will adopt the formula Euclid's formula says that, (a,b,c) are a Pythagorean triple, $a^2 + b^2 = c^2$ where a,b,c are integers, if and only if

$$a=m^2-n^2,\ b=2ab,\ c=m^2+n^2 \qquad (3)$$

for some integers m,n.

In particular, this paper is focus on the second-degree Pythagoras equation and the following class of its solutions from N (in more general case we can consider its solutions from Z or Q) (3).

2.2 Elliptic-Curve Diffie-Hellman key exchange

The Elliptic Curve Diffie Hellman (ECDH) is protocol to exchange key by using Diffie Hellman (DH) way depending on the elliptic curve discrete logarithm problem (ECDLP) instead of the discrete logarithm problem (DLP). ECDH is an undisclosed key agreement protocol which accepts two side, A and B, to create a shared secret key over an insecure channel without sending it directly to each other, where each of the side have an elliptic curve public-private key pair[26][27].

2.3 Secret sharing

In a secret sharing scheme, a secret S is divided into n shares and distributed among a groups of n shareholders by a mutually trusted dealer in such a way that only a subset of shareholders who have been given permission to do so can reconstruct the secret; shareholders who have not been given permission cannot learn the secret. A strategy is considered perfect if no unauthorized subgroup of shareholders can learn what the secret is.[28][2].

2.4 SSS based on Circulant matrix for multi-group communications:[21]

- A circular matrix is a square matrix in which the subsequent rows are created by repeatedly right shifting the current row by one element, starting with the first row.

$$C=\begin{pmatrix} c_0 & c_2 & c_1 \\ c_1 & c_0 & c_2 \\ c_2 & c_1 & c_0 \end{pmatrix}$$

- Distributing n members as $\{U_1,U_2,U_3 \ldots U_n\}$in multi m secure groups $\{G_1,G_2,G_3 \ldots G_m\}$with their long term secrets $\{K_{seckey1}, K_{seckey2}, K_{seckey} \ldots K_{seckeyn}\}$ to secure communication.

2.5 Secret reconstruction protocol

To compute s_{ji} there is need for two values, the first consider of Circulant matrix of long term secrets $K_{seckeyj}^{i}$ as privet key where $(1\le j \le n, 1\le i \le m)$,so Circulant matrix of long term secrets $K_{seckeyj}^{i}$ for U_j is ($K_{seckey1}^{1}, K_{seckey1}^{2}, K_{seckey1}^{3} \ldots K_{seckeyn}^{m}$) where n *is* sequence participant in particular group and m is number participants in particular group G_i second value is represented by Circulant matrix $[r_{ji}]$ as public key for$(1\le j \le n, 1\le i \le m)$ as below:

$$[r_{ji}]=\begin{pmatrix} r_1 & r_2 \ldots & r_n \\ r_n & r_1 \ldots & r_{(n-1)} \\ & & \end{pmatrix}$$

$r_2 \quad r_3 \; \ldots . \quad r_1$

$$[\, r_{ji}] = circ\,(r_{1i}, r_{2i}, r_{3i}, \ldots, r_{ji})$$

So computing s_{ji} by :

$S_{ji} = circ\,(K_{seckey1}{}^{1}, K_{seckey1}{}^{2}, K_{seckey1}{}^{3} \ldots\ldots K_{seckeyn}{}^{m}) * circ\,(r_{1i}, r_{2i}, r_{3i}, \ldots, r_{ji}))$

3 Mathematical model of information security system

In this paper we relied on difficultly solved problem Diophantine equation (1) to build an asymmetrical cryptosystem.

$$f(x_1, x_2, \ldots, x_n) = 0 \qquad (1)$$

Where the solutions of equations in second-degree Pythagoras equation (2)

$$x^2 + y^2 = z^2 \qquad (2)$$

We are considered encryption and decryption the following class of its solutions from N (in more general case we can consider its solutions from Z or Q, where (m and n) are arbitrary natural numbers, and a > b.

$$x = a^2 - b^2, \;\; y = 2ab, \;\; c = a^2 + b^2 \qquad (3)$$

In encryptions we deal with equation as terms separately, it means we choose *x or y or c* to encryption the plain text in parameter of each term, where each parameter have two parameters the first is secret key which is generated by CEDH exchange protocol, while the second parameter is considered the plain text, each of term represent coding message in the group as the following steps:-

$E_0 = ((x = ((a^n)^2 - b^2)$, *where a=secret Key , b = plain text ,n= sequence users in their group.*

$E_1 = (y = (2(a)^n * b)$, *where a=secret Key , b = plain text ,n= sequence users in their group.*

$E_2 = (z = ((a^n)^2 + b^2))^n$, *where a=secret Key , b = plain text ,n= sequence users in their group.*

$\sum = \langle M = \{A, B, \ldots, Z/0 \ldots 9\}^*, KE\,(x,y,z)^n/M\,), KD\,(a,b/\,C\,)\,\rangle$

4 The proposed protocol for multi-group communications

Our proposal depended on Nathani et al. in 2018[21] but it is concentrated to improve important point confidentiality and authentication with best performance through using new approach to represent of Diophantine's equation (Pythagoras) where it is used as protocol to distribute of keys, so the main idea of our proposal rely on principle of multi-groups $\{G_1, G_2, G_3, \ldots\ldots, G_m\}$ each group containing participated n users $\{U_1, U_2, U_3, \ldots\ldots\ldots, U_n\}$ so user is required to registered at *KGC* which keeps track all participated and responsible for adding or removing any unsubscribed group participants. In order for all required tasks to be completed among multiple groups, *KGC* must define session keys for these groups and distribute their keys to all authorized and registered member, each according to its group. So just authorized member can easily derive this group's session key. Our distributed protocol Consist of four phases they are: **Initialization Phase, User Registration Phase, Multi-group key generation distribution and establishment Phase and Authentication Phase**.

I. Initialization Phase:

During this phase, *KGC* creates parameters *(p,a,b,G,n,h)* to generation shared secret key by the following steps:

I. Generating public number *p, G* to $User_i$ and *KGC*.
II. Selecting a private key for both parties $User_i$ is K_{priui} and *KGC* is k_{prikgc}.
III. Computing public key for both parties $User_i$, K_{pubui} and *KGC,* k_{prikgc} and exchange between of them .
IV. computing symmetric keys K_{seckey} for both parties $User_i$ and *KGC* and.
V. Prepar *h()*.

II. User Registration Phase:

I. $User_i$ requires registering at *KGC* which keeps track all participated and responsible for adding or removing any unsubscribed group's participants.
II. $User_i \in U_j \; (1 \le j \le n)$ and *KGC* get the shared secret key $K_{seckey} \in k \; (1 \le j \le n)$.

III. Multi-group key generation, distribution and establishment Phase:

In this Phase there a group of n participated $\{U_1, U_2, U_3, \ldots, U_n\}$ in multi-groups which are assumed as $\{G_1, G_2, G_3, \ldots, G_m\}$ where are communicating with *KGC* by their shared secret key over an insecure channel without sending it directly to each other. The process of Multi-group key generation, distribution and establishment five steps:

I. Firstly participants sends request a key generation to *KGC* which selects some users for each group as a list of groups $\{G_1, G_2, G_3 \ldots G_m\}$ so each list it can represented as $G_i = \{ U_1, U_2, U_3 \ldots U_j\}, (1 \le j \le n)$ *where* $j \in \{1, 2, \ldots, n\}$
II. *KGC* is advertising all of groups $\{G_1, G_2, G_3, \ldots, G_m\}$ to all participants as response.
III. *KGC* generates list of random numbers r_{ji} for $(1 \le j \le n, 1 \le i \le m)$ a corroding each participant $User_i \in U_j \; (1 \le j \le n)$ who joined his/her groups $G_i \; (1 \le j \le m)$.
IV. *KGC* saves the values of random number r_{ji}

which is used it to *make circulate matrices.*

V. KGC selects general key groups $k_{Gi}(1\leq i \leq m)$for all groups G_i $(1\leq j \leq m)$ and then it is computed $S_i(1\leq i \leq m)$ of each user U_j in each particular group G_i $(1\leq j \leq m)$ by the following strrctures of Diophantine equation in second degree (Pythagoras) :-

$E_0= ((x= (a^n)^2-b^2))$, *where* $a= K_{seckey}$, b = random number r_{ji} ,n= sequence *user in* their *group.*

$E_1 =(y = (2*(a^n)^*(b^2))$, *where* $a= K_{seckey}$, b = random number r_{ji} ,n= sequence *users in* their *group.*

$E_2= (z=(a^n)^2+b^2))$, *where* $a= K_{seckey}$, b = random number r_{ji} ,n= sequence *users in* their *group.*

Our proposed protocol have two value to compute s_{ji} the first vector of shared secret key K_{seckey} which is represented by term a while the second is Circulant matrix of random numbers r_{ji} which represented by b so according to terms of equation in our proposal is the following protocol:

$((s_{ji} = ((K_{seckey}{}^n)^2- (r_{ji})^2)$, $(s_{ji} =(2* (K_{seckey}{}^n * r_{ji})$, $(s_{ji} =((K_{seckey}{}^n)^2+ (r_{ji})^2)$

For example if we have m groups as $G_i=\{G_1,G_2,\ldots.G_m\}$ each group have n participate as $G_1=\{U_2,U_3,U_5,U_7\}$, to compute s_{ji} for each user we need make vector of shard secret key for U_2 in G_1 is $K_{seckey21} =\{ K_{seckey1}{}^1, K_{seckey1}{}^2, K_{seckey1}{}^3, K_{seckey1}{}^4\}$and needing value Circulant matrix $r_{ji} = \{ r_{21}, r_{31}, r_{51}, r_{71} \}$, now according of equation in our proposal terms of $G_1=\{x^2,y^2,z^2,x^2\}$,so

$U_2= ((K_{seckey2}{}^1)^2-circ(r_{ji})^2), ((K_{seckey2}{}^2)^2-circ(r_{ji})^2),((K_{seckey2}{}^3)^2-circ(r_{ji})^2), ((K_{seckey2}{}^4)^2- circ (r_{ji})^2)$

$U_3=(2*(K_{seckey3}{}^1)*circ(r_{ji})),(2*(K_{seckey3}{}^2)*circ(r_{ji})),(2*(K_{seckey3}{}^3)*circ(r_{ji})),(2*(K_{eckey3}{}^4)*circ(r_{ji}))$

$U_5 = ((K_{seckey5}{}^1)^2+circ(r_{ji})^2), ((K_{seckey5}{}^2)^2+ circ(r_{ji})^2), ((K_{seckey5}{}^3)^2+circ (r_{ji})^2), ((K_{seckey5}{}^4)^2+ circ(r_{ji})^2)$

$U_7=((K_{seckey7}{}^1)^2- circ(r_{ji})^2), ((K_{seckey7}{}^2)^2-circ(r_{ji})^2),((K_{seckey7}{}^3)^2-circ(r_{ji})^2), ((K_{seckey7}{}^4)^2-circ(r_{ji})^2)$

So for first user U_2 .Same steps repeat with other groups but in each time term is changed to encrypt and distributed keys.so we have different forms for secret key in one group.

$s_{21} =\{((K_{seckey2}{}^1)^2- circ (r_{21}, r_{31}, r_{51}, r_{71})^2) ,((K_{seckey2}{}^2)^2- circ (\{ r_{21}, r_{31}, r_{51}, r_{71})^2) ,((K_{seckey2}{}^3)^2- circ (r_{21}, r_{31}, r_{51}, r_{71})^2),((K_{seckey2}{}^4)^2- circ (r_{21}, r_{31}, r_{51}, r_{71})^2\}$

$s_{31} =\{2*((K_{seckey3}{}^1)* circ (r_{21}, r_{31}, r_{51}, r_{71})) ,(2*(K_{seckey3}{}^2)* circ (r_{21}, r_{31}, r_{51}, r_{71})^2) ,(2*(K_{seckey3}{}^3)* circ (r_{21}, r_{31}, r_{51}, r_{71})) ,(2*(K_{seckey1}{}^4)* circ (r_{21}, r_{31}, r_{51}, r_{71}))\}$

$s_{51} =\{((K_{seckey5}{}^1)^2+ circ (r_{21}, r_{31}, r_{51}, r_{71})^2),((K_{seckey5}{}^2)^2+ circ (r_{21}, r_{31}, r_{51}, r_{71})^2) ,((K_{seckey5}{}^3)^2+ circ (r_{21}, r_{31}, r_{51}, r_{71})^2) ,((K_{seckey5}{}^4)^2+ circ (r_{21}, r_{31}, r_{51}, r_{71} r_{41})^2) \}$

$s_{71} =\{((K_{seckey7}{}^1)^2- circ (r_{21}, r_{31}, r_{51}, r_{71})^2) ,((K_{seckey7}{}^2)^2- circ (r_{21}, r_{31}, r_{51}, r_{71})^2) ,((K_{seckey7}{}^3)^2- circ (r_{21}, r_{31}, r_{51}, r_{71})^2),((K_{seckey7}{}^4)^2- circ (r_{21}, r_{31}, r_{51}, r_{71})^2) \}$

VI. KGC computes addition values:

$u_{ji}= S_i-s_{ji}$, *where* $S_i = Circ(K_{Gi}{}^1,K_{Gi}{}^2,\ldots\ldots,K_{Gi}{}^j)$ *for* $(1\leq j \leq n,1\leq i \leq m)$and $Auth_i=h(K_{Gi}, U_1,U_2,U_3\ldots U_j, r_{1i}, r_{2i} ,\ldots\ldots, r_{ji} , u_{1i} , u_{2i,\ldots\ldots}, u_{ji})$.At last, finally KGC is advertising $(Auth_i ,(u_{ji})_{Gi}$ *for* $(1\leq j \leq n,1\leq i \leq m)$.Here, i represent number of groups and j represents number of participants in each group G_i.

IV. Authentication phase:

1. Now each participating group member U_j $(1\leq j \leq n)$ knowing their corresponding public value u_{ji} in each particular group G_i $(1\leq j \leq m)$ is able to compute the value of s_{ji} according of own term but firstly we need compute share secret key K_{seckey} so :
$$K_{seckey} = K_{seckey}{}^*$$
2. If K_{seckey} authorized, now compute the value of s_{ji} according of own term as:
$((K_{seckeyj}{}^i)^2- circ (r_{ji})^2)$,$(2*((K_{seckeyj}{}^i)* circ (r_{ji})))$, $((K_{seckeyj}{}^i)^2+ circ (r_{ji})^2$
3. Recover the group key K_{Gi} by computing, $S_i = u_{ji} +s_{ji}$ this is of the form $S_i =Circ(K_{Gi}{}^1,K_{Gi}{}^2,\ldots\ldots,K_{Gi}{}^j)$ *for* $(1\leq j \leq n,1\leq i \leq m)$.
4. Each (u_{ji}) *for* $(1\leq j \leq n,1\leq i \leq m)$ authenticates their corresponding groups G_i by computing : $Auth_i=h(K_{Gi}, U_1,U_2,U_3\ldots U_j, r_{1i}, r_{2i} ,\ldots\ldots, r_{ji} , u_{1i} , u_{2i,\ldots\ldots}, u_{ji})$ for $(1\leq j \leq n,1\leq i \leq m)$ and then checks this value by:
$$Auth_i = Auth_i{}^*$$
If this result is correct then each participant U_j $(1\leq j \leq n)$ in the group G_i $(1\leq j \leq m)$ authenticates the group key k_{Gi} is sent from KGC.

5 Security analyses

Our proposal protocol has security features which are analyzed in this section:

1. **Theorem 1** *This protocol provides important feathers with key freshness, key confidentiality and key authentication.*

   Key freshness: our proposal is remained renewed with each new communication session by *m* group key $S_i = Circ(K_{Gi}^1, K_{Gi}^2, \ldots, K_{Gi}^j)$ *for* $(1 \le j \le n, 1 \le i \le m)$ which it can be generated with multi group G_i $(1 \le i \le m)$ depended on *KGC* whenever requested. As well as new secret key s_{ji} for each U_j $(1 \le j \le n)$ by compute secret key K_{seckey} $(1 \le j \le n)$ with random number r_{ji} for $(1 \le j \le n, 1 \le i \le m)$ in the group G_i $(1 \le i \le m)$, so it is generated new s_{ji} secret key for each new communication service as request.

   Key confidentiality: Our proposal is provide security feature based on combination techniques SSS and Circulant Matrix and an increase in security complexity some techniques are integrated by using *CEDH* exchange protocol for each $U_j \in$ $(1 \le j \le n)$ and shared with *KGC*, that means just authorized member can recover the shared secret key K_{seckey}. Second technique it is used terms of Diophantine equations where each term represent two parameter the first is secret key K_{seckey} which is represented by term *a* as vector while the second parameter is random numbers r_{ji} which is represented by Circulant Matrix of *b*. So we have different forms for secret key s_{ji} in one group, and cannot get keys $S_i = Circ(K_{Gi}^1, K_{Gi}^2, \ldots, K_{Gi}^j)$ *for* $(1 \le j \le n, 1 \le i \le m)$, $S_i = u_{ji} + s_{ji}$ unless a number of important values are available as K_{seckey} and r_{ji} as well as how to calculate s_{ji} by $((K_{seckeyj}^{i})^2 - circ(r_{ji})^2)$,($2*((K_{seckeyj}^{i})* circ(r_{ji})))$, $((K_{seckeyj}^{i})^2 + circ(r_{ji})^2$.

   Key authentication: in our proposal able to examine at the first time if the participant authorized or not by K_{seckey} :

   $$K_{seckey} = K_{seckey}^*$$

   If K_{seckey} authorized now it is recovered the group key K_{Gi} by computing, s_{ji} and then $S_i = u_{ji} + s_{ji}$ where $S_i = Circ(K_{Gi}^1, K_{Gi}^2, \ldots, K_{Gi}^j)$ *for* $(1 \le j \le n, 1 \le i \le m)$. each ,(u_{ji}) *for* $(1 \le j \le n, 1 \le i \le m)$ authenticates their corresponding groups *Gi* by computing : $Auth_i = h(K_{Gi}, U_1, U_2, U_3 \ldots U_j, r_{1i}, r_{2i}, \ldots, r_{ji}, u_{1i}, u_{2i}, \ldots, u_{ji})$ for $(1 \le j \le n, 1 \le i \le m)$ and then checks this value by:

   $$Auth_i = Auth_i^*$$

   If this result is correct then each participant U_j $(1 \le j \le n)$ in the group G_i $(1 \le j \le m)$ authenticates the group key k_{Gi} is sent from KGC so our proposal has to condition to complete authentication phase.

2. **Theorem 2** *our proposed protocol can resist the attacks in both synchronous and asynchronous networks.*
   Proof: Since *KGC* responsible for generation K_{seckey} for U_j $(1 \le j \le n)$ and retrieving information by $S_i = u_{ji} + s_{ji}$, so attacker can't obtain any sense information because we depended on shared secret key K_{seckey} by CEDH between user and KGC.
3. **Theorem 3** *our proposed protocol achieves the back- ward secrecy and the forward secrecy.*
   Proof: because of K_{seckey} for U_j $(1 \le j \le n)$ and refresh r_{ji} which shared just with *KGC* in each session any unauthorized or old member left his/her group they can't join to their groups unless they get the sense information.
4. **Theorem 4** *our proposed protocol can resist the outside attacks.*
   Proof: because there is more than one sense information that secures the entry of any participant into their group, any outside attack fails to penetrate this system because it needs more than obtaining the group key only, as it needs the key K_{seckey} in addition to the Circulant Matrix of random numbers $(r_{1i}, r_{2i}, \ldots, r_{ji})$, as well as its need to guess which one the terms by that the keys are generated(((($K_{seckeyj}^{i}$ $)^2$- $circ(r_{ji})^2$) ,($2*((K_{seckeyj}^{i})* circ(r_{ji})))$, (($K_{seckeyj}^{i}$ $)^2 + circ(r_{ji})^2$ each these elements are difficult for immigrants to obtain together so our proposed protocol can resist the outside attacks .
5. **Theorem 5** *our proposed protocol can resist the inside attacks.*
   Proof: because of any user U_j $(1 \le j \le n)$ has different K_{seckey} , r_{ji} , $((K_{seckeyj}^{i})^2 - circ(r_{ji})^2)$,($2*((K_{seckeyj}^{i})* circ(r_{ji})))$, $((K_{seckeyj}^{i})^2 + circ(r_{ji})^2$ in the same group G_i can't other participant to obtain sense information for another in the same group so our proposal can provide this feature.

6 Comparison with related work

It is applied new protocol on Nathani's scheme but in different approach by using Diophantine equation where we are focused about three point refresh key, confidinality key and authentication key by new representing of Diophantine equation(Pythagoras) with ECDH protocol for generation of secret key so in this section will compare new point in our protocol.

6.1 Comparison 1

Fist point we want to compare a long-term secret key in [19] [21] [22] where the authors just mentioned use a long-term secret key and shared with *KGC* in secure manner without more details so we can guess maybe the long-term secret key not secure so in our protocol we use ECDH protocol for generation of secret key which it provided at the beginning whether the participant is trusted or not, adding to its tracking in the event of addition or deletion. As for the length of the key the least is 4 byte. In this paper we camper our protocol with [21] in term of performance where we use a long-term secret key and random number as the author mentioned in his example as simple numbers but the result was simple keys within close groups compared to our results as explain 1 figure.

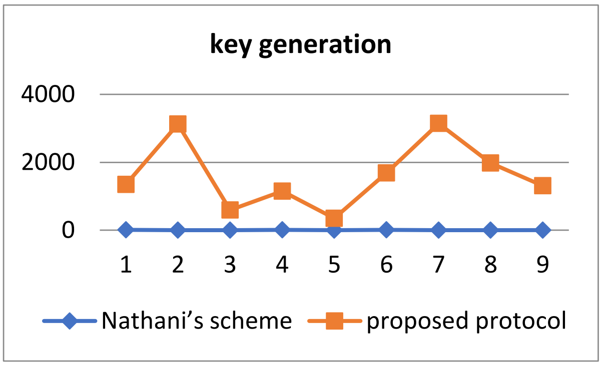

Figure 1: Comparison key generation with Nathani's scheme.

While we use privet key as a long-term secret key and public key as a random number the Nathani's scheme spent 12.9 minute when it is ran with 10 group and each of group have 10 all groups are generated 1000 key, while minimum when it is ran with 3 groups and each group have 3 to generate 27 K_{seckey} , spent 0.04 second as explain in figure 2.

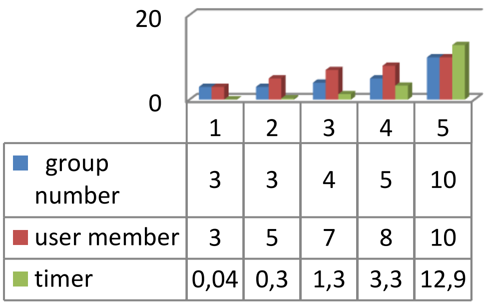

Figure 2: Complexities Nathani et al schema.

In our proposed protocol along secret key as a result of a combination of privet key and public key which is spent 1.8 minute when it is ran with 10 groups and groups are generated 1000 K_{seckey} as maximum. While minimum when it is ran with 3 groups and each group have 3 user to generate 27 K_{seckey} 0.02 second. The following diagram is explained our proposal result as explain in figure 3.

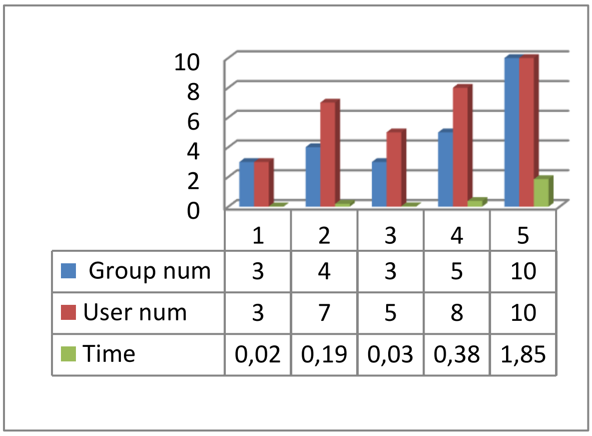

Figure 3: Complexities of our proposal's schema.

Table 1: Explain the values and difference a resulted between our proposed protocol and Nathani et al schema

Group No.	User No.	Nathani et al schema	our proposal's schema
3	**3**	**0.04**	**0.02**
4	**7**	**0.3**	**0.19**
3	**5**	**1.3**	**0.03**
5	**8**	**3.3**	**0.38**
10	**10**	**12.9**	**1.85**

6.2 Comparison 2

The performance consist many important point as communication cost, computational costs and storage requirement. We discuss our protocol compared with [19] [21] [22].

Communication cost means total volume of data transmission during user registration phase and multi-group keys establishment phase .Where $|p|$ is the size of the adopted finite fieldGF(p), and $|h|$ is the output size of one-way hash function. In [19] produce single group key protocol in multi time where there are n user m groups where communication cost is $\sum_{i=1}^{m} t_i\,|pq| + 2t_i|pq| + |h|$ while [22] produce multi group key protocol in multi time where there are n user m_i groups where communication cost is $1 + m|h| + \sum_{i}^{n} m_i|p| + 2m_i\,|p|$. In [21] produce multi group key protocol in multi time that means it has been expanded group's key establishment and communication cost is $\sum_{i}^{k}\sum_{j}^{n} m_{ij}\,|p| + |h|$. So, in our protocol we rely on [21] as communication cost which is $\sum_{i}^{k}\sum_{j}^{n} \boldsymbol{m}_{ij}{}_{\pm}^{*}|\boldsymbol{p}| + \boldsymbol{m}|\boldsymbol{h}|$but in different approach to be more secure with the same cost as table 2 explain that.

Table 2: Communication cost for [19] [21] [22] and our protocol

Scheme	Initialization Phase	User registration key	Group key generation and distribution	Total
Harn's et al scheme	$2n\|pq\|$	$(x_i, y_i)key$	$\sum_{i=1}^{m} t_i \|pq\| + 2t_i\|pq\| + \|h\|$	$2n\|pq\| + 3\sum_{i=1}^{m} t_i \|pq + m\|h\|$
Hus's et al scheme	$n\|p\|$	$(x_i)key$	$1 + m\|h\| + \sum_{i}^{n} m_i\|p\| + 2m_i\|p\|$	$n\|p\| + 1 + m\|h\| + 3\sum_{i}^{n} m_i\|p\|$
Nathani et al scheme	$n\|p\|$	$(x_i)key$	$\sum_{i}^{k}\sum_{j}^{n} m_{ij}\|p\| + \|h\|$	$n\|p\| + \sum_{i}^{k}\sum_{j}^{n} m_{ij}\|p\| + m\|h\|$
Our Proposed protocol	$n\|p,\mathrm{a},\mathrm{b},\mathrm{G}\|$ ECDH exchange	$(x_i)key$	$\sum_{i}^{k}\sum_{j}^{n} m_{ij\overset{*}{\pm}}\|p + \|h\|$	$n\|p\| + \sum_{i}^{k}\sum_{j}^{n} m_{ij\overset{*}{\pm}}\|p\| + m\|h\|$

Cost of computation TM, TI, and TH as the execution times for a one-way hash function, a modular multiplication, and a modular inverse, respectively. The time required for executing modular addition or subtraction in the suggested approach can be disregarded in comparison to TM or TI, it's clear to a count of secret key x , key K and random number r in general .

In [21] explain difference the Cost of computation of [19] and [21] as mentioned table 3 in [22] and our protocol it is clear computation cost is less than [22] because diversity between modular addition and subtraction not just modular multiplication as [22] use. The time for performing modular addition or subtraction required in the proposed scheme can be ignored [21]. So the computational cost of our protocol less. Table 3 explains that.

Table 3: computational cost of [19][21][22] and our protocol

scheme	Distributing the group key	Recovering the group key
Harn's et al scheme	$\sum_{i=1}^{m}(t_i * (t_i + 1) * t_i * (TM + TI) + TH$	$(t_i + 1) * t_i * (TM + TI) + TH$
Hus's et al scheme	$MTH + \sum_{i=1}^{n} m_i * (m_i + 1) * m_i * (TM + TI) + m_i TH$	$(m_i + 1) * m_i * (TM + TI) + m_i TH$
Nathani et al scheme	$\sum_{i=1}^{k} \sum_{j=1}^{n}(m_i * m_{ij}) * m_i * m_i(TM + TI) + m_i TH$	$m_i * (TM + TI) + m_i TH$
Our Proposed protocol	$\sum_{i=1}^{k} \sum_{j=1}^{n}(m_i \overset{*}{\pm} m_{ij}) * m_i * (TM + TI) + m_i TH$	$m_i * (TM + TI) + m_i TH$

storage requirement in [19] using two keys *(x, y)* while in [21][22] using a long secret key *x* as in our protocol in spite of is generated user key by ECDH protocol except if use and save just one key.

7 Conclusion

In this research, we intended to improve two main points in the algorithm, which are security and reliability, by generating K_{seckey} in the ECDH method and confidentiality by using the terms of the Diophantine equation to generate (($K_{seckeyj}^{i}$)^2- *circ* (r_{ji})^2) ,(2*(($K_{seckeyj}^{i}$)* *circ* (r_{ji}))), (($K_{seckeyj}^{i}$)^2+ *circ* (r_{ji})^2. According to the analysis and comparison, this led to the improvement of the previous algorithm in terms of security and reliability, in addition to improve performance by using the same communication cost but with less computational cost by complicated approach to get safe manner to distributed group key with expanded multi group. In addition, we explain this result by programming the last algorithm [22] and our protocol with use the Python language.

References

[1] L. Harn and C. Lin, "Authenticated group key transfer protocol based on secret sharing," IEEE Trans. Comput., vol. 59, no. 6, pp. 842–846, 2010, doi: 10.1109/TC.2010.40.

[2] K. Meng, F. Miao, and Y. Yu, "A secure and efficient on-line/off-line group key distribution protocol," Des. Codes, Cryptogr., vol. 87, no. 7, pp. 1601–1620, 2019, doi: 10.1007/s10623-018-0554-6.

[3] A. Shamir, "New directions in croptography," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 2162, p. 159, 2001, doi: 10.1007/3-540-44709-1_14.

[4] I. Ingemarsson, D. T. Tang, and C. K. Wong, "A Conference Key Distribution System," IEEE Trans. Inf. Theory, vol. 28, no. 5, pp. 714–720, 1982, doi: 10.1109/TIT.1982.1056542.

[5] D. G. Steer, L. Strawczynski, W. Diffie, and M. Wiener, "A secure audio teleconference system," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 403 LNCS, pp. 520–528, 1990, doi: 10.1007/0-387-34799-2_37.

[6] E. Bresson, O. Chevassut, and D. Pointcheval, "Dynamic group diffie-hellman key exchange under standard assumptions," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 2332, pp. 321–336, 2002, doi: 10.1007/3-540-46035-7_21.

[7] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman key distribution extended to group communication," Proc. ACM Conf. Comput. Commun. Secur., pp. 31–37, 1996, doi: 10.1145/238168.238182.

[8] J. Bohli, "A Framework for Robust Group Key Agreement," pp. 355–356, 2006.

[9] E. Bresson, O. Chevassut, and D. Pointcheval, "Provably secure authenticated group Diffie-Hellman key exchange," ACM Trans. Inf. Syst.

Secur., vol. 10, no. 3, pp. 1–45, 2007, doi: 10.1145/1266977.1266979.
[10] J. Katz and M. Yung, "Scalable protocols for authenticated group key exchange," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 2729, pp. 110–125, 2003, doi: 10.1007/978-3-540-45146-4_7.
[11] W. G. Tzeng, "A practical and secure fault-tolerant conference-key agreement protocol," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 1751, no. 4, pp. 1–13, 2000, doi: 10.1007/978-3-540-46588-1_1.
[12] J. C. Cheng and C. S. Laih, "Conference key agreement protocol with non-interactive fault-tolerance over broadcast network," Int. J. Inf. Secur., vol. 8, no. 1, pp. 37–48, 2009, doi: 10.1007/s10207-008-0062-1.
[13] K. H. Huang, Y. F. Chung, H. H. Lee, F. Lai, and T. S. Chen, "A conference key agreement protocol with fault-tolerant capability," Comput. Stand. Interfaces, vol. 31, no. 2, pp. 401–405, 2009, doi: 10.1016/j.csi.2008.05.015.
[14] C. F. Hsu, L. Harn, Y. Mu, M. Zhang, and X. Zhu, "Computation-efficient key establishment in wireless group communications," Wirel. Networks, vol. 23, no. 1, pp. 289–297, 2017, doi: 10.1007/s11276-016-1223-1.
[15] L. H. Chi Sung Laih, Jau Yien Lee, "No Title," Inf. Process. Lett., vol. 32, no. 3, pp. 95–99, 1989, [Online]. Available: https://www.sciencedirect.com/science/article/pii/0020019089900082.
[16] G. Sáez, "Generation of key predistribution schemes using secret sharing schemes," Discret. Appl. Math., vol. 128, no. 1, pp. 239–249, May 2003, doi: 10.1016/S0166-218X(02)00448-1.
[17] C. H. Li and J. Pieprzyk, "Conference key agreement from secret sharing," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 1587, pp. 64–76, 2010, doi: 10.1007/3-540-48970-3_6.
[18] Kamesh and N. Sakthi Priya, "A survey of cyber crimes Yanping," Secur. Commun. Networks, vol. 5, no. June, pp. 422–437, 2012, doi: 10.1002/sec.
[19] L. Harn and C. Lin, "Efficient group Diffie-Hellman key agreement protocols," Comput. Electr. Eng., vol. 40, no. 6, pp. 1972–1980, 2014, doi: 10.1016/j.compeleceng.2013.12.018.
[20] R. F. Olimid, "Cryptanalysis of a password-based group key exchange protocol using secret sharing," Appl. Math. Inf. Sci., vol. 7, no. 4, pp. 1585–1590, 2013, doi: 10.12785/amis/070444.
[21] S. Nathani, B. P. Tripathi, and S. K. Bhatt, "UMK Gm TP : User Friendly Multi Group Key Transfer Protocol with Circulant Matrices," 2018.
[22] C. F. Hsu, L. Harn, and B. Zeng, "UMKESS: user-oriented multi-group key establishments using secret sharing," Wirel. Networks, vol. 26, no. 1, pp. 421–430, 2020, doi: 10.1007/s11276-018-1825-x.
[23] V. Osipyan, "Different models of information protection system, based on the functional knapsack," ACM Int. Conf. Proceeding Ser., pp. 215–218, 2011, doi: 10.1145/2070425.2070461.
[24] S. Devi, "A study on system of linear diophantine equations," vol. 2, no. 4, pp. 637–639, 2017.
[25] V. O. Osipyan, K. I. Litvinov, R. K. Bagdasaryan, E. P. Lukashchik, S. G. Sinitsa, and A. S. Zhuk, "Development of information security system mathematical models by the solutions of the multigrade diophantine equation systems," ACM Int. Conf. Proceeding Ser., 2019, doi: 10.1145/3357613.3357624.
[26] R. Haakegaard and J. Lang, "The elliptic curve diffie-hellman (ECDH)," Retrieved Febr. 10, 2020, from http//koclab.cs.ucsb.edu/teaching/ecc/project/2015Projects/Haakegaard+Lang.pdf, no. December, p. 4, 2015.
[27] B. N. Koblitz, "Elliptic Curve Cryptosystems," vol. 4, no. 177, pp. 203–209, 1987.
[28] A. Shamir, "How to Share a Secret," Commun. ACM, vol. 22, no. 11, pp. 612–613, 1979, doi: 10.1145/359168.359176.