Simple and Energy Efficient Group Key Management in Mobile Ad hoc Networks

Renuka A.

Dept. of Computer Science and Engg. Manipal Institute of Technology Manipal-576104-India 91-0820-2924515

renuka.prabhu@manipal.edu

Dr. K.C.Shet

Dept. of Computer Engg. National Institute of Technology, Karnataka Surathkal, 575025 India

kcshet@rediffmail.com

ABSTRACT

A Mobile Ad-hoc Network (MANET) is a collection of autonomous nodes or terminals which communicate with each other by forming a multi-hop radio network and maintaining connectivity in a decentralized manner. The conventional security solutions to provide key management through accessing trusted authorities or centralized servers are infeasible for this new environment since mobile ad hoc networks are characterized by the absence of any infrastructure, frequent mobility, and wireless links. We propose a simple and energy efficient group key management scheme that is 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 message in our scheme uses less number of bits which result in the saving of nearly 25% of energy. We show through analysis and simulations that our scheme has less computation, communication and energy consumption compared to the existing schemes.

Categories and Subject Descriptors

C2.0 [Computer Communications-Networks]: General -Security and Protection ; C 2.1 [Computer Communications-Networks]:Network Design and Architecture - Wireless Communication

General Terms

Security , Performance

Keywords

group key, key management, mobile ad hoc network, rekeying

1 INTRODUCTION

A mobile ad hoc network (MANET) is a collection of autonomous nodes that communicate with each other, most frequently using a multi-hop wireless network. Nodes do not necessarily know each other and come together to form an ad hoc group for some specific purpose. Key distribution systems usually require a trusted third party that acts as a mediator between nodes of the network. Ad hoc networks typically do not have an online trusted authority but there may be an off line one that is used during system initialization. A node in an ad hoc network has direct connection with a set of nodes, called neighboring nodes, which are in its communication range. The number of nodes in the network is not necessarily fixed. New nodes may join the network while existing ones may be compromised or become un-functional.

Group key establishment means that multiple parties want to create a common secret to be used to exchange information securely. Without relying on a central trusted entity, two people who do not previously share a common secret can create one based on the party Diffie Hellman (DH) protocol. The 2-party Diffie Hellman protocol can be extended to a generalized version of n-party DH. Furthermore, group key management also needs to address the security issue related to membership changes. The modification of membership requires refreshment of the group key. This can be done either by periodic rekeying or updating right after member change. The change of group key ensures backward and forward security. With frequently changing group memberships, recent researches began to pay more attention on the efficiency of group key update. ecently, collaborative and group-oriented applications in MANETs have been an active research area. Obviously, group key management is a central building block in securing group communications in MANETs. However, group key management for large and dynamic groups in MANETs is a difficult problem because of the requirement of scalability and security under the restrictions of nodes' available resources and unpredictable mobility

We propose a distributed group key management approach wherein there is no central authority and the users themselves arrive at a group key through simple computations. In large and high mobility mobile ad hoc networks, it is not possible to use a single group key for the entire network because of the enormous cost of computation and communication in rekeying. So, we logically divide the entire network into a number of subgroups called clusters headed by the cluster head. Though we use the term cluster head, the cluster head is in no way different from the other members except for maintaining the information about the members of the cluster and sending beacon information to its members. The transmission power and memory of the cluster head is same as other members. The members within the cluster communicate with the help of a group key. Inter cluster communication take place with the help of gate way nodes. Each member also carries a public key, private key pair used to encrypt the rekeying messages exchanged. This ensures that the backward secrecy is preserved.

The rest of the paper is organized as follows. Section 2 focuses on the related work in this field. The proposed scheme is presented in

Section 3. Performance analysis of the scheme is discussed in Section 4. Experimental Results and Conclusion are given in Section 5and Section 6 respectively.

2 RELATED WORK

Key management is a basic part of any communication system. Most cryptosystems rely on some underlying secure, robust, and efficient key management system. Group key establishment means that multiple parties want to create a common secret to be used to exchange information securely. Secure group communication (SGC) is defined as the process by which members in a group can securely communicate with each other and the information being shared is inaccessible to anybody outside the group. In such a scenario, a group key is established among all the participating members and this key is used to encrypt all the messages destined to the group. As a result, only the group members can decrypt the messages. The group key management protocols are typically classified in four categories: centralized group key distribution (CGKD), de-centralized group key management (DGKM), distributed/contributory group key agreement (CGKA), and distributed group key distribution (DGKD).

In CGKD, there exists a central entity (i.e. a group controller (GC)) which is responsible for generating, distributing, and updating the group key. The most famous CGKD scheme is the key tree scheme (also called Logical Key Hierarchy (LKH) proposed in [1] is based on the tree structure with each user (group participant) corresponding to a leaf and the group initiator as the root node. The tree structure will significantly reduce the number of broadcast messages and storage space for both the group controller and group members. One Way Function (OFT) is another centralized group key management scheme proposed in [2].similar to LKH. However, all keys in the OFT scheme are functionally related according to a one-way hash function

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. The first DGKM scheme to appear was IOLUS [3]. 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 [4] and *n*-party Diffie-Hellman key agreement [5, 6]. Tree Based Group Diffie Hellman (TGDH) is a group key management scheme proposed in [4]. The basic idea is to combine the efficiency of the tree structure with the contributory feature of DH. The DGKD scheme, proposed in [7], eliminates the need for a trusted central authority and introduces the concepts of sponsors and co distributors. Any group member could be a potential sponsor of other members or a co-distributor. Whenever a member joins or leaves the group, the member's sponsor initiates the rekeying process. The sponsor generates the necessary keys and securely distributes the keys to co-distributors respectively. The co distributors then distribute in parallel, corresponding keys to corresponding members. In addition to the above four typical classes of key management schemes, there are some other forms of key management schemes such as hierarchy and cluster based ones [6, 8]. 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 [9, 10]. In Simple and Efficient Group Key (SEGK) management scheme for MANETs proposed in [11] group members compute the group key in a distributed manner. Also, a new approach was developed in [12] called BALADE, based on a sequential multi-sources model, and takes into account both localization and mobility of nodes, while optimizing energy and bandwidth consumptions. Most of these schemes involve complex operations which is not suitable for large and high mobility networks. In Group Diffie- Hellman, the group agrees on a pair of primes and starts calculating in a distributive fashion the intermediate values. The setup time is linear since all members must contribute to generating the group key. Therefore, the size of the message increases as the sequence is reaching the last members and more intermediate values are necessary. With that, the number of exponential operations also increases. Therefore this method is not suitable for large networks. Moreover the computational burden is high since it involves a lot of exponentiations.

Another approach using logical key hierarchy in a distributed fashion was proposed in [13] called Distributed One-way Function Tree (D-OWT) This protocol uses the one-way function tree. A member is responsible for generating its own key and sending the blinded version of this key to its sibling. Reference [14] also uses a logical key hierarchy to minimize the number of key held by group members called Diffie–Hellman Logical Key Hierarchy. 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. In Chinese Remainder Theorem Diffie-Hellman (CRTDH) [15] each member computes the group key as the XOR operation of certain values computed. This requires that the members agree on two large primes. CRTDH is impractical in terms of efficiency and security. such as low efficiency, possibly a small key, and possessing the same Least Common Multiple (LCM). However this CRTDH scheme was modified in [16] wherein the evaluation of the LCM was eliminated and other steps were modified slightly, so that a large value for the key is obtained. In both these methods, whenever membership changes occur, the new group key is derived from the old group key as the XOR function of the old group key and the value derived from the Chinese Remainder Theorem values broadcast by one of its members. Since it is possible for the leaving member to obtain this message, and hence deduce the new group key backward secrecy is not preserved.

In this paper, we propose a distributed approach in which members contribute to the generation of group key by sending the hash of a random number during initialization phase and regenerate the group key themselves by obtaining the rekeying message from one of its members during rekeying phase or whenever membership changes occur. Symmetric key is used for communication between the members of a cluster and asymmetric key cryptography for distributing the rekeying messages to the members of the cluster.

3 PROPOSED SCHEME

3.1 System model

In order to make our concept scalable, to avoid expensive long-range traffic, and to enhance availability by providing service locally, we partition an ad hoc network into a number of clusters. The network layout is shown in Figure 1.In each cluster, exactly one distinguished node – the cluster head (CH) – is responsible for establishing and organizing the cluster. The cluster head is similar to other members with same computational capability as other members in the network. Gateways (GWs, which need not necessarily be CHs) manage communication with adjacent clusters. The CHs are responsible for sending CH beacons in their clusters, containing administrative information for the cluster members, e.g., lists of nodes and GWs in the cluster. Also, GWs periodically transmit GW beacons to inform their respective clusters about adjacent clusters.

Each member generates a public key, private key pair for secure communication between any pair of members. Public Key Cryptography forms a secure channel for communication between the nodes in the cluster. Each member contributes his share in arriving at a group key. Whenever membership changes occur, one of the nodes (the cluster head or any other node) sends a rekeying message to other nodes which aids in constructing the group key. This rekeying message is encrypted by the respective public keys of the other nodes so that they can use the private key to decode the message sent. Thus the burden on the cluster head is reduced and avoids the depletion of energy of the cluster head. Our scheme consists of two phases, initialization phase and group key agreement phase.

The various terms used in this paper and their meanings in the context of our discussion is given below

Cluster: The network topology is divided into small partitions with independent control. Every partition is a cluster.

Local node: It is only in a cluster and does not belong to any other cluster.

Adjacent nodes: A pair of nodes can communicate with each other in one hop.

Member: General term used to refer to a node including the cluster head and the gateway.

Gateway (relay node, or intermediate node): It belongs to multiple clusters or can directly communicate with the nodes of multiple clusters.

Neighbor clusters: Clusters have at least one same gateway.

We have made a few assumptions that are listed out here.

1. Each node in the mobile ad hoc network is assigned a unique integer valued id for identifying the node. The status code for the cluster heads is 1, for the members is 0 and the gateway is 2.

2. We assume that there is an offline authority that deploys the nodes into the network and issues a valid certificate binding the id with the public keys for authenticating the member, when it initially enters the network.

3. There are two types of keys that are used within a cluster, (i)Group key GK, used for encryption of messages exchanged within the cluster.

(i)Public key pk, of individual members used as Key encrypting key (KEK), for encrypting the rekeying messages generated whenever membership changes occur.

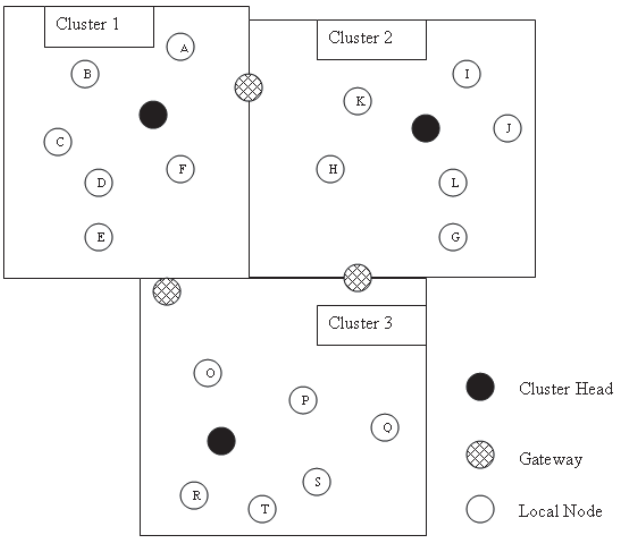

Figure 1. Network Layout

3.2 Initialization

Step 1: After deployment, the nodes broadcast their id value to their neighbors along with the HELLO message.

Step 2: When all the nodes have discovered their neighbors, they exchange information about the number of one hop neighbors. The node which has maximum one hop neighbors is selected as the cluster head. Other nodes become members of the cluster or local nodes. The nodes update the status values accordingly.

Step 3: The cluster head broadcasts the message “I am cluster head” so as to know its members.

Step 4: The members reply with the message “I am member” and in this way clusters are formed in the network.

Step 5: If a node receives more than one “I am cluster head” messages, it becomes Gateway which acts as a mediator between two clusters.

In this manner clusters are formed in the network. The nodes other than the cluster head and the gateway are the local nodes of the cluster.

3.3 Group Key Agreement

Step 1: Each member broadcasts the public key along with its id to all other members of the cluster along with the certificate for authentication.

Step 2: The members of the cluster generate the group key in a distributive manner. Each member generates a random number

and sends the hash of this number to the other members encrypted with the public key of the individual members, so that the remaining members can decrypt the message with their respective private key.

Step 3: Each member concatenates the hash values of the received members in the ascending order of the ids and mixes it using a one way hash function on the concatenated string. This is the group key used for that cluster.

Let HR_i be the hash of the random number generated by node i and GK denote the group key then

$GK=f(HR_1, HR_2, HR_3, ... HR_n)$

where

HR_i = hash(Random number i)

f is a one way function and

hash is secure hash function such as SHA1.

All the members now possess a copy of the same key as same operations are performed by all the nodes. The gateway nodes take part in the key agreement operations of both the clusters and store the group key of both the clusters.

Each member stores a group key, and public keys of other members in the cluster. The gate way nodes store 2 group keys of the clusters with which it is associated and the public keys of the members of both the clusters.

3.4 Communication Protocol

There are two types of communication that takes place in the network- intra cluster communication- between members of the same cluster and inter cluster communication-between members belonging two different clusters.

3.4.1 INTRA CLUSTER COMMUNICATION

The members within the cluster communicate using the group key. Suppose local node A wants to send a message to local node B, it encrypts the message with the group key GK1.

$A \rightarrow B: E_{GK1}[m]$

B decrypts the received message with the same group key GK1 in order to retrieve the message.

$B : D_{GK1}[E_{GK1}[m]]$

where E_{GK1} and D_{GK1} represent the encryption and decryption operations respectively using group key GK1

Thus symmetric key cryptography is used for data communication between the local nodes.

3.4.2 INTER CLUSTER COMMUNICATION

The members in one cluster communicate with the members in other cluster through the gate way nodes. Suppose member E in cluster 1 wants to communicate with member I in cluster2 it first sends the message to the gate way node encrypted with the group key GK1. The gate way node decrypts the message with GK1, and then encrypts the message using the group key GK2 of the destination cluster and transmits it to the destination. The destination node I then decrypts the received message with GK2.

and retrieves the data. This involves 2 encryptions and two decryption operations. Similarly, node S in cluster 3 sends the message through the intermediate node encrypted with the group key GK3 to the gateway node which decrypts the message with GK3 and again encrypts it with the group key GK1 and forwards to the destination node. This is illustrated in Figure 2. The arrow marks show the path of communication

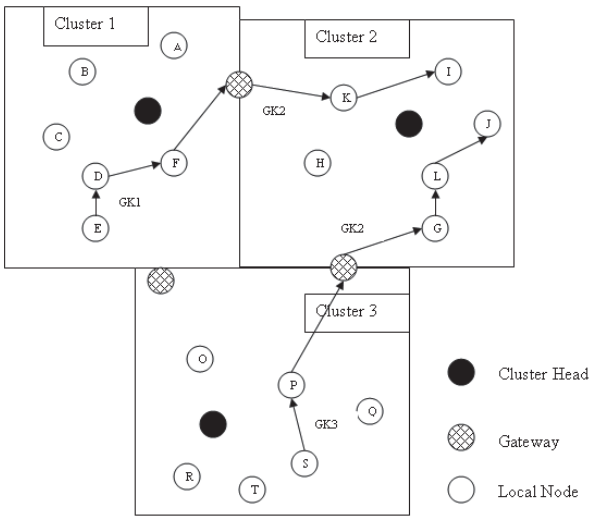

Figure 2 .Encryption and decryption of messages with cluster group keys.

3.5 Network dynamics

The number of nodes in the network is not necessarily fixed. New nodes may join the network or existing nodes may leave the network. In this section we address these issues. For cluster maintenance each cluster member needs to forward the new information to all other cluster members. If cluster member leaves the network, the trusted cluster member who identifies the change must inform this to all other members. A misbehaving node can also send a similar message to all other cluster members. But to prove the identity of the sender the trusted cluster member can encode the data using its private key. All other members maintain the public keys of other trusted members and the data could be decoded using the corresponding public key.

3.5.1 MEMBER JOINS

Step 1: When a new member joins the network, it sends a join request message to the adjacent node along with its public key and node id.

Step 2: After receiving the join request message and verification of authenticity, the adjacent node then sends the public key and id of the new node to all the old members encrypted with the old group key. All members update their existing member list.

Step 3: The adjacent node then generates two random numbers, (i) a 16 bit number indicating the position of insertion of the second random number and

(ii) 16 bit random number – the number that is to be inserted. This is sent to all the old members encrypted with the old group key.

Step 4: All members insert the random number received at the received position in the old group key and hash the result to get the 256 bit string which becomes the new group key for future communications

A simple example illustrating how to insert the random number at given position is given below.

Suppose

previous key=110101001010110011000100

random number=10100011

position =1011

new string=1101010010101000111011001100100

new key=hash(1101010010101000111011001100100)

This is the new group key generated.

Step 6: The adjacent node sends the new group key to the newly joined member encrypted with the new member's public key.

3.5.2 MEMBER LEAVES:

When a node leaves, there are three cases

(i) The local node leaves

(ii)The cluster head leaves

(iii)Gateway node leaves

Case (i): When the local node leaves

Step 1: Before leaving, the member informs the adjacent node that it is leaving.

Step 2: This node notifies the other members about the leaving member.

Step 3: The member nodes update the existing member list by deleting the corresponding public key entry from memory.

Step 4: The adjacent node generates a new 16 bit random number to be inserted and the number indicating the position where it has to be inserted and distributes it to the members using unicast message encrypted with the members' public key. This is required to ensure that the leaving member is not able to access the new messages.

Step 5: All the members regenerate the group key by inserting the received random number at the indicated position and hashing the result.

Case (ii): When the cluster head leaves:

Step 1: Before leaving the cluster, the cluster head sends a leave message to its own members. The nodes delete the entry corresponding to the cluster head from its memory.

Step 2: The cluster head delegates the role of the cluster head to its neighboring node and transfers all the information to this node.

Step 3: The new cluster head generates a new 16 bit random number to be inserted and the 16 bit number indicating the position of insertion and distributes it to the members using unicast message encrypted with the members' public key.

Step 4: All the members regenerate the group key by inserting the received random number at the received position and hashing the result.

Case (iii): When the gateway node leaves:

Step 1: The gate way node forms the link between two clusters and so the group key belonging to both the clusters need to be changed.

Step 2: The gateway node selects its neighbor node to be the gateway node and transfers the node ids, public keys and the hash value of the other cluster to this adjacent node.

Step 3: The gateway node then informs the cluster heads of both the clusters with which it is associated that it is leaving and also informs the cluster leaders about the appointment of the new gate way node.

Step 4: The gateway node also sends the public key of the new gate way node to the other cluster head.

Step 5: Both the cluster heads initiate the process of rekeying as indicated in case (ii).

4 PERFORMANCE ANALYSIS

4.1 Security Analysis

In this section we discuss the general attributes of key management system and how our scheme satisfies each one of these. Key management services should adhere to the following generic security attributes:

4.1.1 CONFIDENTIALITY

Key management schemes should guarantee key secrecy, that is, ensure the inability of adversaries or unauthorized parties to learn keying material (or even partial keying material).

We use hashing function to generate new keys from the old keys and due to the inherent property of one way hash function, it is not possible for an adversary to learn or predict the group key. Thus confidentiality is ensured.

4.1.2 BACKWARD SECRECY

A key management scheme with a backward secrecy property prevents an adversary from discovering preceding keys from a contiguous subset of new keys.

When a new member joins we ensure that it is not able to receive the previous information that was exchanged prior to it joining the network. In our scheme, the new group key is generated from the old group key by inserting a random number at a random position and hashing the result. The new group key is then sent to the newly joined member. Because of the one way property of the hashing function, the new node is not able to deduce the old group key and thus backward secrecy is maintained.

4.1.3 FORWARD SECRECY:

A key management scheme with a forward secrecy property prevents an adversary from discovering subsequent keys from a compromised contiguous subset of old keys

When a node leaves, we have to make sure that node is not able to receive any future information from the network. A new random number by the adjacent node and the position is generated and sent to the individual members in a secure manner. This allows the existing members to generate the new group key. Since the rekeying information is sent encrypted with the individual node's public key, the leaving member cannot obtain the random numbers sent and hence is not able to deduce the new group key. Therefore, forward secrecy is preserved.

4.1.4 KEY INDEPENDENCE

Key independence guarantees that a passive adversary who knows a proper subset of keys cannot discover any other keys Key independence subsumes forward and backward secrecy.

Since our scheme ensures forward and backward secrecy, it also ensures key independence.

4.1.5 AVAILABILITY

A high-availability feature prevents degradation of key management services and ensures that keying material is provided to nodes in the network when expected.

The keying material is available with all the nodes belonging to a group and this can be provided to any authentic node on request. Thus, our scheme also satisfies this property.

4.1.6 ROBUSTNESS

The key management scheme should tolerate hardware and software failures, asymmetric and unidirectional links, and network fragmentation/partitioning.

Our scheme does not rely on a central authority and the failure or compromise of one node will not affect the key management scheme. This is because, the contribution of each node to the generation of group key is considered only during the key agreement phase and not later.

4.1.7 SURVIVABILITY

Survivability is the capability of the key management service to remain available even in the presence of threats and failures. Survivability goes beyond security and fault tolerance to focus on the delivery of services, even when the system is partly compromised or experiences failures.

We have discussed in the previous section that the system does not depend on one particular node and thus it survives even in the event of compromise or failures.

4.1.8 EFFICIENCY

The key management service should be efficient with respect to communication, computational, memory, and energy resources.

Our scheme is efficient in all these aspects as it has less communication overhead, computational complexity and consumes less energy. This is discussed in detail in section 5.

4.1.9 SCALABILITY

Scalability ensures efficiency and availability as the number of networking nodes rapidly and significantly changes; the key management scheme should thus seamlessly scale to network size.

When the network size increases, by deploying a group of nodes in a particular locality, new clusters would be formed as discussed in the initialization phase. The nodes in the new cluster mutually agree on a group key as discussed in group key agreement phase. One of the nodes would initiate communication with the adjacent node already in the network and acts as a gateway node. Thus our scheme is scalable and tolerates single and multiple join.

4.1.10 RESISTANCE TO KNOWN KEY ATTACKS

A key management scheme is vulnerable to *known key attacks* (KKA) if a compromised *past* session key or subset of past session keys allows the following:

(1) a *passive* adversary to compromise future session keys and (2) an *active* adversary to *impersonate* other protocol participants

Our scheme provides resistance to the passive adversary from compromising future session keys with the knowledge of past session keys because all keying materials are encrypted before transmission.

Our protocol does not resist man in the middle attack. An active adversary can impersonate the existing members and send the joining or leaving information because nodes are not authenticated during communication with the public keys. However, this can be solved by the source node encrypting the message with its own private key after encrypting the message with the public key of the destination. Since all the nodes know the public key of the sender they can decode the messages. In case misbehavior of any node is observed, the cluster head takes action by eliminating it from the group.

4.2 Communication Cost Analysis

We compare the communication cost of our scheme with the other distributed schemes.

4.2.1 MEMBER JOINS

When a new member joins, the public key of the new member is broadcast to all old members encrypted with the old group key. Suppose the average number of members in a cluster is M, two 16 bit numbers or a message of 32 bits is transmitted to all the existing members encrypted with the old key. This scheme requires one round and 2 broadcast messages. The group keys of other clusters need not be changed.

Scheme	**No. of rounds**	**No. of messages**	
		Broadcast	Unicast
Burmester and Desmedt(BD)	3	2M	0
Group-Diffie Hellman(GDH)	M	M	M-1

Scheme	No. of rounds	No. of messages	
		Broadcast	Unicast
Distributed Logical Key Hierarchy (D-LKH)	3	1	M
Distributed One Way Function Trees(D-OFT)	Log_2 M	0	2Log_2 M
CRTDH	1	1	
Modified CRTDH	1	1	
Our scheme(join)	1	1	0
Our scheme(leave)	1	0	M-1
Gateway leave	1	0	2(M-1)

Table 1. Communication cost of rekeying

4.2.2 MEMBER LEAVES

When a node leaves, there are three cases

(i) The local node member leaves

(ii)The cluster head leaves

(iii)The gateway node leaves

(i)When the cluster member leaves, the random numbers are sent to the existing members encrypted with their respective public keys and unicast to the existing members. Therefore this requires one round and M-1 unicast messages.

Therefore, this requires one round in each cluster and M-1 unicast messages in each cluster that is a total of 2 (M-1) messages.

(ii)When the cluster head leaves, rekeying is similar to member leave because no extra updating is required.

(iii)When gateway node leaves, the group key of both the cluster with which it is associated have to change the group keys

Table 1 gives the communication cost of rekeying operation for various distributed schemes. From this table we observe that the rekeying procedure requires only one round in our scheme and CRTDH and modified CRTDH, in GD-H and BD it is a constant 3 whereas in other schemes such as D-LKH and D-OFT, it depends on the number of members. Regarding the number of messages sent, BD method involves 2M broadcast messages and no unicast messages, whereas in our technique, the number of unicast messages is M-1. We also observe that CRTDH has the least communication cost among all the methods, but it does not provide backward secrecy as even the leaving member can obtain the rekeying message since the message is broadcast and hence derive the new group key. Moreover, in our scheme the rekeying message is only 32 bits wide and thus the communication overhead is greatly reduced

4.3 Computation Complexity

Let

Exp=Exponential operation

D=Decryption operation

OWF=One Way Function

X=Exclusive OR operation

CRT=Chinese Remainder Theorem method for solving congruence relation

i= node id

M= Cluster size

LCM(N)=Least Common Multiple of N numbers

Sort=Sorting in ascending order

Scheme	During Set up phase		During rekey
	Cluster head	Members	
Burmester and Desmedt	(M+1)Exp	-----	(M+1)Exp
Group-Diffie Hellman	(i+1)Exp	------	(i+1)Exp
Distributed Logical Key Hierarchy	Log_2(MExp)	Log_2MD	Log_2MD
Distributed One Way Function Trees	(Log_2M+ 1) Exp	------	(Log_2M + 1)Exp
CRTDH	-----	LCM(M-1) + (M-1) X +MExp + CRT	LCM+X+CRT →leader CRT+X→members
Modified CRTDH	------	+ (M-1) X +MExp +CRT	X+CRT
Our scheme	Sort+ OWF	Sort+ OWF	D+OWF

Table 2. Computational Complexity

The computational complexity during the setup phase and rekeying operation of our scheme compared with the different distributed schemes discussed in Section 2 are given in Table 2.

4.4 Energy Analysis

The total energy consumed by a node for rekeying is the sum of the energy consumed by the node for transmission and reception of rekeying messages. We evaluate the total energy consumed by the node for communicating the rekeying messages for various message sizes. Since the energy required to transmit the rekeying message to a member depends on the network topology and path loss parameters of the medium, we introduce the average energy as a metric to evaluate the energy efficiency. We also examine the dependency of the average energy on the cluster size M. Since ad

hoc network is a multi hop network, the message from the source traverses through various intermediate nodes before reaching the destination. We show through simulations that the average energy consumed by the nodes is large for large message size. Since our scheme uses rekeying messages of 32 bits, less energy is consumed by the nodes for rekeying. Also the energy consumed by the nodes increases as cluster size increases. The simulation setup and results are discussed in the next section.

5 EXPERIMENTAL RESULTS

The simulations are performed using Network Simulator (NS-2.32) [17], particularly popular in the ad hoc networking community. The MAC layer protocol IEEE 802.11 is used in all simulations. The Ad Hoc On-demand Distance Vector (AODV) routing protocol is chosen for the simulations. Every simulation run is 500 seconds long. The simulation is carried out using different number of nodes. The simulation parameters are shown in Table 3.

Simulation time	1000 sec
Topology size	500m X 500m
Initial energy	100 Joules
Transmitter Power	0.4W
Receiver Power	0.3W
Node mobility	Max. speed 0m/s,5m/s, 10m/s, 20m/s
Routing Protocol	AODV
Traffic type	CBR, Message
MAC	IEEE 802.11
Mobility model	Random Waypoint
Max. no. of packets	10000
Pause time	200sec

Table 3. Simulation parameters

To evaluate the effect of various message sizes on the energy consumed by the various nodes during the set up phase, in the first experiment, we considered a cluster size of 8 nodes. Each node sends a message to every other node in the cluster. A total of 56 messages are exchanged between the nodes. The experiment is conducted with different mobility patterns generated using the setdest tool of ns2. These are stationary nodes located at random positions, nodes moving to random destinations with speeds varying between 0 and a maximum of 5m/s, 10m/s and 20m/s. The random waypoint mobility model is used in which the nodes move to a randomly selected position with the speed varying between 0 and maximum speed, pauses for a specified pause time and again starts moving with the same speed to a new destination. The pause time is set to 200 secs. Different message sizes of 16, 32, 48, 64, 128, 152, 180, 200, 256, 512 and 1024 bits are used. We observed that in all the four scenarios the energy consumed by the node increases as the message size increases. This is depicted by the column graphs in Figures.3, 4, 5, and 6. Each column in the graph represents the energy consumed by a node. There are 8 colums for each message size, representing the energy consumed by eight nodes. Since the nodes in a mobile ad hoc network communicate in a hop by hop manner, the energy consumed by all the nodes is not the same, even though same number of messages are sent and received by the nodes. This is clearly visible from the graphs.

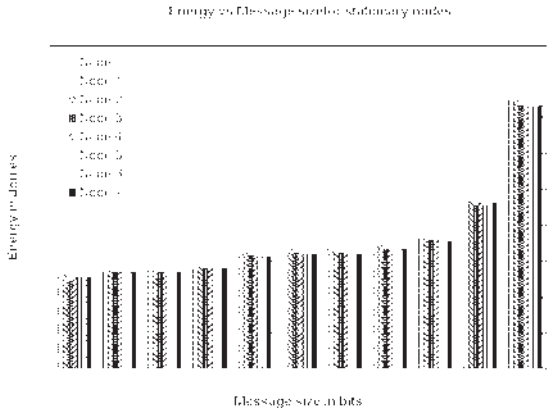

Figure 3. Energy consumed by the nodes vs. message size for stationary nodes when nodes transmit only messages

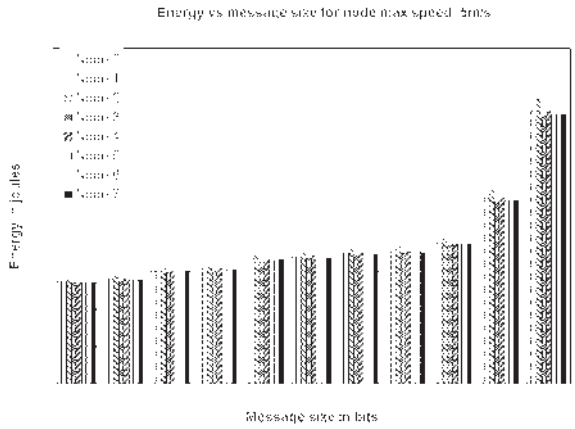

Figure 4. Energy consumed by the nodes vs. message size for mobility pattern with max. speed=5m/s when nodes transmit only messages

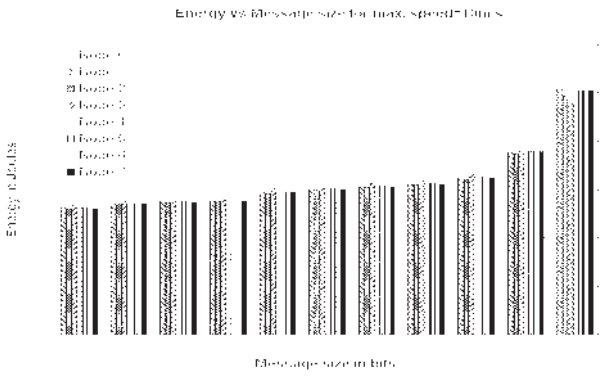

Figure 5. Energy consumed by the nodes vs. message size for mobility pattern with max. speed=10m/s when nodes transmit only messages

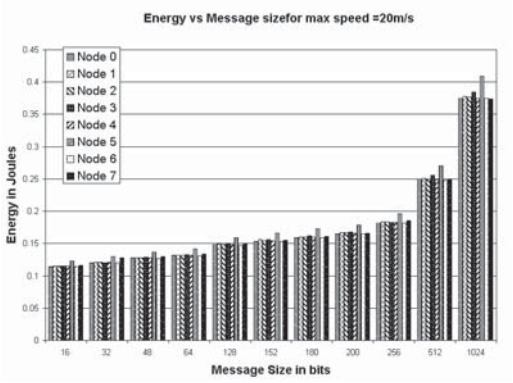

Figure 6. Energy consumed by the nodes vs. message size for mobility pattern with max. speed=20m/s when nodes transmit only messages

In the next experiment we considered the same topology with same number of nodes. But now in addition to the messages, the nodes transmit CBR traffic of maximum 10000 packets with packet size of 512 bits. The CBR traffic is generated with a maximum of 8 connections established between randomly selected pair of nodes. The traffic rate is set to 10 packets/sec. The graph of average energy consumed by all the nodes versus the message size in bits for different scenarios is shown in Figure 7. The reduction in energy is not clearly observed because of the large volume of CBR traffic generated. But, we observed that some of the node's energy got exhausted with increase in message size after which they were neither able to send nor receive any messages or traffic.

In the third experiment, we varied the cluster size and observed the effect of the cluster size on the average energy consumed by the nodes for communicating only rekeying messages. In this setup one node sends a message to every other node in the cluster. For M nodes, M-1 messages are exchanged. This is indicated in Figures 8, 9, 10 and 11. We observe that the energy consumed by the nodes increases as the network size increases and this is true with message sizes also as is given in Table 4.

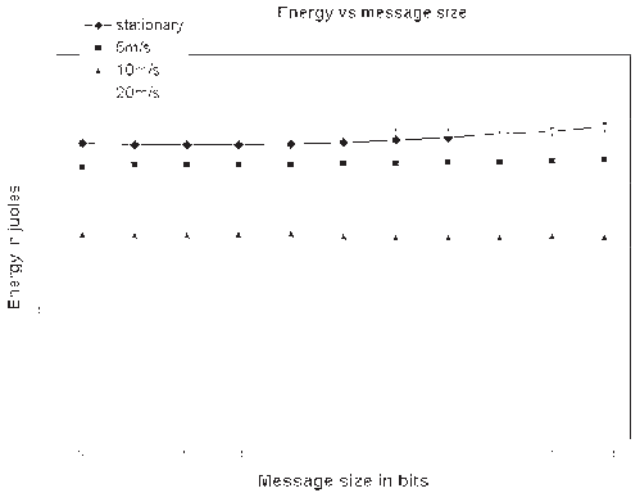

Figure 7. Average energy consumed by the nodes vs. message size for different mobility patterns when nodes transmit messages and CBR traffic

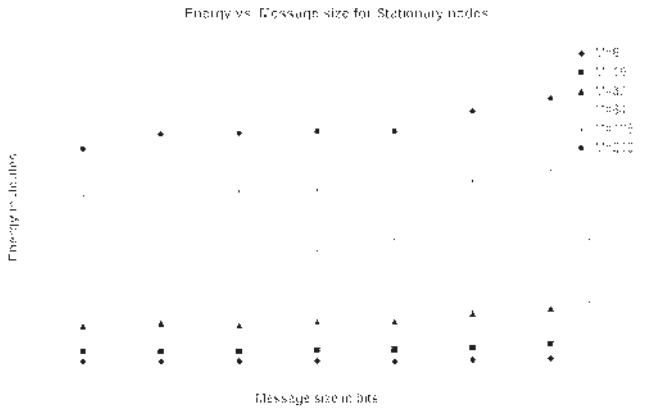

Figure 8. Average energy consumed by the nodes vs. message size for different cluster sizes with stationary nodes

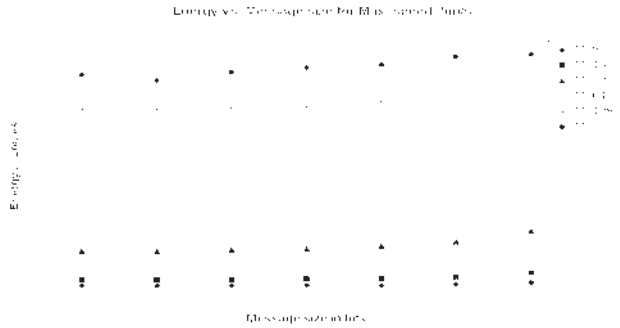

Figure 9. Average energy consumed by the nodes vs. message size for different cluster sizes with mobility pattern of max. speed=5m/s

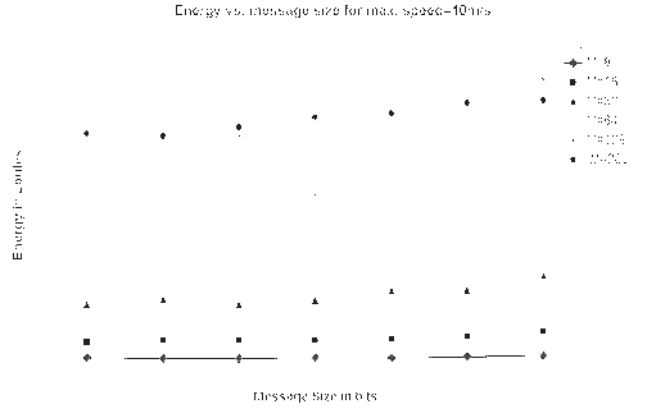

Figure 10. Average energy consumed by the nodes vs. message size for different cluster sizes with mobility pattern of max. speed=10m/s

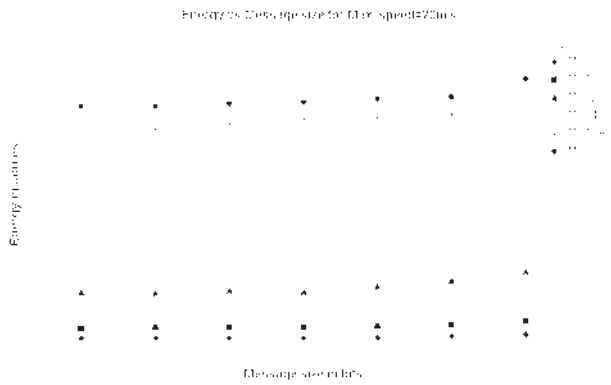

Figure 11. Average energy consumed by the nodes vs. message size for different cluster sizes with mobility pattern of max. speed=20m/s

Cluster Size	% Energy saved
8	41.5
16	39.4
32	30.4
64	25
128	20.5
200	14.75

Table 4: % Energy saved

6 CONCLUSION

We proposed an energy efficient scheme for group key management that does not rely on a centralized authority for regenerating a new group key. Any node can initiate the process of rekeying and so the energy depletion of any one particular node is eliminated unlike the centralized schemes. Our approach satisfies most of the security attributes of a key management system. The communication and computational overhead is small in our scheme compared with other distributed schemes. The energy saving is approximately 41% for 8 nodes and 15% for 200 nodes when the message size is reduced from 1024 to 16 bits. This indicates that small message size and small cluster size is most suitable for energy limited mobile ad hoc networks.

7 REFERENCES

[1] Wallner, D.M., Harder, E.J. and Agee, R.C. (1998) "Key management for multicast: issues and architectures", Internet Draft, draft-wallner-key-arch-01.txt

[2] Sherman, A.T. and McGrew, D.A. (2003) "Key establishment in large dynamic groups using one-way function trees", IEEE Transactions on Software Engineering, Vol. 29, No. 5, pp.444–458

[3] S. Mittra. Iolus: "A framework for scalable secure multicasting", Journal of Computer Communication Reviews, 27(4):277–288, 1997.

[4] Y. Kim, A. Perrig, and G. Tsudik., "Tree-based group key agreement. ACM Transactions on Information Systems Security", 7(1):60–96, Feb. 2004.

[5] 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.

[6] M. Burmester and Y. Desmedt,. "A secure and efficient conference key distribution system" In Advances in Cryptology - EUROCRYPT, 1994.

[7] P. Adusumilli, X. Zou, and B. Ramamurthy, "DGKD: Distributed group key distribution with authentication capability" Proceedings of 2005 IEEEWorkshop on Information Assurance and Security, West Point, NY, USA, pages 476–481, June 2005.

[8] J.-H. Huang and S. Mishra, "Mykil: a highly scalable key distribution protocol for large group multicast", IEEE Global Telecommunications Conference, (GLOBECOM), 3:1476–1480, 2003.

[9] B. Wu, J. Wu, E. B. Fernandez, M. Ilyas, and S. Magliveras, "Secure and efficient key management in mobile ad hoc networks" Journal of Network and Computer Applications, 30(3):937–954, 2007.

[10] Z. Yu and Y. Guan., "A key pre-distribution scheme using deployment knowledge for wireless sensor networks" Proceedings of the 4th ACM/IEEE International Conference onInformation Processing in Sensor Networks (IPSN), pages 261–268, 2005.

[11] Bing Wu, Jie Wuand Yuhong Dong, "An efficient group key management scheme for mobile ad hoc networks", Int. J. Security and Networks, 2008.

[12] MS. Bouassida, I. Chrisment, and 0. Festor , "A Group Key Management in MANETs. in International Journal of Network Security", Vol.6, No. 1, PP.67-79, Jan. 2008

[13] Dondeti L., Mukherjee S., and Samal A. 1999a. "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.

[14] Kim Y., Perrig, A., And Tsudik G. 2000, "Simple and fault-tolerant key agreement for dynamic collaborative groups", In Proceedings of the 7th ACM Conference in Computer and Communication Security, (Athens, Greece Nov.). (S. Jajodia and P. Samarati, Eds.), pp. 235–241.

[15] 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.

[16] Spyros Magliveras and Wandi Wei Xukai Zou, "Notes on the CRTDH Group Key Agreement Protocol" The 28th International Conference on Distributed Computing Systems Workshops 2008

[17] The network simulator, http://www.isi.nsnam/ns

Author Biographies

Renuka A. is a Reader in the Dept. of Computer Science and Engg., Manipal Institute of Technology, Manipal. She obtained her B.E. degree in 1991 and M.Tech. degree in the year 1998 from Mysore University. Presently she is pursuing Ph.D. in Computer Engineering, at National Institute of Technology, Karnataka, India. She has published papers in refereed international conferences and journals. Her research interests include Mobile Ad hoc Networks, Image Processing, Computer Architecture. and O.S. .

Dr. K. C. SHET is a Professor in the Dept. of Computer Engineering, National Institute of Technology Karnataka, India. He has more than 36 years of experience in teaching and research. He holds a Ph. D. from IIT Bombay, India. He is a member of Computer Society of India, and Indian Society of Technical Education. He is a Fellow of Institution of Engineers (INDIA). His research interests include software testing, Security Solution for Web Services, Cyber Laws, Anti spam solutions, Wireless Networks, Mobile Computing, Ad hoc Networks. He has published more than 210 papers in refereed conference proceedings and journals.