International Journal of Safety and Security Engineering
Vol. 13, No. 6, December, 2023, pp. 1077-1082
Journal homepage: http://iieta.org/journals/ijsse

Simple Hash Based Symmetric Encryption Mechanism for Dynamic Groups

Redamalla Rekha[1], Medi Sandhya Rani[2*], Kura Shailaja[3], Nagaraju Krishna Chythanya[4]

[1] Department of Computers and Informatics, University College of Engineering & Technology, MGU, Nalgonda 508254, India
[2] Department of Information Technology, Bhoj Reddy Engineering College for Women, JNTUH, Hyderabad 500059, India
[3] Department Computer Science and Engineering, Vasavi College of Engineering, Hyderabad 500031, India
[4] Department of Computer Science and Engineering, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad, Telangana 500090, India

Corresponding Author Email: sandhyarani.medi@slv-edu.in

https://doi.org/10.18280/ijsse.130610

Received: 1 September 2023
Revised: 25 October 2023
Accepted: 8 November 2023
Available online: 25 December 2023

Keywords:
decode, encipher, group controller, group encryption, join, leave, security

ABSTRACT

Security in Dynamic Groups is an open research problem in Mobile Adhoc Networks (MANETs) as users join/leave a group with insecure open wireless networks. To avoid information leakage in multicast communication during group dynamics, forward secrecy, and backward secrecy primitives to be ensured. In this context, a proper and secure group key is to be derived and updated whenever users join or leave the group. An efficient, simple, and secure symmetric multicast group encryption mechanism for group-oriented communications in MANETs is proposed and implemented with the derived group key. In this paper, the encryption mechanism for the "Multicast Symmetric Secret Key Management Scheme (MC-SSKMS)" is illustrated. This protocol provides O(1) computational overhead which is less than the other existing approaches and the results for both join and leave events are provided.

1. INTRODUCTION

Mobile Ad-hoc Networks are an unstructured collection of Mobile Hosts (MHs) that can also act as routers and are linked by wireless networks to form a communication network. Physical security risks like snooping, phishing, and denial-of-service attacks are common on mobile wireless networks. Security is the most important concern in MANETs to prevent attacks, especially in multicast applications where a single user transmits a message to a group of users. To provide security in such an environment, a secure group key must be derived which is used to encrypt the message by the sender and only intended group members must decrypt that message. The same group key is used by both sender and recipients for the transmission of messages, which is called the Symmetric Group Encryption mechanism. Group Encryption key can be calculated and disseminated based on the specific multicast key agreement approach.

The categories of Group communication Key Agreement protocols are as follows: (i). Group Key Agreement protocols with centralized control, (ii) Decentralized Group Key Agreement protocols and (iii) Distributed Group Key Agreement protocols. Distributed key management methods tend to have little computational complexity during rekeying than centralized and decentralized group key agreement protocols because in distributed protocols each user individually calculates the rekey [1]. In Centralized and Decentralized methods there is a problem of a single point of failure which is not there in Distributed key agreement mechanisms. Due to less computational overhead and flexibility in key calculation, we developed a distributive group key management scheme for secure group communication in dynamic groups. In this distributive environment, each user contributes a secret share to prepare a shared key which is used for both encryption and decryption. To compute a shared secret group key in startup phase, a secure hash technique known as MD5 is used. During updation phase of secret key calculation, a rekey needs to be computed. Rekeying is the process of creating a new shared Secret Group Key during a user join or quits a multicast group. We designed and developed a symmetric distributive Group Key Encryption protocol for MANETs in which a common secret group key is safely disseminated to each user in the group. With that Group Key the sender encrypts the message and the remaining group members decipher the encrypted message to retrieve the original information.

The remaining part of the paper is organized as follows: Section 2 describes related work, Section 3 presents the methodology of the proposed protocol, Section 4 demonstrates the Results and performance analysis of proposed protocol and Section 5 presents the conclusion.

2. RELATED WORK

Many studies have been done on various Group key management strategies. An effective and secure protocol for group communication in MANETS was devised after a survey of the literature on multicast communication methods. A centralized method based on a tree structure, the Logical Key Hierarchy (LKH), was created by Wallner et al. [2], provides linear initial keying performance and improved logarithmic rekey performance. Key storage requirements at each user is d+1 keys while the group controller must store all Key Encryption Keys (KEKs) and the Group Traffic Encryption Keys (GTEKs).

Like LKH, One Way Function (OFT) proposed by Sherman, and McGrew [3] is a centralized technique for group key management mechanism which reveals how each key in the OFT scheme functions about the others. The number of keys stored by group members, the number of keys broadcast to the group when new members are added or evicted, and the computational efforts of group members, are logarithmic in the number of group members. Among the hierarchical methods, OFT is the first to achieve an approximate halving in broadcast length, an idea on which subsequent algorithms have built. Splitting a large group into smaller subgroups is a decentralized GKM method in which group coordinator is selected for each subgroup that was demonstrated in IOLUS [4]. Protocols based on Iolus can be used to achieve a variety of security objectives and may be used either to directly secure multicast communications or to provide a separate group key management service to other "security-aware" applications.

Tree Based Group Diffie Hellman (TGDH), a technique for Decentralized group key management that combines the effectiveness of the tree structure with the contributing characteristic of DH, was created by the authors of Kim et al. [5] in which key distribution to the local members of each subgroup is handled by a subgroup controller. N-party Diffie-Hellman key agreement was suggested for group communication by Amir et al. [6], Burmester and Desmedt [7]. In these schemes, there is no need for a group controller whose cost is more than the other participants. And furthermore, users have a public key and authenticate their messages using an appropriate authentication scheme.

Huang and Mishra [8] demonstrated a fully distributed scheme that has the advantage of configuration flexibility. However, it lacks any trusted security anchor in the trust structure. Many certificates need to be generated. Every node should collect and maintain an up-to-date certificate repository. Certificate chaining is used for authentication of public keys. The certificate graph, which is used to model this web of trust relationship, may not be strongly connected, especially in the mobile ad hoc scenario. In that case nodes within one component may not be able to communicate with nodes in different components. A variety of Distributed group key management systems have been proposed for SGC in wireless networks [9, 10]. SEKM is based on the secret sharing scheme, where the system secret is distributed to a group of server nodes [9]. The server group creates a view of a CA. The advantage of SEKM is that it is easier for a node to request service from a well-maintained group rather than from multiple ‘‘independent’’ service providers which may be spread in a large area.

A cluster-based technique was proposed by Renuka and Shet [11], is hierarchical and fully distributed with no central authority and uses a simple rekeying procedure which is suitable for large and high mobility mobile ad hoc networks. The rekeying procedure requires only one round in this scheme. Authors reduced the energy consumption during communication of the keying materials by reducing the number of bits in the rekeying message. They demonstrated through analysis and simulations that the developed scheme has less computation, communication and energy consumption compared to the DLKH and GDH schemes. But the limitation is uncasing the rekeying messages which lead to a greater number of messages in the network. And this scheme does not provide forward and backward secrecy.

The distributed group key is computed in the Simple and Efficient Group Key (SEGK) management strategy for MANETs is presented by Wu et al. [12], where each node stores two logical trees in its local memory and group maintenance role is transmitted from one member to other. The advantage of this work is sharing the load and maintaining forward and backward secrecy. To update the key $O(n^2)$ exp computational complexity required during join and leave events. Double multicast tree maintenance and high computational complexity are the limitations in this work.

Another distributive GKM system that permits dynamic group membership was introduced by Kang et al. [13] by Sukin Kang et al and demonstrated group key secrecy, backward and forward secrecy, key independence, and implicit key authentication under the decisional Diffie-Hellman (DDH) assumption. In this work group dynamics are also considered for calculating Group key. To update the key, each node has to perform only 1 modular exp operation i.e., O(1) exp. But the limitation of this work is Initial phase consists of two rounds for computation of Group key and Group master has additional computational complexity.

Vaishnavi and Upadhay [14] demonstrated Symmetric management schemes, Asymmetric management schemes, Group key management schemes and Hybrid management schemes. Each scheme is explained with one or two protocols with the merits and demerits in it. Vanathy and Ramakrishnan [15] proposed Escrow based Elliptic Curve Cryptographic group key management scheme in MANETs. It is a public key based cryptographic approach where efficient encryption mechanism is used. In this work two secure group key schemes are proposed: one is within the subgroup another is external to the group. Authors analyzed the Quality-of-Service metrics like storage cost, throughput and communication overhead with simulation environment and demonstrated that ECC-based asymmetric cryptographic group communication outperformed previous Group key mechanisms in terms of metrics. But the drawback of this work is group dynamics are not considered which is more important in MANETs.

Sandhya Rani et al. [16] proposed symmetric group key management mechanism and measures for service quality such as Key Delivery Ratio, Battery consumption and End-to-end Delay are assessed. These metrics use symmetric GKM to offer both forward and backward security. NS2 tool is used for experimental set up in which the above metric values are recorded. The authors compared these values with two contemporary methods and proved that their scheme possesses better Quality of Service than other methods. But they ignored computational overhead of group key calculation and key updation.

In summary, there are many researchers who developed and analyzed different key management schemes as key management is the base for Encryption especially in multicast nature of MANETS. In our work we addressed the limitations of above literature by broadcasting the rekeying messages to group members, using simple and secure hash-based algorithm to provide less computational complexity in initial phase as well as key updation phase by maintaining forward secrecy and backward secrecy.

3. METHODOLOGY OF THE PROPOSED PROTOCOL

In a symmetric group key agreement, each group member has access to a shared key that is used to encipher the data and the same key is used to decipher the data by all the members in the group [13]. If key management is centralised, TTP may calculate this key. In Distributed key management, each user provides their share value to compute the group key. The following two security primitives are ensured in multicast communication especially in highly dynamic MANET environments:

- Forward secrecy: The member who evicts from the group cannot access the further messages with old key.
- Backward secrecy: A new user cannot decrypt or access the past messages with newly calculated key.

Secure Hash Function, a mathematical operation performed at the sender's side and added to the original message, is the foundation upon which group keys are calculated. The receiver performs the same calculation and compares the tag value to confirm authenticity. Secure hash function generates Message Digest without the need for a key. In our proposed work, we employed the one-way secure hash function MD5 to compute the Group key.

The MD5 method generates a 128-bit message digest from an input of any length. It has a hexadecimal representation of 8 digits. Blocks of 512 bits from the input are processed. To break the message digest it takes 2^{128} operations, which is complicated in this work because it requires knowing the hash value of n users. The MD5 algorithm's benefit over other secure hashing algorithms like SHA-256, SHA-512 is its fast-processing speed. On the other side of the coin SHA-3, SHA-256, SHA-512 are more secure than MD5 because of its output i.e. double the length of MD5. However, in our protocol we used hash algorithm at individual users and also at Group Leader which leads to n+1 hash operations. This requirement leads to high computation with SHA algorithms. To compensate the computation time with sufficient security we employed MD5 for calculation of group key.

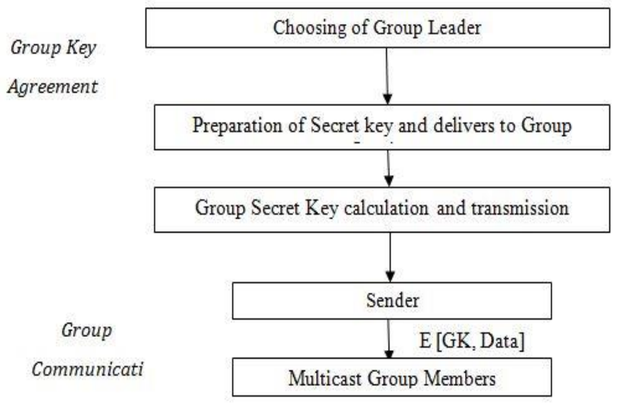

Figure 1. Process flow of simple and hash based symmetric group key encryption

To generate a Group key, each user gives a share that is a hash of the random integer in our distributed key management protocol. The Group Leader/Group Controller creates a common Group Key by using the safe MD5 hash technique on shared values that have been received. This is done during initialization, and the created Group key needs to be sent separately through a secure unicast message to each user. Any team member can encode a message with the help of the shared Group.

Only group members can decipher the information using the key that has been sent to the group. Additionally, our work offers backward and forward secrecy for user join and departure events. In other words, a new user cannot decode older communications using a newly calculated key, and a person who leaves the group cannot access later messages using a previously calculated key.

We extended and implemented a simple Hash based symmetric encryption mechanism based on design (Figure 1) and development of MC-SSKMS (Multicast Symmetric Secret Key Management System). Every node in this protocol contributes its own part to compute the Group key that is utilized for both encoding and decoding.

Each group member executes the Group Key Calculation and Distribution Algorithm. It consists of two stages:

A) Group encryption key calculation and distribution algorithm

Step 1: The first step is for nodes (M_i) that want to participate in multicast communication have to broadcast their ID and a "hello" message. The one with the least ID in the group is the group leader. To recognize its members, the group head broadcasts the phrase "I am Group head."

$$M_i \xrightarrow{\text{Id}} G_n: (i \in [1, n])$$

Step 2: Using the MD5 algorithm, each member of the group M_i, generates a random number R_i and computes its hash HR_i and transmits the computed hash value to GL.

$$M_i \xrightarrow{(HR_i, Id_i)} GL: (i \in [1, n], i \neq GL)$$

Step 3: The group key is then generated by the group administrator using the one-way hash function.

$$GEK = f\,(HR_1, HR_2, HR_3, \ldots, HR_n)$$

where, HR_i is a Hash of the random value and f is the MD5 one-way hash function.

Step 4: GL sends this group key to all participants in the multicast connection using Key Encryption Key (KEK).

B) Group encryption key updation

Step 1: If updation = "**Join event**", each joined member chooses a share $R_{j,}$, calculates hash of the chosen value and sends to GL.

$$M_j \xrightarrow{(HRj, IDj)} GL: (j \in [n+1, n+m])$$

Step 2: GL calculates Group key by applying MD5 one way hash function.

$$GK = f\,(GK, HR_{n+j}): (j \in [1, m])$$

Step 3: GL distributes refreshed Group key to all users in the group.

$$GL \xrightarrow{E\,[KEK,\,GK]} G_n$$

Step 4: Else//Assume that m members are leaving from the group G_n.

GL generates the group key by applying MD5 one-way hash function on existing members.

$$GK' = f(HR_i) : (i \in [1, n-m]$$

Step 5: GL distributes refreshed group key to all users in the group.

$$GL \xrightarrow{E\,[KEK,\,GK']} G_n$$

3.1 Multicast group communication (encryption and decryption)

Each user acquires GEK during set up phase or updates the key during updation phase. Whenever a user transmits a multicast data in the group, it encrypts the data and sends in a multicast secure transmission environment. In this proposed protocol Advanced Encryption Standard algorithm is used for enciphering the data while maintaining group secrecy. Only the group members who know GEK can decrypt the data by that forward secrecy and backward secrecy is maintained.

Advanced Encryption Standard (AES) is a symmetric cryptographic algorithm employs a block size of 128 bits and input of any arbitrary length. The computational overhead of AES based on key size. For instance, the number of rounds is 10 for a key size of 128, number of rounds is 12 for a key size of 192, and 14 for a key size of 256 bits. In AES although longer keys are harder to crack, the problem is computation time. But compared to exponentiations in asymmetric cryptographic approaches AES leads to less computational overhead.

3.2 Illustration of the proposed method

A) Group encryption key calculation

1) Suppose the group has 4 users.
2) Each one of them select a random number between 1 and 1000, which is then hashed using MD5 secure hash algorithm.

   - R1=165 and H(R1): 9766527f2b5d3e95d4a733fcfb77bd7e
   - R2=72 and H(R2): 32bb90e8976aab5298d5da10fe66f21d
   - R3=431 and H(R3): 66368270ffd51418ec58bd793f2d9b1b
   - R4=674 and H(R4): 0d7de1aca9299fe63f3e0041f02638a3

3) The above hash values of 4 random numbers are concatenated and the result as follows:

9766527f2b5d3e95d4a733fcfb77bd7e32bb90e8976aab5298d5da10fe66f21d

66368270ffd51418ec58bd793f2d9b1b0d7de1aca9299fe63f3e0041f02638a3

4) Group Encryption Key (GEK)=64266a3af1856b2d3b6e00638113bf4e (By applying MD5 on step3 result).

B) Group encryption key updation

For Join Event:

1) Suppose 3 members joined the group.
2) Each one of them selects a random number between 1 and 1000 which is then hashed using MD5 secure hash algorithm.

   - R6=854 and H(R6): 2303bee891431336538b2b4c0bb756db
   - R7=357 and H(R7): 24f281b4d6a9688a3cc37292f7c75e90
   - R8=625 and H(R3): 50e06e8355e9ba35fd49fc08de8b0347

3) Concatenation of the hashes is Res = 2303bee891431336538b2b4c0bb756db24f281b4d6a9688a3cc37292f7c75e9050e06e8355e9ba35fd49fc08de8b034.
4) Updated Group Encryption Key (UGEK) = MD5(GEK,Res). UGEK=51634b65cf970f372684d4bb518aa774.

These 3 members can not decrypt the past messages with this UGEK thus achieved backward secrecy. In the same way if any user leaves, from this group a new key is calculated and with the old key the evicted user can not access further messages of the group.

C) Multicast group communication

Encryption: Suppose user2 wants to transmit a message in the group, it performs encryption using GEK.

If GEK = 64266a3af1856b2d3b6e00638113bf4e (which is calculated in Group Encryption Calculation stage in (**B**) and sample original message in Hexadecimal is 6A2C000000000000.

Then the generated cipher text using Advanced Encryption Standard (AES) algorithm is 42BDDD2DBE787C9174C6604D34655449274FDF6D5A8786797E349A733F8BC7EB.

User2 transmits this cipher text in the group.

Decryption: When all group members in the group receive the above cipher text, each user begins the decryption procedure as follows:

Each user applies AES decryption algorithm with GEK=64266a3af1856b2d3b6e00638113bf4e and cipher text 42BDDD2DBE787C9174C6604D34655449274FDF6D5A8786797E349A733F8BC7EB, and original plain text 6A2C000000000000 is retrieved by all the members.

By the above illustration, it is understood that only group members who knows GEK can decrypt the message. Hence it is proved that our protocol provides Forward and Backward secrecy

4. RESULTS AND PERFORMANCE ANALYSIS

Our suggested approach is implemented in Java with varied user counts (Intel Core i5 processor, 8GB RAM, Windows 10

Operating System). We estimated the computation time of Group Encryption Key calculation in this environment. Group Encryption Key calculation computational complexity is compared to that of existing key management techniques such as Simple and Efficient Group Key Management (SEGK) and Collaborative Diffie Hellman (CODH) techniques.

In contrast to the two mentioned protocols SEGK [12] and CODH [13], we used the MD5 secure hash technique in our proposed work to calculate the Group Key. The computational complexity of SEGK is O(n^2), whereas the computational complexity of CODH is O(n) exponentiations to calculate Group Encryption Key. Our suggested approach requires O(1) computational overhead to calculate the group key during initialization than the other two protocols since it employs a one-way hash function, as illustrated in Table 1 and Figure 2.

Table 1. Group Key computation time with varying Number of nodes

Group Key Computation Time (sec)					
	Number of Nodes				
Protocols	20	40	60	80	100
SEGK	4.27	6.32	7.98	9.63	11.45
CODH	3.89	5.46	6.82	8.12	9.68
MC-SSKMS(Proposed)	3.13	3.98	4.28	5.72	5.92

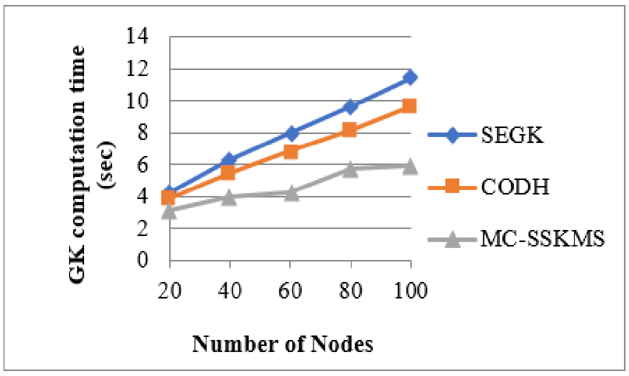

Figure 2. Group Encryption Key calculation time

Table 2. Rekey computation time for join event with a variable number of nodes

Rekey Computation Time (sec) for Join Event When p=10%					
	Number of Nodes				
Protocols	20	40	60	80	100
SEGK	1.25	1.75	2.25	2.80	3.41
CODH	1.60	3.20	4.85	6.50	7.70
MC-SSKMS (Proposed)	0.85	0.89	1.21	1.56	1.62

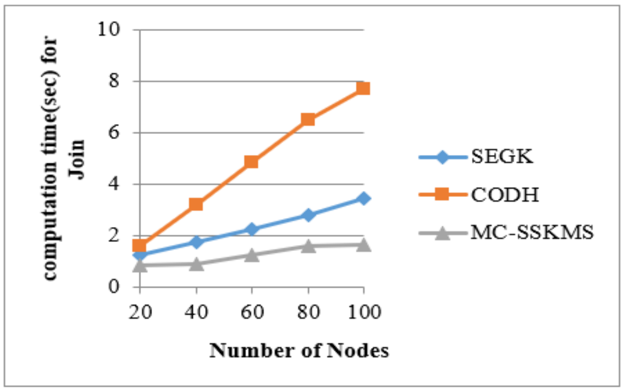

Figure 3. Key updation time (Join)

SEGK and CODH computation time values for Join events are taken with membership changes (p=10%) and compared with our protocol. Table 2 and Figure 3 show that CODH requires more time to compute the rekey for the Join event than SEGK does, as in SEGK, only new members compute the group key, while existent members refresh the group key with the blinded key of the new members. In contrast, CODH prepares a fresh lock list for each member after a new member enters the group, adding to the overhead. By executing a one-way hash function of the old Group key with that of the new Member's concatenated hash value, our MC-SSKMS lowers the join rekey cost.

The calculation time of our protocol compared to SEGK and CODH for the leave event where membership changes (p=10%) are shown in Table 3 and Figure 4. SEGK and CODH require more time for rekey computation for Leave events than our protocol due to their use of modular exponentiations in rekey computation that are like group key calculations performed at the initial phase. Our protocol computes the fresh Group key using a one-way hash function reduces the time required to rekey.

Table 3. Rekey computation time for leave event with a variable number of nodes

Rekey Computation Time (sec) for Leave Event When p=10%					
	Number of Nodes				
Protocols	20	40	60	80	100
SEGK	3.23	4.92	5.89	7.89	9.29
CODH	2.88	4.12	5.24	7.23	8.72
MC-SSKMS (Proposed)	1.82	2.56	3.28	3.57	3.36

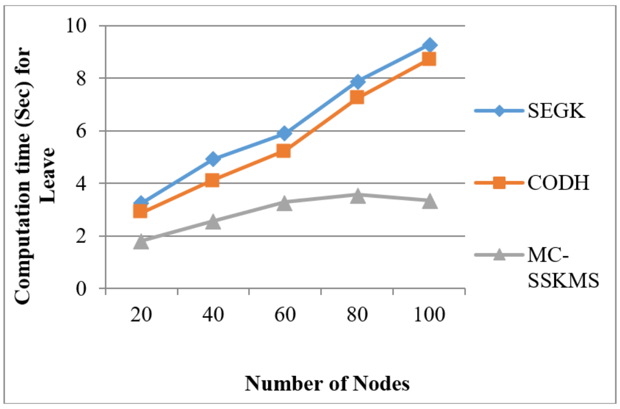

Figure 4. Key updation time (Leave)

5. CONCLUSIONS

Secure and effective key management is crucial for group communication in MANETs because they support a greater range of secure multicast applications like video conferencing, military applications etc. In our proposed work, we created a secure and effective symmetric Group Key Management technique by reducing the computation complexity of the rekey time when membership changes. Each user of this protocol contributes to the calculation of the group key in a distributive environment. Here, the group key was determined using a one-way hash function during the initialization phase and key is updated during join and leave events. In this work, we used MD5 secure hash algorithm to calculate the group key which leads to less computational burden. Due to maintenance of double multicast tree in SEGK and DH calculations in CODH leads to more computations in SEGK and CODH protocols respectively. The findings in the aforementioned section make it clear that our protocol has less computational complexity than the SEGK and CODH protocols. And it also proved that our protocol ensures backward and forward secrecy. We can extend our work to simultaneous join and leave events in calculating rekey computation time.

REFERENCES

[1] Bouassida, M.S., Chrisment, I., Festor, O. (2008). Group key management in MANETs. International Journal of Network Security, 6(1): 67-79.

[2] Wallner, D.M., Harder, E.J., Agee, R.C. (1998). Key management for multicast: Issues and architectures. Internet RFC 2627.

[3] Sherman, A.T., McGrew, D.A. (2003). Key establishment in large dynamic groups using one-way function trees. IEEE Transactions on Software Engineering, 29(5): 444-458. https://doi.org/10.1109/TSE.2003.1199073

[4] Mittra, S. (1997). Iolus: A framework for scalable secure multicasting. Journal of Computer Communication Reviews, 27(4): 277-288. https://doi.org/10.1145/263109.263179

[5] Kim, Y., Perrig, A., Tsudik, G. (2004). Tree-based group key agreement. ACM Transactions on Information Systems Security, 7(1): 60-96. https://doi.org/10.1145/984334.984337

[6] Amir, Y., Kim, Y., Nita-Rotaru, C., Schultz, J.L., Stan, J., Tsudik, G. (2004). Secure group communication using robust contributory key agreement. IEEE Transactions on Parallel and Distributed Systems, 15(5): 468-480. http://doi.org/10.1109/TPDS.2004.1278104

[7] Burmester, M., Desmedt, Y. (1994). A secure and efficient conference key distribution system. In: De Santis, A. (eds) Advances in Cryptology - EUROCRYPT'94. EUROCRYPT 1994. Lecture Notes in Computer Science, vol 950. Springer, Berlin, Heidelberg. https://doi.org/10.1007/BFb0053443

[8] Huang, J.H., Mishra, S. (2003). Mykil: A highly scalable key distribution protocol for large group multicast. In GLOBECOM '03. IEEE Global Telecommunications Conference (IEEE Cat. No.03CH37489), San Francisco, CA, USA. https://doi.org/10.1109/GLOCOM.2003.1258483

[9] Wu, B., Wu, J., Fernandez, E.B., Ilyas, M., Magliveras, S. (2007). Secure and efficient key management in mobile ad hoc networks. Journal of Network and Computer Applications, 30(3): 937-954. https://doi.org/10.1016/j.jnca.2005.07.008

[10] Yu, Z., Guan, Y. (2005). A key pre-distribution scheme using Deployment knowledge for wireless sensor networks. In Proceedings of the 4th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), Boise, ID, USA, pp. 261-268. https://doi.org/10.1109/IPSN.2005.1440934

[11] Renuka A., Shet, K.C. (2009): Hierarchical approach for key management in mobile Ad hocNetworks. International Journal of Computer Science and Information Security, 5: 87-95.

[12] Wu, B., Wu, J., Dong, Y. (2008). An efficient group key management scheme for mobile ad hoc networks. International Journal of Security and Networks, 6(4): 560-577.

[13] Kang, S., Ji, C., Hong, M. (2014). Secure collaborative key management for dynamic groups in mobile networks. Journal of Applied Mathematics, 2014: 601625. https://doi.org/10.1155/2014/601625

[14] Vaishnavi, N., Upadhay, H. (2016). Comprehensive study on key management schemes in MANET. International Academy of Science, Engineering and Technology, 5(2): 81-90.

[15] Vanathy, B., Ramakrishnan, M. (2020). Dynamic key distribution management using key escrow based ECC algorithm in MANETs. International Journal of Advanced Research in Engineering and Technology (IJARET), 11(1): 116-128.

[16] Sandhya Rani, M., Rekha, R., Sunitha, K.V.N. (2020). Multicast symmetric secret key management scheme in mobile Ad-hoc Networks. In: Satapathy, S.C., Raju, K.S., Shyamala, K., Krishna, D.R., Favorskaya, M.N. (eds) Advances in Decision Sciences, Image Processing, Security and Computer Vision. Learning and Analytics in Intelligent Systems, vol 3. Springer, Cham. https://doi.org/10.1007/978-3-030-24322-7_24