I. J. Computer Network and Information Security, 2013, 4, 48-56
Published Online April 2013 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2013.04.07

A Study on Contributory Group Key Agreements for Mobile Ad Hoc Networks

CH. V. Raghavendran, G. Naga Satish, P. Suresh Varma
Associate Professor, Ideal College of Arts & Sciences, Kakinada, India.
Associate Professor, Ideal College of Arts & Sciences, Kakinada, India.
Professor, Adikavi Nannaya University, Rajahmundry, India.
raghuchv@yahoo.com, gantinagasatish@gmail.com, vermaps@yahoo.com

Abstract **—** Wireless networks, in particular Mobile Ad hoc Networks (MANETs) have revolutionized the field of networking with increasing number of their commercial and military applications. Security on the other hand, is now an essential requirement for these applications. However, the limitations of the dynamic, infrastructure-less nature of MANETs impose major difficulties in establishing a secure framework suitable for such services. Security for MANETs is a dynamic area of research. Most of the traditional routing protocols proposed for MANETs are focused on routing only not on the security aspects. As in traditional wired networks, wireless networks also require security. Unlike the wired networks, where dedicated routers, servers control the network, in MANETs nodes act both as terminals and also as routers for other nodes. A popular mechanism to satisfy the security requirements is the Group Key Management in which the group key is to be shared by each group communication participant. But to establish and manage the group key efficiently imposes new challenges – especially in infrastructure less MANETs. The basic needs of such networks require that the group key schemes must demonstrate not only high performance but also fault-tolerance.

Index Terms **—** Mobile Ad hoc Networks (MANETs), Wireless Networks, Security, Group Key Management

I INTRODUCTION

A Mobile Ad hoc Network (MANET) consists of a collection of wireless mobile nodes that are capable of communicating with each other without the use of any centralized administration or network infrastructure. With the explosion of cheaper, smaller, and more powerful mobile devices, Mobile Ad hoc Networks (MANETs) have become one of the fastest growing areas of research. This new type of self-organizing network combines wireless communication with a high-degree node mobility. The union of nodes forms an arbitrary topology. This flexibility makes them attractive for many applications such as military applications, where the network topology may change rapidly to reflect a force's operational movements, and disaster recovery operations, where the existing infrastructure may be non-operational. The ad hoc self-organization also makes them suitable for virtual conferences, where setting up a wired network infrastructure is a time-consuming and high-cost task.

Nodes on MANETs use multi-hop communication: nodes that are within each other's radio range can communicate directly through wireless links, whereas those that are far apart must rely on intermediate nodes to act as routers to relay messages. Mobile nodes can move, leave, and join the network, and routes need to be updated frequently due to the dynamic network topology.

In the literature there are number of protocols proposed for MANET routing. There are communication overheads in detecting optimum routes with power saving and detection of malicious nodes or captured nodes. The basic requirements for a Secured network protocol for MANETs are – Confidentiality, Integrity, Availability and Non-repudiation. Issues and challenges for MANETs in security provisioning are – Dynamic Topology, Scalability, Autonomous, Poor Transmission Quality, Bandwidth Optimization, Device Discovery, Infrastructure less and Self Operated, Limited Resources, Limited Physical Security, Ad hoc Addressing, and Topology Maintenance.

The security attacks can generally be distinguished into two types – Passive and Active attacks. A MANET should provide a reliable and secure communication mechanism as nodes join or leave the network and their time of association with the network cannot be predicted. The data traffic in the ad hoc network travels through multiple hops routed through a vulnerable wireless medium, enhancing the security risk.

MANETs are vulnerable to diverse types of security attacks as the transmission takes place in the open medium and constraint resources. Portability has made devices each time smaller, with resource limitation, and thus easy targets for overload attacks [1, 2]. The network decentralization, absence of support infrastructure and the dynamic topology increase the vulnerability to many attacks as impersonation attacks, Sybil attacks [3], selective forwarding, black-hole, wormhole attacks [4, 5], among others. Many solutions have been proposed for security problems on ad hoc

net-works [1, 5, 6, 7]. In general, these solutions work in the preventive or reactive way and apply mechanisms and techniques to protect basic protocols and applications. However, techniques and mechanisms are used for a specific goal, being effective to one given case, but inefficient to others. Moreover, all existent techniques and mechanisms are themselves incapable of individually defending against all types of attacks and intrusions.

This paper studies the routing protocols for MANET based on the Group Key Management. The rest of the paper is organized as follows. Section II introduces the Group Key Management. Section III discusses the Operations and Metrics for Group Key Management. Section IV focus on classification of Group Key Management schemes. Section V discusses on Contributory Group Key agreements for MANETS. Section VI is challenges and future directions for key management. Finally Section VII represents the conclusion of this paper.

II Introduction to Group Key Management

Key management is a central part of the security of MANETs. In MANETs, the computational load and complexity for key management are strongly subject to restriction by the node's available resources and the dynamic nature of network topology. The fundamental goals of key management systems are to manage the keys used in the networks and simultaneously prevent the improper use of legally issued keys, such as the unauthorized modification, disclosure, or replaying keys, as well as the use of out-of-date keys, etc.

Cryptographic algorithms are security primitives that are widely used to provide the services like authentication, data integrity, non-repudiation and data confidentiality to establishment a Key Management Protocol. A group key is a piece of input information for cryptographic algorithms. If it was released, the encrypted information would be disclosed. For secure group communication in an Ad hoc network, a group key shared by all group members is required. Key management for large dynamic groups is a difficult problem because of scalability and security. Each time a new member is added or an old member is evicted from the group, the group key must be changed to ensure backward and forward security. Backward security means that new members cannot determine any past group key and discover the previous group communication messages. Forward security means that evicted members cannot determine any future group key and discover the subsequent group communication information. The group key management should also be able to resist against colluded members.

Mobile nodes come together to form an ad hoc group for secure communication purpose. A key distribution system requires a trusted third party that acts as a mediator between nodes of the network. But in Ad hoc networks characteristically do not have a trusted authority. Group Key Agreement means that multiple parties want to create a common secret key to be used to exchange information securely. Furthermore, group key agreement also needs to address the security issues related to membership changes due to node mobility. The membership change requires frequent changes of group key. This can be done either periodically or updating every membership changes. The changed group key ensures backward and forward secrecy.

With frequent changes in group memberships, the recent researches began to pay more attention on the efficiency of group key update. Group key agreement is a building block in secure group communication in Ad hoc networks. The Group Key Management scheme includes important phases like – key generation phase, key distribution phase, and frequency-based key update phase.

According to recent literature, the centralized approach is regarded as inappropriate for MANETs because of the dynamic environment and the transient relationships among mobile nodes. Most researchers prefer the decentralized trust model for MANETs. Several decentralized solutions have been proposed in recent papers with different implementations, such as how the CA's responsibility is distributed to all nodes, or to a subset of nodes.

III Operations and Metrics for Group Key Management

A. Dynamic Group Key Operations:

In order to accurately and fairly evaluate group key management protocols, definitions are needed that explain the operations used by the main group key management protocols. In the literature, it was found that there are eight major operations that were essential for establishing and sharing keys across the dynamic group. The Fig. 1 shows these operations.

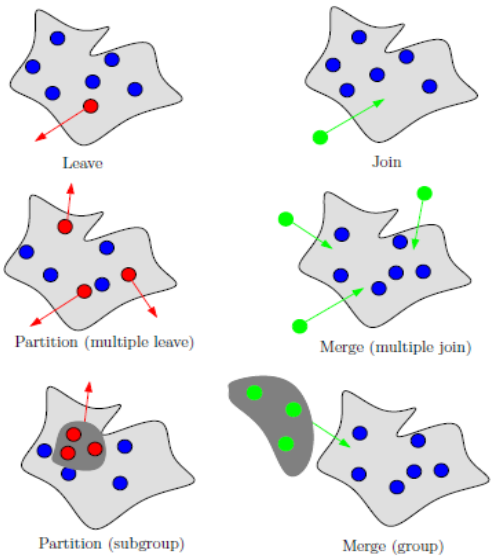

Figure 1. Group key operations

Initialization operation – This is the initial creation of the group key and organization of the key management infrastructure

Join – This operation brings a new member into the existing group.

Mass join (Mass add) – This operation allows many new members to be added to an existing group simultaneously when these new members have not already formed a group of their own.

Merge (group fusion) – This operation, as opposed to mass join, is used when another group is combined with the existing group to become a new group.

Leave – His operation is used to remove a member from the group.

Mass leave – This operation is used when multiple members are simultaneously removed from the existing group.

Split (partition, or group fission) – This operation, different from mass leave, occurs when a single group is divided into two or more component groups.

Key refresh – This operation is to prevent the secret key from being used for a long time. Moreover, to prevent an adversary from breaking in, we should refresh the original key and generate a new secret key periodically.

B. Performance Metrics:

The following attributes are Performance metrics used to evaluate the efficiency of contributory key management protocols:

Number of rounds – The protocol should try to minimize the number of iterations among the members to reduce processing and communication requirements.

Number of unicast messages – This is number is the sum of the number of messages every member sends to other single members in the group per operation. This number is useful for determining total communication and is important if many or all nodes are on the same network collision domain, thus forcing these messages to be sent sequentially rather than simultaneously.

Number of broadcast messages – This is the sum of the number of messages sent by each member to all the other members in the group per operation. Since the messages go to all members of the group, it greatly affects total communication costs depending on the underlying network topology.

Number of messages – This is the sum of the number of unicast messages and broadcast messages. This number is used to determine the total time of communication in an underlying broadcast network The overhead introduced by every message exchanged between members produces unbearable delays as the group grows. Therefore, the protocol should require a minimum number of messages.

Processing during setup – Computations needed during setup time. Setting up the group requires most of the computation involved in maintaining the group, because all members need to be contacted.

DH key – Identify whether the protocol uses Diffie–Hellman (DH) [Diffie and Hellman 1976] to generate the keys. The use of DH to generate the group key implies that the group key is generated in a contributory fashion.

Number of sequential exponentiations – During an operation there will be a series of computationally expensive cryptographic operations (such as modular exponentiation used in the DH protocol). The protocols in the literature often require the results of one cryptographic operation prior to the execution of another. This metric represented the worst case scenario, the longest sequence of dependencies of these cryptographic calculations in the operation.

Number of signatures – This is the sum of digital signatures used in every round. In every round, the node initiating the operation sends one digital signature.

Number of verifications – Given that each message needs to be verified, the number of verifications is equal to the number of messages; however, several verifications can occur in parallel so care is needed with the number of sequential verifications that must occur during an operation

IV Group Key Management Agreements

The Group key management protocols can be approximately classified into three categories [8] as

- Centralized Group Key Distribution (CGKD)
- De-centralized Group Key Management (DGKM)
- Contributory/distributed Group Key Agreement (CGKA)

A. Centralized Group Key Distribution (CGKD):

In CGKD, there exists a central entity called as group controller (GC) which is responsible for generating, distributing, and updating the group key. One of the famous CGKD scheme is the Logical Key Hierarchy (LKH). This was proposed by several research groups nearly at the same time, followed by many researchers proposing improvements and enhancements [9, 10, 11, 12, 13, 14]. Some of the Centralized Group Key Management protocols are

- Group Key Management Protocol
- Logical Key Hierarchy
- One-way Function Tree
- One-way Function Chain Tree
- Hierarchical a-ary Tree with Clustering
- Centralized Flat Table
- Efficient Large-Group Key

B. De-centralized Group Key Management (DGKM):

The DGKM approach involves splitting a large group into small subgroups. Each subgroup has a subgroup controller which is responsible for the key management of its subgroup. Subgroup controllers are also in charge of relaying encrypted data messages. The first DGKM scheme was IOLUS [15]. There followed some improvements and hierarchical group key management schemes [16, 17, 18]. Some of the De-centralized Group Key Management protocols are

- Scalable Multicast Key Distribution

- Iolus
- Dual-Encryption Protocol
- MARKS
- Cipher Sequences
- Kronos
- Intra-Domain Group Key Management\
- Hydra

C. Contributory Group Key Agreement (CGKA):

The CGKA schemes involve the participation by all members of a group towards key management. Such schemes are characterized by the absence of the GC. The group key in such schemes is a function of the secret shares contributed by the members. Being contributory in nature, the distributed schemes help in the uniform distribution of the work-load for key management and eliminate the requirement for a central trusted entity. Typical CGKA schemes include binary tree based ones [19] and n-party Diffie-Hellman key agreement [20, 19, 21]. Some of the Contributory Group Key Management protocols are

- Burmester and Desmedt Protocol
- Octopus Protocol
- Group Diffie–Hellman Key Exchange
- Conference Key Agreement
- Distributed Logical Key Hierarchy
- Tree–Based Group Diffie–Hellman
- Skinny Tree
- Distributed One-way Function Tree
- Diffie–Hellman Logical Key Hierarchy
- Distributed Flat Table

V Contributory Group Key Agreements

MANETs can be used in all those situations where there is no time or resources available to setup a backbone network or infrastructure. With their increasing usage, Secure Group Communication (SGC) over such networks becomes vital. In MANETs, since there is no pre-defined/fixed infrastructure, a Central Authority (CA) is not usually available, and there are generally equivalent levels of power and trust among the participating members. A contributory group key agreement scheme is most appropriate for SGC in this kind of environment. Several group key management schemes have been proposed for SGC in wireless networks. The Fig. 2 shows the required components for implementing contributory group key management schemes.

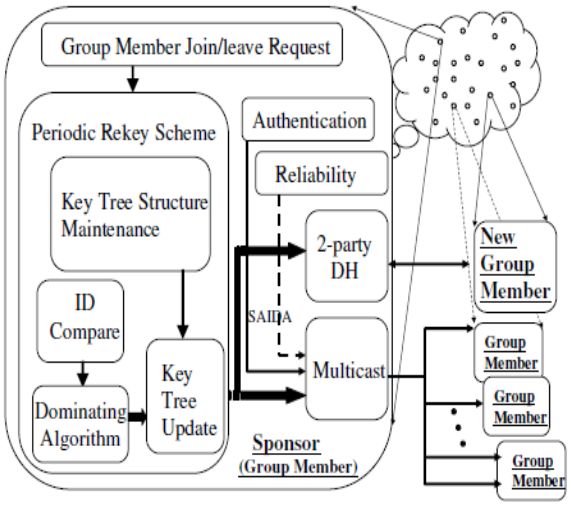

Figure 2: Contributory Group Key Management

A. Burmester and Desmedt (B-D) Protocol:

BD is a distributed group key management scheme proposed by Burmester and Desmedt [22]. It is an extension of the Diffie-Hellman key distribution system. The group key can be calculated in three rounds which are as follows:

Round 1 – Broadcast the partial keys
Round 2 – Calculate key material
Round 3 – Compute the group key

However, when a new group member joins or when a group member leaves, most of the group members need to refresh the random session and follow the steps mentioned above one by one. Therefore, this scheme requires high resources.

B. Octopus Protocol:

This protocol was proposed by Becker and Wille [1998]. This protocol is also based on DH key exchange protocol. In Octopus, the large group of *n* members is split into four subgroups namely, A, B, C and D. In every sub-group, there is one group member leader available, namely M_A, M_B, M_C or M_D, respectively. The leader member in each subgroup is responsible for collecting contributions from all its subgroup members and calculating the intermediary DH value I_A (or I_B, I_C, I_D,). Then, the four group leaders launch the DH scheme to compute group key G and send G back to every sub-group member. Specifically, the group is split and the intermediate values are computed as below (r_i is the contribution of group member M_i): Subgroup A includes group members M_1 … M_n=4; the leader of sub group A, calculates $I_A = II_{1<i<n/4}\ r_i$. The sub-group leader, for example, A, should has a secure channel between every other subgroup member. Via these channels, A can obtain $r_1,\ldots, r_{n/4}$ one by one. The same applies to B, C, or D. After the completion of this procedure, group key G can be computed as described below.

First, *A* and *B*, using DH, exchange their intermediary values (I_a and I_b) creating $\alpha^{Ia.Ib}$. Also, *C* and *D* do the same and create $\alpha^{Ic.Id}$. Then, *A* and *C* exchange $\alpha^{Ia.Ib}$ and $\alpha^{Ic.Id}$. Leaders *B* and *D* do the same.

Now, all of them can calculate $\alpha^{Ia.Ib.Ic.Id}$. After that, *A*, *B*, *C* and *D* send to their respective subgroups $\alpha^{Ia.Ib.Ic.Id / ui}$, where $i = 1 \ldots (n\text{-}4)/4$, and all members of the group are capable of calculating the group key.

C. Group Diffie–Hellman Key Exchange (GDH):

GDH is a group key distribution scheme proposed by Steiner, Tsudik, and Waidner [23] in three versions. GDH extends the two-party Diffie-Hellman key exchange protocol into a group operation. GDH actually contains three key distribution schemes that are extended from the DH protocols. In GDH.1 and GDH.2, the overhead of computation is quite considerable due to the total of $O(n^2)$ exponentiation calculations. GDH.3 has been proposed to reduce this, in which every member only needs to perform a constant small number of exponentiation computations.

The *first stage* involves collecting contributions from all group members and user U_{n-1} obtains

$$g^{\prod\{N_k | k \in [1, n-1]\}} \quad (1)$$

In the *second stage* this value broadcasts to all other group members. At the *third stage*, every user U_i ($i \neq n$) factors out its own exponent and forwards the result to the last user U_n. At the *final stage*, U_n collects all inputs from the previous stage, raises every one of them to the power of N_n and broadcasts the resulting n-1 values to the rest of the group. In the end, every group member has a value of the form

$$g^{\prod\{N_k | k \in [1, n] \wedge k \neq i\}} \quad (2)$$

and can easily compute the group key K_n.

An important drawback of GDH.3 is that the success or failure of the group key generation depends on the performance of the last member in the group. The last group member receives *n* messages and needs to perform *n* exponentiations computations. This requires the member to have plenty of storage space and strong computational power. However, not every member has such power, especially in wireless networks. As the group size increases, the time taken for the key generation significantly increases and this causes a slow response to membership changes. Member addition and deletion can be handled easily in this scheme.

D. Conference Key Agreement (CKA):

Boyd [24] proposed a protocol for conference key agreement (CKA) where all group members contribute to generate the group key. The group key is generated with a combining function: $K = f(N_1, h(N_2), \ldots, h(N_n))$, where *f* is the combining function, *h* is a one-way function, *n* is the group size and *Ni* is the contribution from group member *i*. The protocol specifies that *n-1* members broadcast their contributions (*Ni*) in the clear. The group leader, for example U_1, encrypts its contribution N_1 with the public key of each *(n-1)* group member and broadcasts it. All group members who had their public key used to encrypt *N*1 can decrypt it and generate the group key.

E. Distributed Logical Key Hierarchy (D-LKH):

This approach is proposed by Rodeh [25] as an extension to Centralized Logical Key Hierarchy without a support of a centralized server. In this approach, the Group Controller (GC) is completely abolished and the logical key hierarchy is generated among the members, therefore there is no entity that knows all the keys at the same time. In this scheme, every group member plays a symmetric role. This solution utilizes the logical tree which has two groups of members namely, left sub tree L and right sub tree R. Member M_L is assumed to be L's leader and member M_R is R's leader. Every group member in L agrees on a shared key K_L, and those in R, a shared key K_R. The protocol used to agree on a mutual key goes groups L and R as follows:

1. M_L, the group leader of L, chooses a new key K_{LR} and sends it to the group leader of R, M_R, using a secure channel.
2. M_L encrypts K_{LR} with key K_L and multicasts the cipher text to its group members in L. M_R encrypts K_{LR} with key K_R and multicasts the cipher text to its group members in R.
3. All members within L and R receive the cipher text and decrypt the group key, K_{LR}.

F. Tree–Based Group Diffie–Hellman (TGDH):

In [26][19] Kim et al. has proposed a contributory group key agreement protocol named Tree-based Group Diffie-Hellman (TDGH) as an extension to the two-party DH protocol. This brought two important trends in group key management together viz., 1) key trees to efficiently compute and update group key and 2) Diffie-Hellman key exchange to achieve secure and fully distributed protocols. TGDH protocol suite has four protocols: *join*, *leave*, *merge*, and *partition*. These share a common framework with the following features:

- Each group member contributes its equal share to the group key, which is computed as a function of all shares of current group members.
- This share is secret (private to each group member) and is never revealed.
- As the group grows, new member's shares are factored into the group key but old member's shares remain unchanged.
- As the group shrinks, departing member's shares are removed from the new key and at least one remaining member changes its share.
- All protocol messages are signed, time-stamped, sequence-numbered and type-identified by the sender.

In TGDH the group key is derived from the contributions of all group members. All members maintain an identical virtual binary tree that may or may not be balanced. Each member is associated with a leaf node in the key tree. Members use Diffie-Hellman protocol to generate the keys along the path from its leaf node to the root.

In this protocol group member can take on a special *sponsor* role which involves computing intermediate

keys and broadcasting to the group. Each broadcasted message contains the sender's view of the key tree which contains each blind key known to the sender. Any member in the group can unilaterally take on this responsibility, depending on the type of membership event. In case of join or merge, all group members identify a unique sponsor. This sponsor is responsible for updating its secret key share, computing affected [*key, blind key*] pairs and broadcasting all blind keys of the new tree to the rest of the group. In response to a leave or partition, all members update the tree in the same manner. Group partition results in a smaller tree since some leaf nodes disappear. As a result, some sub-trees acquire new siblings; therefore, new intermediate keys and blind keys must be computed through a Diffie-Hellman exchange between the new sibling's sub-trees.

TGDH is operational efficient in communication and computation, because only one round is required to calculate the group key. The sponsor only needs to send one keying message. The keying message contains $\log_2 n$ blinded keys (n is the number of users in the group). The members in the group perform, at most, $\log_2 n$ exponentiations computation to reach the group key. But, this protocol relies on a sponsor. If the sponsor fails, the whole key updating procedure stops. And also, each member needs to maintain an identical virtual binary tree. Substantial, storage space is required to maintain such a tree structure for a large group.

G. Skinny Tree (STR):

This is an extension of one of the earlier tree-based group key management by Steer *et al.* [27] proposed by Kim [19] to handle membership events. This scheme utilizes an unbalanced key tree in which every leaf node represents a group member. All intermediate nodes play a management role. In this the height of the key tree is always ($n-1$), as opposed to $\log(n)$ in TGDH. All other features of the key tree are the same as in TGDH.

Every group member M_i should generate a random secret r_i and calculate its leaf node's blinded key $BK_{<i;1>} = \alpha^{ri}$. In the first round, every member broadcasts $BK_{<i;1>} = \alpha^{ri}$ where $1 \leq i \leq n$ and n is the group size. To handle group members joining, STR adds a new leaf node to represent the new member. This new leaf node is treated as the current root's sibling and a new root node is created which works as the former root and the new member's parent. The group member representing with the leaf node right below the new leaf node is selected as the sponsor. When a group member leaves, the leaf node representing the leaving group member and the corresponding sibling node are deleted. The group member represented by the leaf node right below the leaving member's leaf node is treated as the sponsor. Finally, the updated blinded keys are multicast and every other group member can calculate the new group key. STR reduces the number of rounds needed to update the group key as compared with TGDH.

H. Distributed One-way Function Tree (D-OFT):

Dondeti et al. [28] proposed an approach using logical key hierarchy in distributed fashion as an extension to One-way Function Tree (OFT). As in TGDH, D-OFT uses the binary key tree for group generation and group management. Every group member is trusted with access control and key generation. A member is responsible for generating its own key and sending the blinded version of this key to its sibling. The blind key is calculated as below:

$$K_B = g(K) \quad (3)$$

where K: node key; K_B : blinded key; g: one way hash function;

The leaf node's key is generated by the group member and the intermediate node key is computed according to the formula:

$$K_i = f\,(g\,(K_{\text{left\_child (i)}}\,,\, K_{\text{right\_child (i)}})) \quad (4)$$

where f: mix function to mix together the two parameters.

The leaf member calculate the group key by knowing of all node keys on its key path and all blinded keys on its sibling key path. When group members join/leave, new contributions of the changing member's sibling should be refreshed and the corresponding node key and blinded key on its key path should be updated. A secure channel is assumed between the group members to send the updated node key and blinded key. This method is computationally efficient than TGDH, since the one-way hash function rather than the exponential operation is used to calculate the blinded key and node key. The secure channel between group members is a limit for this proposal.

I. Diffie–Hellman Logical Key Hierarchy (DH-LKH):

Perrig [29] and Kim et al. [26] also used a logical key hierarchy to minimize the number of key held by group members. The difference here is that group members generate the keys in the upper levels using the Diffie–Hellman algorithm rather than using a one-way function. The tree is built recursively from bottom to up. Initially, each member M_i generates a random r_i as a secret key associated to its leaf. To build upper level of the tree, two members: one as a leader of a left sub-tree and another one as a leader of a right sub-tree, broadcast their respective DH computations and hence allow to all the members to calculate the group key corresponding to the root of the tree. Because of the logical key hierarchy the number of key calculations are reduced from the order of $O(n)$ to $O(logn)$.

J. Distributed Flat Table (DFT):

Waldvogel *et al.* [30] extends further its solution, proposing to use the flat table in a distributed (DFT) fashion with no Group Controller (GC). In this scheme, no member knows all the keys at any time. Each member knows only the Key Encryption Keys (KEKs) that it is entitled to. The inconvenience in this is that a joining member is obliged to contact a group of

members to get all the keys needed. Furthermore, since many members could be changing the same key at the same time, there could be serious delays in synchronizing the keys.

VI Open Challenges and Future Directions

Security is an important feature that determines the success and degree of deployment of MANETs. Key management is in the central part of any secure communication and is the weakest point of the security. Security of MANETs is more challenging because of the host mobility, shared wireless medium, resource constraint of physical devices, and lack of a fixed and trustable control point. Designing a key management system is a difficult problem that has received increased attention recently. The current research on key management in MANETs is still at its early stage. The security of group communication involves the management of group keys. Most contributory group key distributions are based on DH protocol with different implementations. Recent references paid more attentions to contributory and collaborative group key agreement, for example: [31–35], etc.

In [31] Shanyu proposed a Communication–Computation Efficient Group Key Algorithm (CCEGK) to provide efficient communication and computation, addressing performance, security and authentication. This group key management algorithm based upon two preceding group key management algorithms, EGK and TGDH. This algorithm fully implements an initialization operation and presents two mass leave operations, mass leave-balanced, and mass leave-imbalanced, while the TGDH algorithm only details a mass leave-imbalanced. TGDH and STR do not implement a balance operation, and CCEGK does.

R. Dutta and R. Barua [32] have proposed a new protocol named as DB protocol, which is an extension to the Burmester-Desmedt (BD) protocol. This has important differences as – simple key computation, less number of rounds for authentication, ability to detect corrupted group members and reduced computation complexities.

Sun and Liu [33] have presented a contributory group key agreement with a new logical key tree structure called PFMH. This scheme needs O(1) rounds of two-party DH for single user join event and O(log n) for single user leave event. This achieves lower rekeying cost than the existing tree-based contributory group key agreements.

In [34] Mao and Sun have proposed a Join-Exit-Tree (JET) key management framework for better time efficiency during member join and departure. This framework has reduced the time for a member join or departure event to O(log (log n)) from O(log n).

Balachandran *et al*. proposed contributory key agreement protocol for MANETs, called Chinese Remainder Theorem and Diffie-Hellman (CRTDH) [35]. This has shown solutions for two important problems of Secure Group Communication – requirement of member serialization and existence of central entity.

VII Conclusion

In this paper we have studied contributory group key management protocols for Secure Group Communications for Mobile Ad-hoc Networks. Based on different assumptions, many key management protocols have been proposed for MANETs. But none of the protocols can be used as a universal solution for all classes of Mobile Ad-hoc groups. A protocol should be chosen according to the expected dynamic behavior in the group, required level of the security for the application, and priority between computation and communication constraints. All key management approaches are subject to various restrictions such as the mobile device's available resources, the network bandwidth, and MANETs' dynamic nature. An efficient key management protocol for MANETs is an ongoing hot research area.

References

[1] P. Papadimitratos and Z. Haas. Handbook of ad hoc wireless networks, chapter Securing mobile ad hoc networks. CRC Press, 2002.

[2] L. Zhou and Z. J. Haas. Securing ad hoc networks. IEEE network, 13(6):24–30, 1999.

[3] J.Douceur. The sybil attack. In Proceedings of the International workshop on peer-to-peer systems (IPTPS), Cambridge, MA (USA), March 2002.

[4] H. Yang, H. Luo, J. Kong, F. Ye, P. Zerfos, S. Lu, and L. Zhang. Ad hoc network security: challenges and solutions. CRC Press, 2004.

[5] B. Wu, J. Chen, J. Wu, and M. Cardei. Wireless/mobile network security, chapter A survey on attacks and countermeasures in mobile ad hoc networks. Springer, 2006.

[6] D. Djenouri, L. Khelladi, and A. N. Badache. A survey of security issues in mobile ad hoc and sensor networks. IEEE Communications surveys & tutorials, 7(4):2–28, 2005.

[7] P. Argyroudis and D. O'Mahony. Secure routing for mobile ad hoc networks. IEEE Communications surveys & tutorials, 7(3):2–21, Third Quarter 2005.

[8] Rafaeli, S. and Hutchison, D. (2003). A Survey of Key Management for Secure Group Communication. ACM computing Surveys, vol. 35, no. 3, pp. 309-329.

[9] P. Lee, J. Lui, and D. Yau. Distributed collaborative key agreement protocols for dynamic peer groups. Proc. IEEE Int'l Conf. Network Protocols (ICNP), pages 322.333, Nov. 2002.

[10] X. Li, Y. Yang, M. Gouda, and S. Lam. Batch rekeying for secure group communications. Proc. 10th Int'l WWW Conf., pages 525.534, May 2001.

[11] W. H. D. Ng, M. Howarth, Z. Sun, and H. Cruickshank. Dynamic balanced key tree management for secure multicast communications. IEEE Transactions on Computers, 56(5):577.589, May 2007.
[12] J. Pegueroles and F. Rico-Novella. Balanced batch lkh: New proposal, implementation and performance evaluation. Proc. IEEE Symp. Computers and Comm. (ISCC), pages 815.820, June 2003
[13] A. T. Sherman and D. A. McGrew. Key establishment in large dynamic groups using one-way function trees. IEEE transactions on Software Engineering, 29(5):444.458, May 2003.
[14] X. B. Zhang, S. S. Lam, D.-Y. Lee, and Y. R. Yang. Protocol design for scalable and reliable group rekeying. Proceedings SPIE Conference on Scalability and Traffic Control in IP Networks, pages 87.108, Aug. 2001.
[15] S. Mittra. Iolus: A framework for scalable secure multicasting. Journal of Computer Communication Reviews, 27(4):277.288, 1997.
[16] S. Banerjee and B. Bhattacharjee. Scalable secure group communication over IP multicast. IEEE Journal on Selected Areas in Communications, 20(8):1151.1527, 2002.
[17] S. Rafaeli and D. Hutchison. Hydra: A decentralized group key management. Proceedings of 11th IEEE International WETICE: Enterprise Security Workshop, 2002.
[18] S. Setia, S. Koussih, and S. Jajodia. Kronos: A scalable group re-keying approach for secure multicast. Proceedings of IEEE Symposium on Security and Privacy, 2000.
[19] Y. Kim, A. Perrig, and G. Tsudik. Tree-based group key agreement. ACM Transactions on Information Systems Security, 7(1):60.96, Feb. 2004.
[20] Y. Amir, Y. Kim, C. Nita-Rotaru, J. L. Schultz, J. Stan, and G. Tsudik. Secure group communication using robust contributory key agreement. IEEE Trans. Parallel and Distributed Systems, 15(5):468.480, 2004.
[21] M. Steiner, G. Tsudik, and M. Waidner. Key agreement in dynamic peer groups. IEEE Transactions on Parallel and Distributed Systems, 11(8):769.780, Aug. 2000.
[22] Burmester, M. and Desmedt, Y. (1994). A Secure and Efficient Conference Key Distribution system. In A. De Santis, editor, Advances in Cryptology – EUROCRYPT '94, no. 950.
[23] Steiner, M., Tsudik, G., and Waidner, M.(2000). Cliques: A New Approach to Group Key Agreement. IEEE Transactions on Parallel and Distributed Systems.
[24] BOYD, C. 1997. On key agreement and conference key agreement. In Proceedings of the Information Security and Privacy: Australasian Conference. Lecture Notes in Computer Science, vol. 1270. Springer-Verlag, New York, 294–302.
[25] RODEH, O., BIRMAN, K., AND DOLEV, D. 2000. Optimized group rekey for group communication systems. In Network and Distributed System Security.(San Diego, Calif.)
[26] Kim,Y., Perrig, A., and Tsudik, G. 2000. Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups. Technical Report 2, USC Technical Report 00-737.
[27] D. Steer, L. Strawczynski, W. Di_e, and M. Wiener, A secure audio teleconference system, Advances in Crytology (CRYPTO 88), pp. 520-528, Santa Barbara, California, USA, Aug. 1988.
[28] DONDETI, L., MUKHERJEE, S., AND SAMAL, A. 1999. A distributed group key management scheme for secure many-to-many communication. Tech. Rep. PINTL-TR-207-99,Department of Computer Science, University of Maryland.
[29] PERRIG, A. 1999. Efficient collaborative key management protocols for secure autonomous group communication. In Proceedings of the International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC'99). (Hong Kong, China, July). M. Blum and C H Lee, Eds. City University of Hong Kong Press, Hong Kong, China, pp. 192–202.
[30] Waldvogel, M., Caronni, G., Sun, D., Weiler, N., And Plattner, B. 1999. The VersaKey framework: Versatile group key management. IEEE J. Sel. Areas Commun. (Special Issue on Middleware) 17, 9 (Aug.), 1614–1631.
[31] Shanyu Zheng, David Manz, Jim Alves-Foss, "A Communication Computation Efficient Group Key Algorithm for Large and Dynamic Groups", Elsevier, Computer Networks, March 2006.
[32] R. Dutta and R. Barua, "Provably Secure Constant Round Contributory Group Key Agreement in Dynamic Setting," IEEE Transactions On Information Theory, vol. 54, no. 5, pp.2007-2025, May 2008.
[33] W. Yu, Y. Sun, and K. J. R. Liu, "Optimizing Rekeying Cost for Contributory Group Key Agreement Schemes," IEEE Transactions On Dependable And Secure Computing, vol. 4, no. 3, pp.228-242, 2007.
[34] Y. Mao, Y. Sun, M.Wu and K. J. R. Liu, "JET: Dynamic Join-Exit-Tree Amortization and Scheduling for Contributory Key Management," IEEE/ACM Transactions on Networking, vol. 14, no. 5, pp. 1128-1140, Oct. 2006.
[35] R. Balachandran, B. Ramamurthy, X. Zou, and N. Vinodchandran. CRTDH: An efficient key agreement scheme for secure group communications in wireless ad hoc networks. Proceedings of IEEE International Conference on Communications (ICC), pages 1123.1127, 2005.

CH.V. Raghavendran has received MCA and M.Tech (CSE) degrees from Nagarjuna University in 1994 and 2010 respectively. He received his M.Phil in Computer Science in 2008 from Alagappa University. He is a research scholar in Compute Science Department of Adikavi Nannaya University, Rajahmundry, AP. He has published over 10 papers in various National and International Conferences. He is working as a Director of P.G. Dept. of Computer Sciences, Ideal College of Arts & Sciences, Kakinada, AP. His areas of interest are Mobile Ad hoc Networks, Swarm Intelligence and Data Mining.

Ganti Naga Satish is working as Associate Professor in P.G. Department of Computer Sciences, Ideal College of Arts & Sciences, Kakinada, Andhra Pradesh, India. His qualifications are M.Sc, M.Phil, M.Tech. He is pursing Ph.D at Adikavi Nannaya University. He has presented and published papers in National and International Conferences. His areas of interest include Computer Networks

Dr. P. Suresh Varma received the Master's degree M.Tech in Computer Science & Technology from Andhra University. He received Ph.D. degree in Computer Science & Engineering from Acharya Nagarjuna University. He is currently working as Professor in Department of Computer Science in Adikavi Nannaya University, Rajahmundry, A.P., India. He published several papers in National and International Journals. He is active member of various professional bodies. His current research is focused on Computer Networks, Cloud Computing and Data Mining.

