Improved Group Key Agreement for Emergency Cognitive Radio Mobile Ad hoc Networks

N. Renugadevi* and C. Mala
Department of Computer Science and Engineering
National Institute of Technology, Tiruchirappalli
Tamil Nadu, India
nrenu79@gmail.com, mala@nitt.edu

Abstract

Cognitive radio based networks can resolve the spectrum scarcity issues present in the current emergency communication networks. Establishing quick and secure communication between rescue team members in the disaster field is a significant task and cognitive radio MANETs are the best choice for disaster management and other emergency situations. This paper presents a scalable group key agreement protocol called 'ITGECDH.2' which uses an improved batch rekeying scheme. It is inferred from the performance analysis that the proposed ITGECDH.2 is suitable for secure group communication in emergency cognitive radio MANETs with inherent dynamism as it reduces the number of renewed nodes and incurs reduction in rekeying time.

Keywords: Group key agreement, Batch rekeying, Cognitive radio, MANETs, Emergency

1 Introduction

The current *Emergency Communication Networks (ECNs)* mostly rely on infrastructure based public radio networks like GPRS/GSM. These types of networks may be destroyed during natural disasters and hence they are not reliable for emergency situations. So, the infrastructure-less wireless ECN is used for quick deployment of emergency communication in disaster sites [1].

The main limitation of existing infrastructure-less ECN is spectrum scarcity and this availability of limited spectrum can not fulfill the requirements of growing emergency communication services [2]. *Cognitive Radio (CR)* [3] performs *Dynamic Spectrum Access (DSA)* [4] in which the unlicensed CR or *Secondary Users (SUs)* opportunistically access the unused portions of the licensed spectrum called *white spaces* without impacting the licensed *Primary Users (PUs)*. CR can work in different frequency bands and can support a wide variety of multimedia services to handle large, unpredictable amount of emergency information [5].

Uchida et al. developed a Never Die Network [6] which incorporates CR networks and satellite network to provide a strong network connection in the case of disaster similar to Great East Japan Earthquake on March 2011. The optimal wireless links and routes are selected through cognition cycle and extended analytic hierarchy process by considering the changes in user policy and network environment.

*Corresponding author: Department of Computer Science and Engineering, National Institute of Technology, Tiruchirappalli-620015, Tamil Nadu, India, Tel: +91-9952843377

The *Mobile Ad hoc NETwork (MANET)* can be used to set up the communication environment between rescue teams as quickly as possible [7]. As *CR based MANET (CRMANET)* gives solution to spectrum scarcity issues through dynamic utilitation of the spectrum, securing CRMANET is a major issue [8]. The rescue team members should perform secure group communication to transfer critical information in the disaster site and hence protecting the user privacy is important. The sensitive information can be protected from malicious users by means of access control.

The common key establishment for the members of rescue team is the foundation for securing *Emergency CRMANETs (ECRMANETs)*. As central controller is not present in ECRMANETs, it can employ *Group Key Agreement (GKA)* scheme for generating a shared key. The GKA scheme generates a group key from the shares contributed by every group members.

Batch Rekeying (BR) modifies the current group key for the collection of join as well as leave requests. Lee et al. [9] compared BR scheme with *Individual Rekeying (IR)* scheme and proved that cost of BR scheme is lower than IR cost. The BR scheme is more appropriate than IR in dynamic ECRMANETs with frequent membership changes.

A logical key tree with ‘n’ member is used in scalable *tree based GKA* technique, in which the complexity involved in group operation is reduced to O(log n) from O(n) [10]. Therefore, this paper improves the BR scheme used in the ternary tree based GKA called *TGECDH.2* [11] and proposes an *Improved TGECDH.2* called *ITGECDH.2* which can be adopted in ECRMANETs.

The rest of this paper is organised as follows. Several GKA protocols are explained in Section 2. Section 3 briefly explains the existing TGECDH.2 and describes the proposed ITGECDH.2 protocol. Section 4 analyses the total number of operations required for computing the initial group key. The performance of ITGECDH.2 is compared with TGECDH.2 in Section 5. The conclusion is given in Section 6.

2 Literature Review

GKA protocols available in the literature which are also applicable to ECRMANETS are described in this section.

An Efficient Group Diffie Hellman (GDH) [12] protocol was proposed in [13] which combines secret sharing scheme with DH, where an one-way hash function is used for providing authentication by generating the key confirmation. Recently, Odelu et al. has presented a client authentication protocol [14] in which Elliptic Curve Cryptography (ECC) based GKA is employed and it preserves the user anonymity.

Sun et al. has presented a GKA protocol for mobile environments [15] in which short certificateless signature scheme has been adopted for providing user authentication. This protocol reduces the cost associated with certificate management and the key escrow problem is avoided by employing certificateless public key cryptography.

A GK management technique using the principle of hyper-sphere was proposed in [16], where the distance between any point and the center point is identical. The member of group is represented by an unique point on the hyper-sphere which is its own private key. The group controller calculates and publishes the center point. Every group member then determines group key by considering its own private point and the center point. This scheme outperforms well than other similar works by reducing the user storage space, amount of computation at user side and number of rekeying information.

Jung et al. presented a GKA protocol [17] for medical emergency environments in ad hoc networks. The nodes with different levels of computing capabilities such as handheld devices, PDAs, ambulance cars, etc were considered and the key tree was constructed accordingly.

Chen et al. modified the existing CLIQUES protocol [18] and developed a new GKA called M-CLIQUES [19] for secure multicast communication. In the initial stage called Static CLIQUES, a group controller distributes the partial keys. Several multi-level sub groups are formed using hierarchical structure in order to support the scalability in the second stage called Hierarchical CLIQUES. By maintaining this hierarchical tree structure of sub groups, the amount of keys transmitted by sub group controller was reduced. This protocol employs interval based rekeying operations to reduce the overhead in rekeying.

The research work in [20] proposed synchronization and nonce based authenticated GKA schemes using extended chaotic maps and biometric key for telecare medical information systems. The proposed two schemes protect privacy of users and require fewer number of transmissions with less computational cost.

The group controller should distribute new keys when the membership changes to maintain the confidentiality in secure group communication. The group controller of existing key management techniques brodcasts the key updates even when some users need them. The key distribution algorithm of proposed scheme in [21] solves this problem by distributing updated keys only to the subset of group members. The proposed algorithm includes a descendant tracking scheme which tracks the multicast tree for downstream users and forwarding method which forwards updated keys. The algorithm which assigns an identifier to group members enhances the performance of proposed key distribution scheme which reduces the bandwidth overhead involved in broadcast mechanism.

Xue et al. proposed a Dynamic Secure Group Sharing Framework (DSGSF) [22] for public cloud computing with improved tree based TGDH protocol [10]. This paper applies proxy signature to grant the privilege of managing the group to specific group members.

GKA protocols enable a set of group members to form a shared symmetric key. In Dynamic Asymmetric GKA (DAGKA), a dynamic set of members in a temporary group can establish a public encryption key which allows the outsiders to securely send message to this temporary group. The authors in [23] presented the first scheme of DAGKA by combining authenticated GKA, multi-signature technique and public key encryption method. After computing the shared private key, its corresponding public key is sent to outsiders.

3 Proposed ITGECDH.2 Protocol

Subsection 3.1 explains briefly the existing TGECDH.2 protocol [11] and the proposed ITGECDH.2 protocol with improved BR scheme is discussed in Subsection 3.2.

3.1 Existing TGECDH.2 protocol

The structure of key tree influences the performance of rekeying technique in tree based GKA scheme. The optimal key tree for batch updates in the group with any size is a ternary key tree [24]. Hence, TGECDH.2 protocol has adopted the ternary key tree for batch updates during GKA. Figure 1 [11] depicts the initial key tree, where the leaf nodes (M1-M9) represent the group members, i.e., SUs in ECRMANETs. The intermediate nodes have subgroup keys and the key for entire group is stored in the root node. The rightmost node in each subtree with three members is the sponsor of the corresponding subgroup. The entire group sponsor (rightmost node) in the key tree computes initial group key.

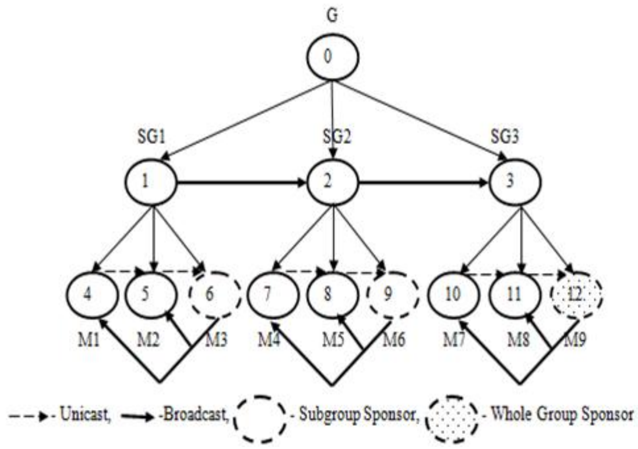

Figure 1: Initial key tree of TGECDH.2

TGECDH.2 protocol has employed two key trees, namely Main Key Tree (MKT) and Temporary Key Tree (TKT) for existing and new members respectively. In addition to this, it has used a queue for new members. TGECDH.2 uses three algorithms, such as a) Ternary_Batch_Process (TBP), b) Ternary_Batch_Merge (TBM) and c) Ternary_Join_Tree (TJT). This protocol considers two phases, namely 1) Preprocess phase and 2) Merge phase.

Let 'L' and 'J' be the total leave and join members in the group. Both join as well as leave requests received are maintained in a queue. At the beginning of next rekey interval, the values of L and J are compared. TBP algorithm uses two cases in its preprocess phase for the possible join and leave operations in the group. It uses Case 1 and Case 2 when $L = 0$ and $L > 0$ respectively. TJT algorithm creates a TKT for new members in Case 1. Then, it checks Shallowest Intermediate Nodes (SINs) in the key tree when the root is full for finding an appropriate Insertion Point (IP) to connect TKT with MKT without increasing the height of MKT.

The first three sub cases of Case 2 in TBP algorithm use a queue for handling the group operations. The TKT is created only in the last subcase in Case 2, i.e., only when TGECDH.2 protocol has more joining members than leaving members. Thus, it has reduced the key tree operations which in turn reduced the complexity in rekeying operations. During merge phase, the newly created TKT is inserted into MKT at chosen IP and then all group members recompute the group key.

3.2 Proposed ITGECDH.2 protocol

The preprocess phase of TGECDH.2 protocol is improved in order to make it well suitable for emergency communications. The ITGECDH.2 protocol suggests some modifications in TBP algorithm.

The modifications introduced in both the Cases of TBP algorithm are as follows:- Case 1 checks SINs for finding an appropriate IP both when the root is full or not full. In Case 2, i.e., when $L < J$, the leave members are pruned instead of being replaced by join members. This reduces the height of current key tree and thus minimizes the number of operations in the subsequent rekeying interval. All the leave members except the one with minimum ID are pruned from the key tree. The location of leave member with minimum ID is chosen as IP. If the insertion of TKT at this IP increases the MKT's height, then a new IP is selected using Case 1.

The improved_TBP() is given in Algorithm 1, which uses the abbreviations, namely Root (R), Child or Children (C), Height (Ht), Node ID (ID), MinID (Minimum ID), MinHt (Minimum Height), New MKT (NMKT), Leave Members (LMs), Leave Member with ID 'i' (LM_i), SIBlings (SIB), Parent node (P) and First Leave Member (FLM).

Algorithm 1 Improved_TBP(MKT, J, L)

```
1:  Begin
2:      Case 1: If ($J > L$ & $L = 0$) then
3:       Begin Case
4:           Create a TKT
5:           If ($C(R) \neq \phi$ or $C(R) = \phi$) then
6:              If ($C(SIN) = \phi$) then
7:                 $IP \leftarrow MinID(SIN)$
8:              Else
9:                 $IP \leftarrow MinHt(SIN)$
10:             Endif
11:             If ($Ht(NMKT) > Ht(MKT)$) then
12:                $IP \leftarrow R$
13:             Endif
14:          Endif
15:      End Case
16:     Endif
17:     Case 2: If ($L > 0$) then
18:      Begin Case
19:          $FREE \leftarrow$ ID's (LMs)
20:          If $LM_i$ & $SIB(LM_i)$ in FREE then
21:              $FREE \leftarrow FREE \setminus \{LM_i \text{ \& } SIB(LM_i)\} \bigcup P(LM_i)$
22:          Endif
23:          Sort 'FREE'
24:          Create a TKT
25:          $IP \leftarrow$ FLM (FREE)
26:          If ($Ht(NMKT) > Ht(MKT)$) then
27:              IP= go to 5
28:          Endif
29:      End Case
30:     Endif
31: End
```

Figure 2 shows MKT with 4 leave members denoted by X symbol and the new key tree after merging TKT created for six join members (M12-M17). The Case 2 of Improved_TBP algorithm first selects M5 with ID 7 as an IP to insert the TKT created for six new members as it has minimum ID than other leave members. But, this insertion will increase the height of current MKT. Therefore, it executes Case 1 and checks the root node followed by all the SINs of the key tree.

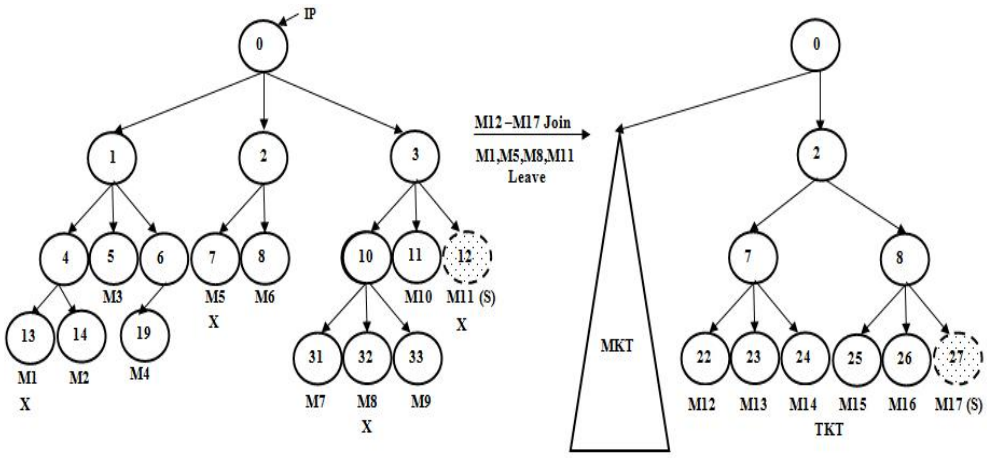

Figure 2: Insertion of TKT with six join members into MKT

Even though the middle SIN has null link, the height of current MKT will be increased if TKT is inserted under this SIN. Therefore, the root of MKT is considered as IP and TKT is inserted as a middle sub tree. Thus, the proposed Improved_TBP() tries to keep the tree balanced so that the number of operations in the key tree are minimized. In this figure, 'S' indicates the sponsor which updates the current group key during membership change.

4 Proposed mathematical analysis

This section analyses the metric called '*Number of operations*' which is used in measuring the computational performance of proposed protocol in initialization process through mathematical expressions.

The proposed protocol 'ITGECDH.2' employs GECDH.2 [11], where the costlier modular exponentiation operation in GDH.2 [12] is replaced by cost-efficient *point multiplication* of ECC. The metric, number of operations indicates the total number of point multiplications required for calculating the group key.

Figure 3 shows the computation of group key using *GECDH.2* for the subgroup with three members. The modular exponentiation in the original GDH.2 has been replaced by point multiplication. In this figure, a, b and c are secret keys of three members M1, M2 and M3 respectively, and G is the Base point (i.e., Generator). The member M1 unicasts (dotted line) its public key (a.G) to its neighbour M2. After computing the intermediate key (ab.G), M2 unicasts it along with its public key (b.G) and (a.G), i.e., the message containing (ab.G), (a.G), (b.G) is sent from M2 to M3. The sponsor or group controller M3 computes the group key (abc.G) and intermediate keys using its secret key (c) and the message received from M2. Then, other members M1 and M2 will compute group key upon receiving the intermediate keys (ac.G), (bc.G) broadcast (dark lines) by the sponsor M3.

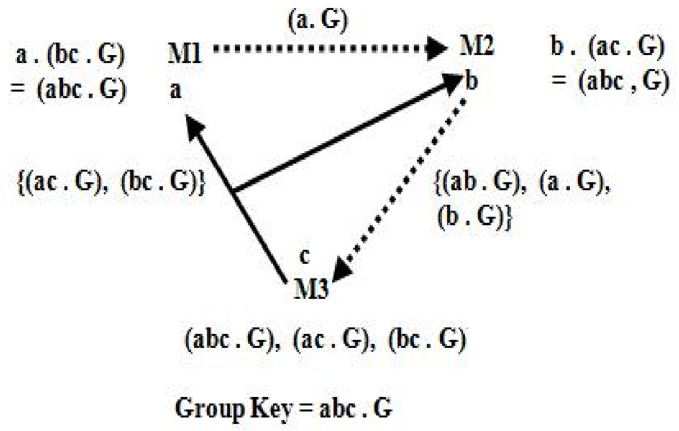

Figure 3: GECDH.2 protocol

The illustration of group key calculation is given in Figure 4. All the members perform point multiplication either using *point addition* or *point doubling* to compute the group key (29,72). As ECC requires smaller key size and performs more efficient computation, it is considered to be most suitable for smaller, mobile devices [25] in ECRMANETs.

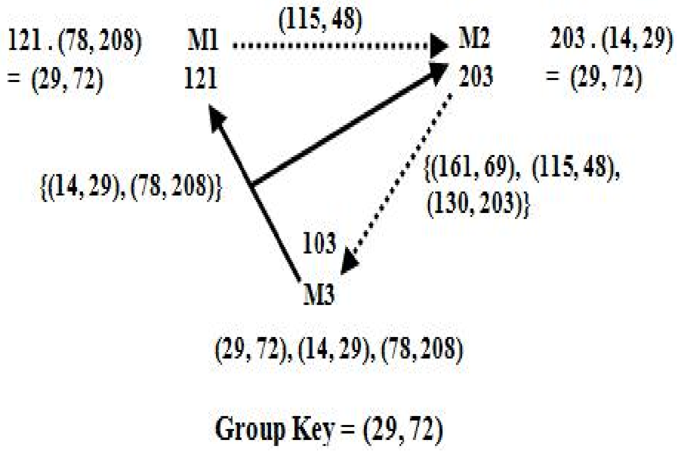

Figure 4: Computation of group key using GECDH.2 protocol

From Figures 3 and 4, it is inferred that the total number of point multiplications to obtain group key is 8, i.e., (a.G), (ab.G), (b.G), (abc.G), (ac.G), (bc.G), (a.(bc.G)), (b.(ac.G)) when the group size N=3. When the group has more than three members, then the proposed protocol divides the group into several sub groups each with 3 members. The total members in the last group may be 3 or less than 3. ITGECDH.2 protocol computes sub group key for all sub groups followed by the group key for entire group using calculated sub group keys.

For example, three separate sub group keys are calculated in each sub group as shown in Figure 5 when the group has 9 members, i.e., N=9. Let X_1, X_2, X_3 be sub group keys calculated using GECDH.2 and the encircled members 3,6 and 9 are sponsor of those sub groups. The group key for entire group is determined using calculated sub group keys by following the same procedure used in Figure 3. The sponsor of first sub group (member 3) broadcasts the blinded version of its own sub group key, i.e.,

$A = X_1.G$ to second sub group. The member 6 computes intermediate key using the blinded key obtained from the first sub group and it broadcasts these values along with public key of its own group to third sub group. Now, the members in third group calculate the final group key. Finally, member 9 computes two more intermediate keys and broadcasts them to other groups. The members in first two groups will compute the group key using those intermediate keys as exemplified in Figure 5.

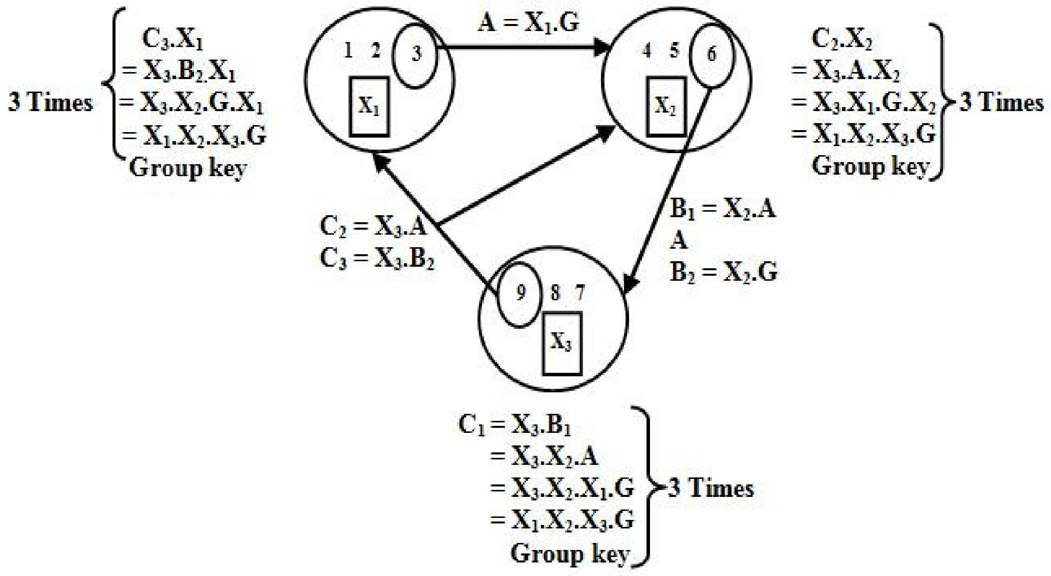
3 Times
C3.X1 = X3.B2.X1 = X3.X2.G.X1 = X1.X2.X3.G Group key
A = X1.G
1 2 3
X1
4 5 6
X2
C2.X2 = X3.A.X2 = X3.X1.G.X2 = X1.X2.X3.G Group key
3 Times
C2 = X3.A
C3 = X3.B2
B1 = X2.A
A
B2 = X2.G
9 8 7
X3
C1 = X3.B1 = X3.X2.A = X3.X2.X1.G = X1.X2.X3.G Group key
3 Times

Figure 5: Group key computation for N=9

From Figure 5, the total number of point multiplications to obtain group key when N=9 is determined as follows:
1. Number of point multiplications to compute three sub group keys is $3*8$
2. Number of blinded keys broadcast between the sub groups is $A = X_1.G$ *and* $B_2 = X_2.G : 2$
3. Number of intermediate keys calculated by the sponsors of second and third group is $B_1 = X_2.A, C_2 = X_3.A$, and $C_3 = X_3.B_2 : 3$
4. Number of final group keys calculated by three members in each sub group is $3*3 : 9$
Therefore, total number of point multiplications required to find the group key is sum of point multiplications used for computing A) Individual Sub group keys, B) Blinded keys, C) Indermediate keys and D) Final group key as written in coined equation (1).

$$Total\ point\ multiplications = A+B+C+D \tag{1}$$

In the above example, the total number of point multiplication is calculated as $3(8)+2+3+9 = 38$. Therefore, total number of point multiplications used to calculate the group key when N=9 is 38. Similarly, it can be calculated for the group with any members using previously calculated sub group keys. The above calculated value, 38 can be used for determining the total point multiplications required for establishing the group key in a group of 27 members, i.e., $3(38)+2+3+27 = 146$.

Table 1 summarizes the above process of calculating total number of point multiplications required for computing the group key for the group size upto 3^5.

Table 1: Total number of point multiplications

S.No	**N**	**Point multiplications**				**Total**
		A	**B**	**C**	**D**	
1	3	0	2	3	3	8
2	9	3(8)	2	3	3(3)	38
3	27	3(38)	2	3	3(9)	146
4	81	3(146)	2	3	3(27)	524
5	243	3(524)	2	3	3(81)	1820

It is inferred from Table 1 that total number of point multiplications for establishing the current group key can be determined from total point multiplications required for finding the keys of its previous sub groups. Let 'TPM' be the Total Point Multiplications and the proposed equation for calculating it is given in equation (2).

$$TPM(N) = 3 * TPM\left(\frac{N}{3}\right) + N + 5 \tag{2}$$

The equation (2) can be used only when TPM for previous sub groups is known. Otherwise, this can be determined from the height of key tree which is used in organizing the group members as shown in Table 2, where $h = log_3 N$.

Table 2: TPM from height of the key tree

S.No	**N**	**h**	**TPM**
1	3	1	8
2	9	2	38
3	27	3	146
4	81	4	524
5	243	5	1820

A new equation is proposed in equation (3) which determines the value for TPM from the group size and height of the key tree.

$$TPM(N) = \frac{5(N-1)}{2} + hN \tag{3}$$

The proposed ITGECDH.2 protocol is compared with other existing tree based GKA methods with respect to total number of operations and is given in Table 3. The details of base protocol, key tree and base operation employed in the existing GKA methods are listed. The comparison between existing GKA methods and proposed ITGECDH.2 protocol based on the total number of operations is shown in Figure 6.

The difference between the number of modular exponentiations between EGK [26] and DHBGKA [27] are very less when the group size is small. However, this difference is increased as group size increases and DHBGKA performs less number of modular exponentiations than EGK when the group has more members. The DHBGKA protocol has more number of operations than TGDH for the group size up to 3^4 and the number of modular exponentiations in DHBGKA is reduced when the group size is greater than 3^4. Initially, when the group size is small, the total number of operations in proposed protocol ITGECDH.2 is close to TGDH protocol. But, when the group size is increased, the number of operations in ITGECDH.2 is reduced when compared to TGDH protocol.

Table 3: Comparison based on number of operations

S. No	**Existing method**	**Base protocol**	**Key tree**	**Base operation**	**Total no. of operations**
1	EGK	DH	Binary key tree	Modular exponentiation	$2N(h+1)$
2	TGDH	DH	Binary key tree	Modular exponentiation	$2(N-1)+Nh$
3	DHBGKA	DH	Binary search tree	Modular exponentiation	$9(2^h-1)$
4	DSGSF	DH	Binary key tree	Modular exponentiation	$2(N-1)+2^{h-1}+Nh$
5	ITGECDH.2	GDH.2 + ECDH	Ternary key tree	Point multiplication	$\frac{5(N-1)}{2}+Nh$

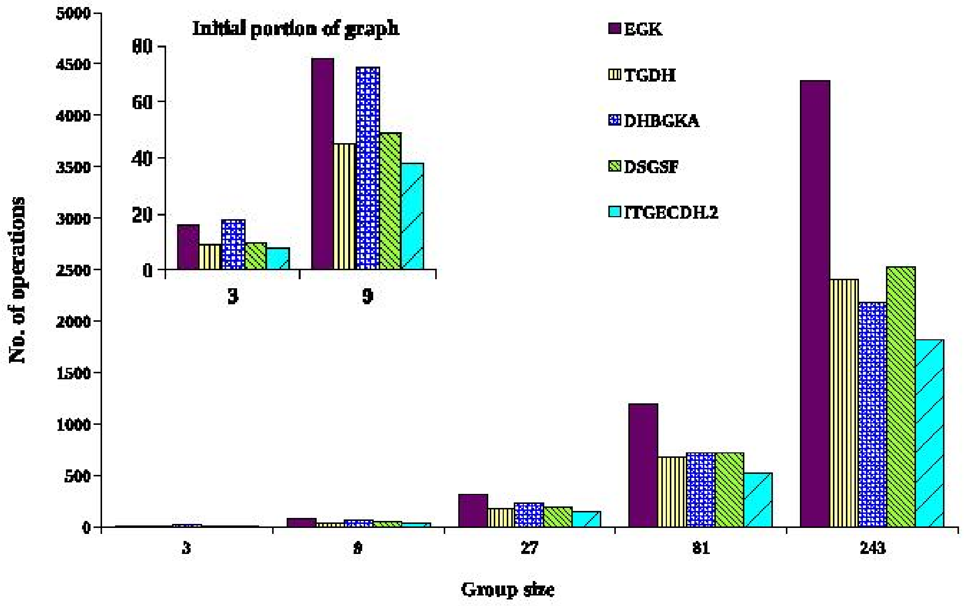

Figure 6: Group size Vs Total number of operations

The proposed protocol is also compared with DSGSF and the number of operations in DSGSF is more when compared to proposed ITGECDH.2 protocol. The reason is, Group Leader (GL) initially generates both secret and public keys for all members in binary tree. Then it computes group key by adopting the scheme used in TGDH and it encrypts secret as well as public keys. GL unicasts the signed copy of these information to each member in binary tree and it uploads the structure of binary tree into the cloud with its related information.

Each member will get public keys of sibling nodes of nodes in its keypath by sending the request to cloud server. Then, each member will compute both secret and public keys of node in its keypath until it computes keys for root node. Thus, group key is computed twice both by GL and all members in the group which increases the number of modular exponentiation. Further, the computation of initial group key has some hidden cost due to encryption and signature operations.

It is clearly seen from Figure 6 that the proposed protocol ITGECDH.2 has reduced number of operations when compared to EGK, TGDH, DHBGKA and DSGSF. The reason is that, a ternary key tree is used in ITGECDH.2 to organize the group members and GDH.2 is employed as a base protocol for finding a group key.

5 Performance analysis

In this section, ITGECDH.2 protocol is compared with TGECDH.2 using total rekeying time and renewed nodes in the ternary key tree.

The time taken to recompute the new group key during BR operation is called rekeying time and it represents the computation cost. The total number of internal nodes whose keys are updated during BR are called renewed nodes [9] which are used for measuring the communication cost. A group with 729 members was considered in the experiment. The values of J and L (join and leave members) were varied from 0 to 243 and the corresponding metric values were measured.

In Figures 7 and 8, X-axis represents the total join members (J), Y-axis indicates the total members who leave (L) the group. The Z-axis in Figure 7 shows the time to establish a new group key, whereas it specifies the renewed nodes in Figure 8.

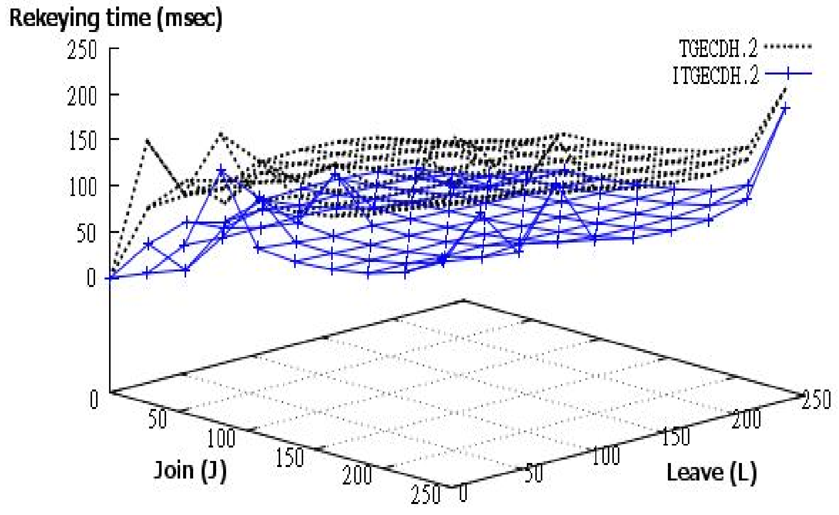

Figure 7: No. of join & leave members
Vs Rekeying time

From Figure 7, it is apparent that, ITGECDH.2 protocol takes less time to recompute the group key. The difference in the performance of these two protocols is significant when the values of both join and leave are high. Because, unlike TGECDH.2, ITGECDH.2 protocol prunes the nodes associated with leave members which reduces the height of key tree. Hence, the number of operations to be performed for establishing the new group key is also reduced.

Figure 8 depicts the performance of these two protocols with respect to number of renewed nodes. ITGECDH.2 has produced lesser number of renewed nodes than TGECDH.2 during frequent join and frequent leave. If both join and leave operations happen frequently in the group, then the renewed nodes count is high in TGECDH.2 as this protocol replaces the leave members with join members instead of pruning them.

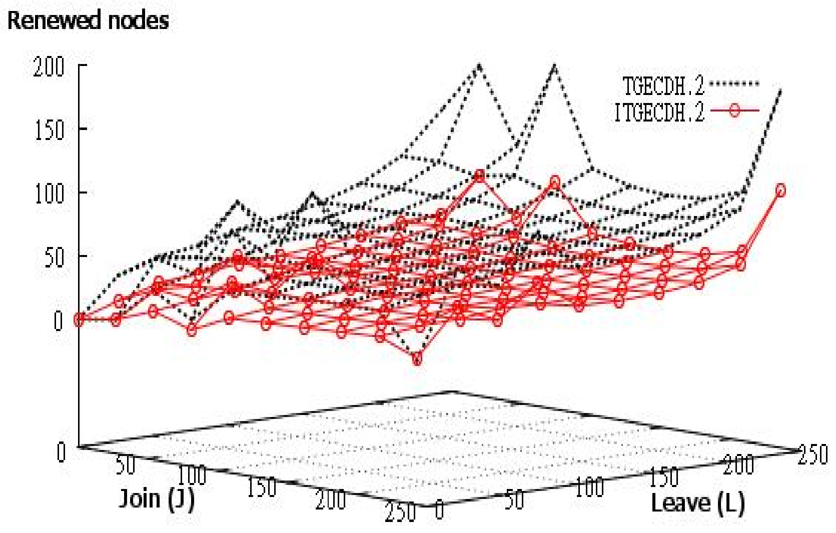

Figure 8: No. of join & leave members
Vs Renewed nodes

It is inferred from the above two graphs that the proposed protocol ITGECDH.2 performs better than TGECDH.2 as it always chooses the appropriate IP in Case 1 at which the height of current key tree will not increase. Thus, ITGECDH.2 protocol always keeps the balanced tree to minimize total number of operations to be performed during BR operation.

The performance difference between these two protocols is very less for the initial values of join and leave requests. But, this is more significant when the group has highly dynamic join and leave members as $Improved\_TBP()$ algorithm creates new key tree by pre-processing all the join requests in highly dynamic group. Also, the height of MKT is reduced by pruning the nodes associated with leave members during leave operations.

Thus, $Improved\_TBP()$ algorithm reduces total rekeying time as well as renewed nodes generated in the ternary key tree. ECRMANET has smaller and portable wireless devices with limited resources such as battery power, memory, computing capability of CPU, etc. Hence, the proposed ITGECDH.2 protocol is well suited for emergency communications among highly dynamic group members in ECRMANETs.

6 Conclusion

The cognitive radio based MANET is well suited for emergency networks as it can deploy a quick communication by solving the spectrum scarcity problem. This paper has proposed a tree based group key agreement with improved batch rekeying for securing the communication among members in the rescue team. The proposed ITGECDH.2 protocol performs $\frac{5(N-1)}{2} + Nh$ number of operations to compute the initial group key which is less when compared to other key agreement protocols. Further, it reduces rekeying time and number of renewed nodes than the existing TGECDH.2 protocol. Hence, this can be adopted for providing secure communication in group oriented emergency cognitive radio MANETs.

References

[1] Q. Zhang, F. W. Hoeksema, A. B. Kokkeler, and G. J. Smit, *Towards Cognitive Radio for emergency networks*. Nova Publishers, 2006.

[2] P. Pawełczak, R. V. Prasad, L. Xia, and I. G. Niemegeers, "Cognitive radio emergency networks-requirements and design," in *Proc of the 1st IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN 2005), Baltimore, Maryland, USA*. IEEE, November 2005, pp. 601–606.

[3] J. Mitola III and G. Q. Maguire Jr, "Cognitive radio: making software radios more personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, August 1999.

[4] D. Grandblaise, D. Bourse, K. Moessner, and P. Leaves, "Dynamic spectrum allocation (dsa) and reconfigurability," in *Proc. of the 2002 Software-Defined Radio Forum (SDR'02), San Diego, California, USA*, November 2002.

[5] S. Ghafoor, P. Sutton, C. Sreenan, and K. Brown, "Cognitive radio for disaster response networks: survey, potential, and challenges," *IEEE Wireless Communications*, vol. 21, no. 5, pp. 70–80, October 2014.

[6] N. Uchida, K. Takahata, Y. Shibata, and N. Shiratori, "Never die network based on cognitive wireless network and satellite system for large scale disaster," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 3, no. 3, pp. 74–93, September 2012.

[7] Y.-N. Lien, H.-C. Jang, and T.-C. Tsai, "A manet based emergency communication and information system for catastrophic natural disasters," in *Proc. of the 29th IEEE International Conference on Distributed Computing Systems Workshops (ICDCS'09),Montreal, Quebec, Canada*. IEEE, June 2009, pp. 412–417.

[8] A. Gorcin and H. Arslan, "Public safety and emergency case communications: Opportunities from the aspect of cognitive radio," in *Proc. of the 3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN'08),Chicago, Illinois, USA*. IEEE, October 2008, pp. 1–10.

[9] P. P. Lee, J. Lui, and D. K. Yau, "Distributed collaborative key agreement and authentication protocols for dynamic peer groups," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 263–276, April 2006.

[10] Y. Kim, A. Perrig, and G. Tsudik, "Simple and fault-tolerant key agreement for dynamic collaborative groups," in *Proc. of the 7th ACM conference on Computer and communications security (CCS'00), Athens, Greece*. ACM, November 2000, pp. 235–244.

[11] N. Renugadevi and C. Mala, "Ternary tree based group key agreement for cognitive radio manets," *International Journal of Computer Network and Information Security (IJCNIS)*, vol. 6, no. 10, pp. 24–31, September 2014.

[12] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-hellman key distribution extended to group communication," in *Proc. of the 3rd ACM conference on Computer and communications security (CCS'96), New Delhi, India*. ACM, March 1996, pp. 31–37.

[13] L. Harn and C. Lin, "Efficient group diffie–hellman key agreement protocols," *Computers & Electrical Engineering*, vol. 40, no. 6, pp. 1972–1980, August 2014.

[14] V. Odelu, A. K. Das, and A. Goswami, "An efficient ecc-based privacy-preserving client authentication protocol with key agreement using smart card," *Journal of Information Security and Applications*, vol. 21, pp. 1–19, April 2015.

[15] H.-M. Sun, B.-Z. He, C.-M. Chen, T.-Y. Wu, C.-H. Lin, and H. Wang, "A provable authenticated group key agreement protocol for mobile environment," *Information Sciences*, vol. 321, pp. 224–237, November 2015.

[16] S. Tang, L. Xu, N. Liu, X. Huang, J. Ding, and Z. Yang, "Provably secure group key management approach based upon hyper-sphere," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3253–3263, January 2014.

[17] S.-J. Jung, J.-H. Lee, and T.-M. Chung, "The effective group key agreement protocol for ad-hoc networks for medical emergency environments," in *Proc. of the 2006 SICE-ICASE International Joint Conference, Busan, Korea*. IEEE, October 2006, pp. 1127–1130.

[18] M. Steiner, M. Waidner, and G. Tsudik, "Cliques: A new approach to group key agreement," in *Proc. of the 18th IEEE International Conference on Distributed Computing Systems (ICDCS'98), Amsterdam,The Netherlands*. IEEE, May 1998, pp. 380–387.

[19] X. Chen, B. N. Ma, and C. Yang, "M-cliques: Modified cliques key agreement for secure multicast," *Computers & Security*, vol. 26, no. 3, pp. 238–245, May 2007.

[20] D.-C. Lou, T.-F. Lee, and T.-H. Lin, "Efficient biometric authenticated key agreements based on extended chaotic maps for telecare medicine information systems," *Journal of medical systems*, vol. 39, no. 58, pp. 1–10, May 2015.

[21] S. S. Kulkarni and B. Bruhadeshwar, "Key-update distribution in secure group communication," *Computer Communications*, vol. 33, no. 6, pp. 689–705, April 2010.

[22] K. Xue and P. Hong, "A dynamic secure group sharing framework in public cloud computing," *IEEE Transactions on Cloud Computing*, vol. 2, no. 4, pp. 459–470, October 2014.

[23] X. Zhao, F. Zhang, and H. Tian, "Dynamic asymmetric group key agreement for ad hoc networks," *Ad Hoc Networks*, vol. 9, no. 5, pp. 928–939, July 2011.

[24] M. Li, Z. Feng, N. Zang, R. L. Graham, and F. F. Yao, "Approximately optimal trees for group key management with batch updates," *Theoretical Computer Science*, vol. 410, no. 11, pp. 1013–1021, March 2009.

[25] L. Liao and M. Manulis, "Tree-based group key agreement framework for mobile ad-hoc networks," *Future Generation Computer Systems*, vol. 23, no. 6, pp. 787–803, July 2007.

[26] J. Alves-Foss, "An efficient secure authenticated group key exchange algorithm for large and dynamic groups," in *Proc. of the 23rd National Information Systems Security Conference (NISSC'00), Baltimore, Maryland, USA*, October 2000, pp. 254–266.

[27] H.-Y. Lin, T.-C. Chiang, C.-Y. Yang, and J.-L. Chang, "Dynamic multicast height balance key agreements for secure multicast communication in ad hoc networks," in *Proc. of the IEEE International Conference on Information Reuse and Integration (IRI'10),Las Vegas, Nevada, USA*. IEEE, August 2010, pp. 55–58.

Author Biography

N. Renugadevi is currently pursuing Ph.D. in full time at the Department of Computer Science and Engineering, National Institute of Technology, Tiruchirapalli, Tamilnadu, India. Her research interests include secure group communication and group key agreement in Cognitive radio networks. She is a life member of Indian Society for Technical Education (ISTE).

C. Mala is currently serving as an Associate Professor in the Department of Computer Science and Engineering, National Institute of Technology, Trichy, Tamilnadu, India. She received Ph.D. from National Institute of Technology, Trichy in 2008. Her research interests include Wireless Networks, Parallel Algorithms, Network Security, Soft Computing and Image processing.