Key Distribution and Update for Secure Inter-group Multicast Communication

Weichao Wang
Department of EECS and ITTC
University of Kansas
Lawrence, KS 66045
weichaow@eecs.ku.edu

Bharat Bhargava
CERIAS and Department of CS
Purdue University
West Lafayette, IN 47907
bb@cs.purdue.edu

ABSTRACT

Group communication has become an important component in wireless networks. In this paper, we focus on the environments in which multiple groups coexist in the system, and both intra and inter group multicast traffic must be protected by secret keys. We propose a mechanism that integrates polynomials with flat tables to achieve personal key share distribution and efficient key refreshment during group changes. The proposed mechanism distributes keys via true broadcast. The contributions of the research include: (1) By switching from asymmetric algorithms to symmetric encryption methods, the proposed mechanism avoids heavy computation, and improves the processing efficiency of multicast traffic and the power usage at the wireless nodes. The *group managers* do not have to generate public-private key pairs when the group member changes. (2) It becomes more difficult for an attacker to impersonate another node since personal key shares are adopted. The additional storage overhead at the wireless nodes and the increased broadcast traffic during key refreshment are justified. In addition, we describe techniques to improve the robustness of the proposed mechanism under the complicated scenarios such as collusive attacks and batch group member changes.

Categories and Subject Descriptors

C.2.0 [**Computer Systems Organization**]: Computer-Communication Networks – *General - Security and protection*

General Terms

Design, Security

Keywords

Security, Inter-group Communication, Key Distribution and Update

1. INTRODUCTION

Group communication has become an important component of many applications in wireless networks. It takes advantage of the broadcast characteristic of wireless communication to accelerate the information propagation speed and improve the energy efficiency at mobile nodes. As an example, traditional multicast, stateless multicast, and overlay multicast protocols have been developed for ad hoc networks. A good review can be found in [44]. To prevent the attackers from paralyzing the network and services by manipulating the multicast packets, secret keys must be distributed and properly maintained throughout the lifetime of the network. Therefore, key establishment and refreshment becomes a critical problem for protecting the environments and must be paid special attention.

In this paper, we focus on the problem of key distribution and update for secure inter-group communication. There are various applications in which the mobile nodes are divided into multiple groups and multicast traffic exists both within the same group and between different groups. Below we give two examples:

In a location based service system [4, 5], the geographical positions of the mobile nodes are shared so that the packets can be routed and the services can be provided. In some applications, the mobile nodes will be divided into different groups based on their security levels and the accuracy of the location information that a node can acquire is determined by this level. To support such requirements, a node needs to encrypt its position information at different accuracy levels with different keys. Secure inter-group communication is expected in the environment: only the members in the target group will be able to recover the position information and all other nodes should not get access.

Another example comes from the security-aware ad hoc routing for wireless networks [1]. The users are divided into three groups according to their ranks: soldiers, officers, and generals. There exist routing requests and multicast traffic between the nodes in different groups, for example, a soldier may send a message that can be read only by all of the generals. Secret keys must be deployed to restrict the nodes that can recover the information and participate in the operations.

Enforcing security in such environments puts new challenges to the researchers. It is different from secure multicast because it involves both intra-group and inter-group communication. It is also different from the pair-wise key establishment or pre-distribution methods. The member changes among groups will bring new difficulties to key management.

A new approach that supports efficient key distribution and update is required to protect the traffic in these applications.

A straight forward solution is to deploy a public-private key pair for every group. Every node knows all the public keys and only the private key of the group that it belongs to. For example, for the application in [1], a soldier will know Pub_{sol}, Pub_{off}, Pub_{gen}, and Pri_{sol}. When he wants to send a message that can only be read by the generals, he can use the Pub_{gen} to encrypt the information.

This approach is simple, yet with three major disadvantages: (1) The public-private key encryption involves exponential computation, which is not efficient for a wireless node when its limited energy and computation capability is considered. (2) When the security level of a mobile node changes or a compromised node is detected and expelled from the current group, the secret keys must be updated. The *group manager* will be overwhelmed by the computation overhead for generating secure public-private key pairs when such changes happen frequently. (3) Since the public keys are known to every node, we cannot determine the identity of the sender based on the encrypted message unless additional authentication methods are adopted. An attacker can easily impersonate another node and this will put a challenge to the procedure of locating the sources of attacks.

In this paper, we propose a new mechanism to overcome these difficulties. First, symmetric keys are used to protect the multicast traffic in the same group. At the same time, polynomials are adopted to determine the keys to protect inter-group communication. We calculate the personal key share of a node by applying its unique ID to the polynomial. When a node changes its group, we adopt the flat tables proposed in [2, 3] to distribute keys via true broadcast and the refreshment of the personal key shares is conducted in a distributed manner.

The contributions of the proposed mechanism are: (1) By switching from asymmetric algorithms to symmetric encryption methods, the proposed mechanism avoids heavy computation. It improves the processing efficiency of the multicast traffic and the power usage at the wireless nodes. (2) It becomes more difficult for an attacker to impersonate another node when the personal key shares are deployed. The costs for these advantages include more keys to be stored by the wireless nodes and more broadcast traffic to be initiated during key refreshment, which are justified in section 6. In addition to these contributions, several other features make the proposed mechanism more attractive to wireless environments. First, the update of the personal key shares is conducted in a distributed way, which reduces the communication overhead for control. Second, compared to the key organization structures such as Logical Key Hierarchy (LKH), the adoption of flat tables can reduce the key storage overhead at the *group managers*. It helps to reduce the unfairness on resource consumption in a self-organized environment such as ad hoc networks.

The remainder of the paper is organized as follows: In section 2, we review the previous research efforts that contribute to our approach. Section 3 presents the assumptions and models of the system. Section 4 describes how secure intra group and inter group communication is achieved. In section 5, we describe the key update operations in detail when a node joins or leaves a group. Forward and backward secrecy are enforced. Section 6 investigates the overhead and robustness of the proposed mechanism. The advantages and disadvantages of several other potential key management schemes for secure inter-group communication are discussed as well. It then presents the future extensions. Finally, section 7 concludes the paper.

2. RELATED WORK

Key management for secure group communication has attracted a lot of research efforts and very encouraging results have been collected. Below we summarize some of the previous approaches.

In the early solutions such as Group Key Management Protocol (GKMP) [6], the centralized controller will distribute a key encryption key (KEK) and a traffic encryption key (TEK) to a node when it joins the group. These one-to-one distribution mechanisms do not scale to large networks.

To address the scalability problem, the members of a multicast group have been organized into a hierarchy. Every node is treated as a leaf and it holds all the keys from the leaf to the root. This Logical Key Hierarchy [7, 8] also reduces the size of the rekeying message. Various approaches have been proposed to improve the method by reducing the number of keys stored at the group members, reducing the broadcast traffic during key refreshment, and supporting forward and backward secrecy. The adopted methods include using one way functions to lessen the key distribution overhead when a node joins the group [2, 9, 10], using a-ary to reduce the tree size [11], using flat tables to reduce the keys held by the KDC [2], and using pseudo random functions to build and manipulate the keys in the hierarchical tree [12].

To avoid the single point of failure and to restrict the impacts of a group member change, several mechanisms have been developed to divide the nodes into multiple subgroups. In Iolus [13] each subgroup uses an independent key and the agents of the subgroups form a top-level management team. The separation of the encryption keys in different groups enables the membership changes to be handled locally. The disadvantage is that the inter-subgroup traffic must be translated by the agents. Dual encryption protocol [14] has been proposed to deal with the trust of the third parties. Cipher sequences [15] have been integrated into the subgroups to improve the efficiency of key distribution and update. A synchronized group key distribution protocol is adopted by Hydra [16] to achieve key refreshment when a membership change in a subgroup happens.

In several mechanisms the keys are updated as a function of time. For example, in [17], the short slices of time are organized as a tree and every slice uses a different key. Every node will receive the decryption keys corresponding to the time duration in which it is a legal group member so that the access to the traffic is granted. The approaches such as Kronos [18] will periodically rekey the group and they provide an efficient solution for the environments in which the membership changes happen very frequently.

Various approaches have been proposed to improve the efficiency and security of group communication in wireless networks. They target at the special features such as node mobility and frequent link changes. The limited resources on computation capability, energy, and available bandwidth are also considered. LKHW [19] extends the application of Logical Key Hierarchy to sensor networks and it enforces both backward and forward secrecy. In [20], a node will join a multicast group by attaching to the closest member so that a physical security tree structure is constructed. The joining and leaving operations are managed by the upstream node in the tree structure. The research efforts in [21, 22] consider the location information and different models of signal attenuation when constructing the multicast hierarchy so that a better energy efficiency can be achieved. To reduce the maintenance overhead of the forwarding state in wireless nodes, stateless multicast protocols [45, 46] and overlay multicast protocols [47, 48] have been developed for ad hoc networks.

Since the schemes such as Diffie-Hellman and the public key infrastructure involve exponential computation, the mechanisms that adopt them [23, 24] will put a severe challenge to the computation capability of the mobile nodes. The researchers have integrated trust with secure group communication and proposed several approaches [25, 26]. The mobile nodes are divided into different areas or clusters and the key distribution and revocation methods under various trust models are studied.

Both CKDS [27] and GKMPAN [28] avoid the adoption of LKH. CKDS uses a matrix-like key distribution structure in which the unknown secrets to the revoked nodes can be used to distribute the new keys. GKMPAN depends on TESLA for the authentication of the multicast packets and the group key updates. It assumes high node mobility and provides the desirable *stateless property*, which allows the mobile nodes that miss the rekeying procedure due to network partition to recover the current group keys.

Polynomial interpolation was first used to implement threshold secret sharing [29]. It allows a dealer to distribute a secret s to n players and at least $t < n$ players are required to recover the information. Staddon *et al* [30] proposed a self-healing key distribution mechanism with revocation capability based on the secret sharing techniques. The users are capable of recovering the lost group keys without interacting with the *group manager*. The *manager* uses a bivariate polynomial as a masking function to privately transmit information to group members. Liu *et al* [31] proposed an efficient self healing group key distribution scheme with revocation capability based on the result. A novel personal key distribution scheme is developed and the storage and communication overhead is reduced. More *et al* [33] have improved their previous result by applying sliding window to the self-healing procedure so that more consistent robustness and less overhead can be achieved.

3. OUR MODEL

Network and Communication Model

We assume that the links among wireless nodes are bidirectional and two neighboring nodes can always send packets to each other. This assumption will hold under most conditions when the power of the nodes has not been exhausted.

We adopt a simplified model to describe the intra and inter group communication. We assume that the nodes are divided into multiple groups and secret keys are deployed to control the access to the multicast packets, whose target could be the members in the same group or in a different group. A node may change its group as time passes by and new members can join the network dynamically. The nodes that are compromised by the attackers will be expelled from the network when they are detected. We assume that the multicast data packets have a much higher frequency than group member changes and they explain a majority of the computation and communication overhead caused by multicasting operations. This model is powerful enough to describe the applications in section 1, and a lenient space has been left for future extensions.

Secret keys must be deployed to protect the multicast traffic so that only the group members with valid keys can send out the messages and get access to the encrypted information. For the simplicity of the presentation, we assume that a centralized *group manager GM* is in charge of key distribution and update for all different groups. Distributed administration through multiple *managers* will be discussed in section 6 to improve the robustness of the proposed mechanism. A multicast packet can be forwarded by both the members in the target group and the nodes in other groups.

Threat Model

The security threats to wireless networks come from all layers. The malicious nodes can jam the physical layer. There have been approaches using spread spectrum [34] to provide resistance to such attacks. There are also Deny of Service (DoS) attacks on the medium access control layer [35]. For example, if a malicious node keeps sending noises and causes collisions, the communication within the neighborhood will be paralyzed. The fairness control mechanisms such as time division multiple access [36] can avoid one attacker consuming all available bandwidth. This paper will not discuss solutions to these attacks.

We assume that the malicious nodes can eavesdrop and record the packets that are transmitted over the wireless medium. They can also conduct active attacks by inserting, modifying, or discarding packets. We assume that the malicious nodes do not have the computation resources to directly break the encryption keys.

When a node changes its group, new keys must be generated to replace the old secrets held by it. During these updates, two features must be enforced by the key management scheme as described in [12, 37]: *forward* and *backward secrecy*. *Forward secrecy* guarantees that when a node is expelled from a group, it cannot discover the subsequent keys based on the knowledge of the old ones. *Backward secrecy* guarantees that when a node joins a group, it cannot discover the old keys based on its current knowledge and get access to previous traffic. The two features together will prevent information leakage in the highly dynamic applications.

Notations

We assume that every node is uniquely identified by a node ID i, where $i \in \{1 \cdots n\}$ and n is the total number of nodes. The nodes are divided into m different groups, which are represented by G_1 to G_m respectively. All operations described in the protocol will take place in a finite field F_q, where q is a prime number with a large enough value.

We use $E_k(msg)$ and $D_k(msg)$ to represent the encryption and decryption of the message msg with a symmetric key k respectively. $H(\cdot)$ represents a hash function that is known to all nodes in the network and $H(msg)$ is a hash result. We use $h(x)$ to represent a t-degree polynomial in $F_q[x]$ and $h(i)$ is the value of the function at point i. We use $S_{GM}(msg)$ to represent the digital signature of the *group manager* on the message and every node in the network can verify this signature. We use r to represent the number of bits that are required to record a node ID, where $r = \lceil log_2(n) \rceil$. The bit values of node i's ID can be represented as $i_1, \cdots, i_r$. k' represents the new key that is adopted to replace the old secret

k. We assume that the mobile nodes use random numbers to protect the freshness of the traffic. These random numbers can be generated off-line and stored at the nodes to avoid the computation overhead.

We assume that a packet has the format (*sender*, *receiver*, *objective*, *data contents*, *integrity protection*). The *group manager* is represented by GM in a packet. If a packet has a group name as the *receiver*, it is a multicast message that targets at all current members of the group.

4. SECURE GROUP COMMUNICATION

During the network initiation procedure, every node will get a set of secret keys from the centralized *manager* through a secure channel such as the physical contact before deployment. These keys are divided into two groups: traffic encryption keys (TEK) to protect the group communication packets, and key encryption keys (KEK) to support secret refreshment. Without losing generality, we assume that the nodes are divided into three groups G_1, G_2, and G_3. Below we use a node i in group G_2 as an example to illustrate the secret keys that it holds.

We assume that node i can communicate with the *group manager* securely. This can be achieved through a pairwise key K_{i-GM} shared between the two entities. As a member of G_2, i will get a copy of the symmetric group key K_2 which is used to encrypt and decrypt the multicast traffic within the group.

We use the t-degree polynomials $h(x)$ to determine the personal key shares and protect the inter group traffic. As a member of G_2, i must be able to recover the multicast packets sent by the nodes in G_1 and G_3. Therefore, it will be aware of two such functions, $h_{21}(x)$ and $h_{23}(x)$. $h_{21}(x)$ represents the polynomial which determines the personal key shares of the members in G_1 to send multicast packets to the members in G_2. A node v in G_1 will get its share $h_{21}(v)$ from the *group manager*. When it wants to send a multicast packet msg to the members in G_2, it will send out $(v, G_2, E_{h_{21}(v)}(msg, H(msg)))$. Since every node in G_2 knows $h_{21}(x)$, it can calculate the personal key share $h_{21}(v)$ by applying v to the polynomial and recover the information. Similarly, i is aware of the polynomial $h_{23}(x)$ so that it can decrypt the multicast messages from the members in G_3. To enable node i to send multicast packets to the members in G_1 and G_3, it will get two personal key shares $h_{12}(i)$ and $h_{32}(i)$ from the *group manager*.

Two advantages have been brought by the personal key shares determined by the polynomials. First, for two different nodes v and w in G_1, they will have different personal keys $h_{21}(v)$ and $h_{21}(w)$ to encrypt the multicast packets to G_2. Therefore, information isolation has been achieved, and only the sender and the members in the target group can recover the packet. Second, it becomes more difficult for an attacker to impersonate another node in the same group unless it can collect $t+1$ personal keys and reconstruct the polynomial. This proof of identity is especially useful in the environments when the members in one group have less trust on the members in another group.

We have described the TEKs of the proposed mechanism and now we introduce how the secret keys can be refreshed using flat tables. Every group in the network has its own flat table. Since r bits are required to represent a node ID, a flat table will consist of $2r$ keys, with one key associated with every possible value of a bit. For example, the flat table of G_2 includes the following keys: ($z_{1.0}$, $z_{1.1}$, $z_{2.0}$, $z_{2.1}$, $\cdots$, $z_{r.0}$, $z_{r.1}$), where the first subindex is the position of the bit, and the second index represents the binary value. Every node will get the keys from the *group manager* that are associated with the values of the bits in its node ID. For example, if $r = 4$, a node with ID 10 can be represented as $(1010)_2$ in binary, and it will have the keys $z_{1.1}$, $z_{2.0}$, $z_{3.1}$, and $z_{4.0}$ from the table.

Every node will have exactly a half of the keys in the flat table for its group. If the bit values of node i's ID are represented as (i_1, $\cdots$, i_r), it will have the keys ($z_{1.i_1}$, $z_{2.i_2}$, $\cdots$, $z_{r.i_r}$) from the flat table, and the other half of the keys can be represented as ($z_{1.\overline{i_1}}$, $z_{2.\overline{i_2}}$, $\cdots$, $z_{r.\overline{i_r}}$). Since every node has a unique ID, every pair of nodes in the same group must have at least one key from the flat table that is different. This feature has brought two advantages to the mechanism. First, if the *manager* sends a message as $E_{z_{1.i_1}} E_{z_{2.i_2}} \cdots E_{z_{r.i_r}}(msg)$, only i has all the keys to decrypt the packet. The other nodes in the same group, unless colluding with some other members, do not have all the keys to recover the information. Second, if the *manager* wants to send a message to all the members but node i, it can broadcast ($E_{z_{1.\overline{i_1}}}(msg)$, $E_{z_{2.\overline{i_2}}}(msg)$, $\cdots$, $E_{z_{r.\overline{i_r}}}(msg)$). Since every node but i must have at least one of the keys, it can recover the information. A key update can be realized by this scheme when node i leaves the group.

The following table summarizes the secret keys that are held by node i and their usage. We assume that i is a member of G_2.

Table 1: Secret keys held by i and their usage.

Secret keys	Usage
K_{i-GM}	pairwise key shared between i and the *group manager*
K_2	group key shared by members of G_2
$h_{21}(x)$	polynomial to determine the keys for decrypting the multicast traffic from a node in G_1
$h_{23}(x)$	polynomial to determine the keys for decrypting the multicast traffic from a node in G_3
$h_{12}(i)$	personal key share to encrypt multicast traffic sent to the members of G_1
$h_{32}(i)$	personal key share to encrypt multicast traffic sent to the members of G_3
$z_{1.i_1}$ $\cdots$ $z_{r.i_r}$	keys in flat table for key update

5. KEY UPDATE DURING GROUP CHANGES

When a group change happens, the corresponding keys must be updated to enforce forward and backward secrecy. In this section, we present the details of the key update operations and illustrate how the new keys can be established efficiently.

5.1 Joining operations

Without losing generality, we assume that a node i who does not belong to any group wants to join the group G_1. To enforce backward secrecy, the following steps must be adopted:

1. The current members of G_1 have been using K_1 to encrypt the multicast traffic within the group. To pre-

vent node i from getting access to the previous information, the new group key K'_1 must be established. The *group manager* will broadcast the packet:

$$(\quad GM, G_1, group\ key\ update\ for\ G_1, \\ E_{K_1}(GM, G_1, H(K_1), K'_1), \\ S_{GM}(GM, G_1, H(GM, G_1, K_1, K'_1)) \ )$$

Since only the current members of G_1 have K_1, they can decrypt the packet and recover the new key K'_1. The hash result of the new group key is protected by the digital signature of the *manager* so that we can prevent a malicious node from sending a random number in the packet to distribute false keys.

2. The polynomials $h_{21}(x)$ and $h_{31}(x)$ determine the personal key shares of a node in G_1 that are used to encrypt the multicast traffic to the members in G_2 and G_3. We argue that these two functions do not have to change. Since node i will only get its key shares $h_{21}(i)$ and $h_{31}(i)$ from the *group manager*, it will not be able to reconstruct the t-degree polynomials and it cannot calculate the key shares of the other nodes in G_1. Therefore, the previous multicast traffic from G_1 to G_2 and G_3 is still safe.

3. To establish the new flat table for group G_1, the *group manager* can broadcast the following message:

$$(\quad GM,\ G_1,\ flat\ table\ update\ for\ G_1, \\ E_{K_1}E_{z_{1.0}}(z'_{1.0}),\ E_{K_1}E_{z_{1.1}}(z'_{1.1}), \cdots\cdots, \\ E_{K_1}E_{z_{r.0}}(z'_{r.0}),\ E_{K_1}E_{z_{r.1}}(z'_{r.1}), \\ S_{GM}(GM, G_1, H(z_{1.0}, z'_{1.0}), \cdots, H(z_{r.1}, z'_{r.1})) \ )$$

Every key in the new flat table will be encrypted with the old group key K_1 and the corresponding key in the old flat table. Since only the members of G_1 have the key K_1, they can open the first level of encryption. Then a node can get an entry in the new flat table only if it has the old key at the same position.

4. The polynomials $h_{12}(x)$ and $h_{13}(x)$ must be updated to protect the multicast traffic from G_2 and G_3 that is transferred before i joins the group. Below we describe how the new functions can be distributed to the members in G_1 and in step 5 we discuss how the nodes in G_2 and G_3 can get their refreshed personal key shares in a distributed manner.

   The *group manager* will choose two t-degree polynomials from $F_q[x]$ as the $h'_{12}(x)$ and $h'_{13}(x)$. It will then broadcast the packet:

$$(\quad GM,\ G_1,\ polynomial\ update\ for\ G_1, \\ E_{K_1}(GM, G_1, H(h_{12}(x), h_{13}(x)), h'_{12}(x), h'_{13}(x)), \\ S_{GM}(GM, G_1, H(h_{12}(x), h_{13}(x), h'_{12}(x), h'_{13}(x))) \ )$$

   Since only the members of G_1 have K_1, they can decrypt the packet and recover the new polynomials.

5. With the distribution of $h'_{12}(x)$ and $h'_{13}(x)$, the personal key shares of the nodes in G_2 and G_3 must be updated as well. Although the *group manager* can encrypt each of the new key shares with the corresponding old secret and broadcast the packet throughout the network, it is not an efficient solution. First, different from the size of a flat table, which only contains $2r$ keys, the groups G_2 and G_3 can contain up to $O(n)$ nodes altogether. Therefore, the message can be too long to fit in one packet and multiple rounds of broadcast must be activated, which will cause too much communication overhead. Second, the messages will unnecessarily reveal information about the new polynomials. When the limited computation and communication resources of a wireless node are considered, a more efficient approach is expected.

   We propose a distributed mechanism to solve this problem. Since every node in G_1 has already got the new polynomials $h'_{12}(x)$ and $h'_{13}(x)$ from the *group manager*, the members of G_2 and G_3 can locate such a node in the neighborhood and acquire their new key shares from it. In the following description, we use a node v in G_2 and w in G_1 as an example.

   (1) The *group manager* will broadcast an authenticated message and notify all nodes in G_2 and G_3 to acquire the new personal key shares.

   (2) After verifying the packet from the *manager*, v will initiate a local broadcast within a few hops and locate a node w from the group G_1. v will then acquire the new key share $h'_{12}(v)$ from w as follows:

$$v \rightarrow w: \ (\ v, w,\ request\ for\ h'_{12}(v), \\ E_{h_{12}(v)}E_{h_{21}(w)}(v, w, H(h_{12}(v), h_{21}(w)), R) \ ) \\ w \rightarrow v: \ (\ w, v,\ reply\ for\ h'_{12}(v), \\ E_{h_{12}(v)}E_{h_{21}(w)}(w, v, h'_{12}(v), H(R, h_{12}(v), h'_{12}(v))) \ )$$

   R is a random number generated by v to guarantee the freshness of the reply. The set of nodes that know $h_{12}(v)$ include all members of G_1 and v, while the set of nodes that know $h_{21}(w)$ include all members of G_2 and w. The intersection of the two sets only include the nodes v and w. Therefore, $E_{h_{12}(v)}E_{h_{21}(w)}(\cdot)$ can be viewed as a secure channel between these two nodes. The hash result of the personal key shares and the random number is attached to the packet to prevent a malicious node from sending a random number in the packet to distribute false keys. We do not assume that v and w are direct neighbors and this procedure can be conducted through a multi-hop path. The other nodes in G_2 and G_3 can acquire the personal key shares in a similar way.

6. After the key update operations, the *group manager* will distribute the keys to node i through a secure channel using the pairwise key K_{i-GM}.

From the above description, we find that when a node joins a group, the keys are refreshed either through true broadcast or in a distributed manner, which fits to the characteristics of wireless networks and only a limited amount of computation overhead will be increased. More discussions of the overhead and the safety of the proposed mechanism will be presented in section 6.

5.2 Leaving operations

In this part, we are going to describe the key update operations when a node leaves a group. The leaving action may happen voluntarily or when a compromised node is detected and expelled from a group. Either way, the keys must be updated to enforce forward secrecy. As an example, we will study the case when node i leaves group G_2. The following steps will be adopted:

1. Node i should not get access to the multicast traffic in G_2 after it leaves the group. Therefore, the group key K_2 must be replaced by the new secret K_2'. Since i holds the keys $z_{1.i_1}$, $z_{2.i_2}$, $\cdots$, $z_{r.i_r}$ from the flat table, the *group manager* can broadcast the following packet to distribute the new group key:

$$\begin{aligned}(\quad & GM, G_2, \mathit{group\ key\ update\ for}\ G_2, \\ & E_{K_2}(E_{z_{1.\overline{i_1}}}(K_2'), E_{z_{2.\overline{i_2}}}(K_2'), \cdots, E_{z_{r.\overline{i_r}}}(K_2')), \\ & S_{GM}(GM, G_2, H(GM, G_2, K_2, K_2')) \quad)\end{aligned}$$

   Only the current members of G_2 and i can open the first level of encryption. Since a node can be uniquely identified by its ID, every member in G_2 but i must have at least one of the keys $z_{1.\overline{i_1}}$, $z_{2.\overline{i_2}}$, $\cdots$, $z_{r.\overline{i_r}}$. Therefore, it can open the second level of encryption and recover the new group key. After receiving this message, the remaining nodes in G_2 have a secure channel to communicate with each other.

2. For the remaining nodes in G_2, the *group manager* can establish the new flat table by broadcasting:

$$\begin{aligned}(\quad & GM, G_2, \mathit{flat\ table\ update\ for}\ G_2, \\ & E_{K_2'}E_{z_{1.0}}(z_{1.0}'), E_{K_2'}E_{z_{1.1}}(z_{1.1}'), \cdots\cdots, \\ & E_{K_2'}E_{z_{r.0}}(z_{r.0}'), E_{K_2'}E_{z_{r.1}}(z_{r.1}'), \\ & S_{GM}(GM, G_2, H(z_{1.0}, z_{1.0}'), \cdots, H(z_{r.1}, z_{r.1}')) \quad)\end{aligned}$$

   Since only the remaining nodes in G_2 know the new group key K_2', i will not be able to decrypt the packet. A node can get a new key in the flat table only if it has the corresponding old secret.

3. To prevent the expelled node from getting access to the multicast traffic from the members of G_1 and G_3, the polynomials $h_{21}(x)$ and $h_{23}(x)$ that determine their personal key shares must be replaced by the new functions $h_{21}'(x)$ and $h_{23}'(x)$. In this part we are going to describe how the new functions can be distributed to the nodes in G_2. In the next part, the update operations for the nodes in G_1 and G_3 will be presented.

   The *group manager* will choose two t-degree functions from $F_q[x]$ as $h_{21}'(x)$ and $h_{23}'(x)$. It will then broadcast the following packet:

$$\begin{aligned}(\quad & GM, G_2, \mathit{polynomial\ update\ for}\ G_2, \\ & E_{K_2'}(GM, G_2, H(h_{21}(x), h_{23}(x)), h_{21}'(x), h_{23}'(x)), \\ & S_{GM}(GM, G_2, H(h_{21}(x), h_{23}(x), h_{21}'(x), h_{23}'(x))) \ )\end{aligned}$$

   Since only the remaining nodes in G_2 know the new group key K_2', they can decrypt the packet and get the new polynomials.

4. As we have discussed in section 5.1, it is not an efficient approach if all personal key update operations are conducted by the *group manager*. On the contrary, the members of G_1 and G_3 can acquire the new secrets in a distributed manner from the nodes in G_2 nearby. Below we use a node v in G_1 and w in G_2 as an example and illustrate how the personal key share can be updated.

   (1) The *group manager* will broadcast an authenticated message and notify all nodes in G_1 and G_3 to acquire the new personal key shares. The ID of the expelled node will also be identified in the packet so that it will be avoided during the key refreshment procedures.

   (2) After verifying the packet from the *group manager*, v will initiate a local broadcast within a few hops and locate a node w in the group G_2. It will then get $h_{21}'(v)$ by sending:

$$\begin{aligned}& v \to w : (v, w, \mathit{request\ for}\ h_{21}'(v), \\ & \quad E_{h_{21}(v)}E_{h_{12}(w)}(v, w, H(h_{21}(v), h_{12}(w)), R) \ ) \\ & w \to v : (w, v, \mathit{reply\ for}\ h_{21}'(v), \\ & \quad E_{h_{21}(v)}E_{h_{12}(w)}(w, v, h_{21}'(v), H(R, h_{21}(v), h_{21}'(v))) \ )\end{aligned}$$

   The random number R is used to guarantee the freshness of the reply. As a secure channel, v can get its new key share from w by using the dual encryption method $E_{h_{21}(v)}E_{h_{12}(w)}(\cdot)$. This procedure can be conducted through a multi-hop path.

5. As an expelled node, i still has the personal key shares $h_{12}(i)$ and $h_{32}(i)$, and it can use these keys to send false information to the members of G_1 and G_3. To prevent such scenarios from happening, the nodes in G_1 and G_3 will maintain a list of the expelled nodes until the new polynomials $h_{12}'(x)$ and $h_{32}'(x)$ are established. Since i will not get the updated personal key shares, it will not be able to generate false information to mislead the wireless nodes in the network.

Studying the key update operations described above, we find that the new keys are established either through true broadcast from the *group manager* or in a distributed manner. All keys held by the expelled node have been aborted to enforce forward secrecy. Although for the clearance of the paper, we put several broadcast messages in separate steps, in the real implementation of the mechanism the information can be merged into one packet. Therefore, more computation and communication resources of the wireless nodes can be preserved for other applications.

The joining and leaving events are the building blocks to describe various member changes in the system, for example, a group switch can be viewed as a leaving action followed by a joining action. The previous investigation focuses on the situations when only one group change happens, and the scenarios in which multiple changes happen simultaneously will be described in section 6.

6. DISCUSSIONS

6.1 Overhead

In this part, we study the storage, computation, and communication overhead of the proposed mechanism. As a comparison, we also illustrate the overhead when every group uses a public-private key pair as described in section 1. We assume that both schemes adopt flat tables for key update. For the simplicity of the analysis, we assume that the plain text and the corresponding ciphertext have the same length. We also assume that all of the public keys, private keys, and personal key shares are in the finite field defined by F_q. The values in table 2 show the analysis results.

Studying the results in table 2, we find that the storage and update of the t-degree polynomials explains most of the additional storage and communication overhead. However, this increase can be justified as follows:

First, compared to the group changes of wireless nodes, the encryption and decryption of the multicast data packets happen much more frequently. By replacing the exponential computation with a symmetric encryption procedure, we can reduce the data processing time at the wireless nodes, thus improving the system efficiency.

The proposed mechanism will help the wireless nodes to achieve a better power usage as well. Since the computation overhead for sending and receiving a multicast packet may include the evaluation of a t-degree polynomial and a symmetric encryption, we study the power consumption of these two operations respectively. Various experiments on real mobile devices [42, 43] have shown that 1.2 - 2.0 mJ energy is required to encrypt/decrypt 1K Byte data when symmetric encryption methods such as AES are used. The transmission and reception of the same amount of data will consume less than 10 times energy. In [32], the authors investigate the computation cost to generate a 64-bit key on MICA2 sensors by evaluating a polynomial with an optimized method. The results show that when $t \leq 80$, the computation overhead is comparable to the calculation of a 64-bit MAC code on a 64-bit message using SkipJack. On the contrary, signing a block of data with RSA or DSA will consume 300 - 550 mJ. By switching to symmetric ciphers, we reduce the energy that is consumed on data encryption/decryption by about 10^2 times. Therefore, the additional transmission and reception overhead for key refreshment is totally paid off.

Second, the adoption of polynomials enables the distribution of personal key shares. Only the sender and the members of the target group can read the information. It becomes more difficult for an attacker to impersonate another node even when additional authentication methods are not applied. The analysis in section 5.1 has also shown that by integrating the personal key shares of two nodes belonging to different groups, we can establish a secure communication channel between them.

Third, when a wireless node changes its group, new keys must be established by the *group manager*. Compared to the overhead of generating a pair of public-private keys, it will be much more efficient to choose several t-degree polynomials from $F_q[x]$. The adoption of this method will simplify the key management operations.

6.2 Security and robustness

The following security and robustness issues of the proposed mechanism are of special interest.

Generating the Group Managers

The *group managers* play an important role in the proposed mechanism: they are in charge of generating the polynomials and flat tables. In addition to the capability to generate secure keys, other features of the *managers* (such as the trustworthiness value, power level) should also be considered.

If a predistributed infrastructure exists in the wireless network, the manager generation procedure can take advantage of those special nodes. For example, in a cellular - ad hoc integrated system, the base stations can maintain the members of every group and generate new keys during changes. As another example, in a wireless mesh network, the fixed nodes can be used for key management.

In a self-organized environment such as ad hoc networks, a more complicated manager election or generation procedure must be adopted. One possible solution is to adopt a variation of the secure leader election algorithms for ad hoc networks [49]. The mobile nodes use a preference function that integrates multiple decision factors to represent the desirability of a candidate. The node that receives the most "votes" will become the manager.

Distributing Key Management Overhead

For the simplicity of the presentation, we have assumed a single *group manager* in the paper. To improve the robustness of the proposed mechanism and avoid the single point of failure, distributed key management can be adopted. Multiple *managers* may perform equally or form a hierarchy to control the key distribution and update procedures for a group. When a joining or a leaving event happens, they can generate the new keys in a collaborative manner to prevent the security defections in one *manager* from degrading the safety of the mechanism. Another advantage is that a wireless node has a higher probability to get the response from a *manager* locally, which will reduce the communication overhead caused by the control traffic. The organization of the *managers* can benefit from the previous results in distributed systems [16, 25].

Defending against Collusive Attacks

The wireless nodes in the same group or from different groups may collude to compromise the proposed mechanism and get illegal access to the multicast traffic. If there is a malicious node in the target group of the multicast packet, nothing can prevent it from decrypting the packet and sharing the information with other attackers. Therefore, in the following discussion, we focus on the intra-group attacks.

The malicious nodes in the same group can benefit from the collusion by reconstructing the polynomials of other groups. They can recover the personal key shares of the innocent members and get illegal access to the multicast traffic that is not destined to them. Since a t-degree polynomial is resistant to the coalition of up to t compromised members, we can adjust the choice of this parameter based on the security levels of different groups to achieve a better tradeoff between the safety of the mechanism and the storage and computation overhead.

Handling Multiple Changes Simultaneously

The adoption of flat tables for key updates restricts the number of leaving events that can be handled simultaneously. In the worst case, two nodes may have complementary IDs and each node holds a half of the keys in the flat table. For example, node i has the keys $z_{1.i_1} \cdots z_{r.i_r}$, and

Table 2: Comparison of the overhead.

	Scheme using public-private key pairs	Proposed mechanism
key storage overhead at the wireless nodes	$(r+4)\log q$	$(r+5+2t)\log q$
the amount of broadcast traffic for key update during a joining event	$(2r+2)\log q$	$(2r+3+2t)\log q$
the amount of broadcast traffic for key update during a leaving event	$(3r+1)\log q$	$(3r+2+2t)\log q$
encryption and decryption overhead for the transfer of a multicast packet	exponential computation	evaluating a t-degree polynomial + a symmetric encryption

node $\bar{i}$ has $z_{1.\overline{i_1}} \cdots z_{r.\overline{i_r}}$. When these two nodes are expelled from the current group at the same time, we will not be able to find a secure channel to distribute the refreshed keys unless they are processed in two separate rounds.

If we denote the set of KEKs as $\mathbf{K}$, the set of keys held by node i as $\mathbf{K}_i$, and the set of expelled nodes as $\mathbf{R}$, the set $\{\mathbf{K}-\bigcup \mathbf{K}_j, j \in \mathbf{R}\}$ represent the secrets that can be used for key refreshment. To guarantee that every remaining node in the group has at least one of such keys, the relationship among the size of $\mathbf{K}$, the size and distribution of $\mathbf{K}_i$, and the largest number of leaving events that can be processed simultaneously must be investigated.

The Exclusion Basis System (EBS) [38] can be used to construct an approach for key management under these conditions. EBS provides a matrix-like key distribution structure, in which every node knows g keys and does not know l keys. When multiple leaving events are processed at the same time, the keys that are unknown to all expelled nodes will be used to establish a secure channel for key refreshment and other operations. Similar ideas have been applied to CKDS [27] and GKMPAN [28] and more details of EBS can be found in [38, 39].

Cross-verifying Personal Key Shares

The distributed refreshment of personal key shares reduces the overhead at the *group manager*. However, an undetected attacker who knows the old polynomial can provide false keys to the wireless nodes so that their multicast traffic cannot be decrypted by the target group members. To defend against such attacks, a wireless node may locate multiple holders of the new polynomial and cross verify the key shares acquired from different sources. If different keys are detected, the node can either refer to the *manager* for a final judgment or adopt the local majority result.

6.3 Other approaches for key organization

Although inter-group multicast has some special features, many approaches for intra-group multicast can be adapted to serve the new environment. In this part, we investigate two such solutions.

The organization of the nodes in Iolus [13] seems to suit the new environment very well: the entities are divided into multiple subgroups, and each subgroup uses an independent key. However, if the same structure is directly applied and every subgroup is mapped to a group, some problems may arise. First, since the inter-group traffic must be translated by the agents and they can get access to the information not destined to them, a higher level of trust is required in these nodes. Second, the agents may become bottlenecks in the system when the extra computation and communication overhead is considered. Although Iolus cannot be directly applied, it can be integrated into the proposed mechanism to manage the nodes in the same group. The separation of keys among different subgroups will allow the member changes to be handled locally, thus reducing the control overhead.

If the KEKs are organized as a Logical Key Hierarchy (LKH) instead of a flat table, a better resistance to collusion will be provided. At the same time, less computation and communication overhead may be required during group changes. As the costs to these advantages, more keys must be generated and maintained at the *group managers*. Based on the available resources to the wireless nodes, a suitable scheme can be chosen when the application environment has been determined.

6.4 Future work

Integrating Self-healing Property

The movement of wireless nodes may lead to topology changes and network partitions in the system. There are chances that the nodes will miss some of the key update messages due to the error-prone transmission medium or unavailable paths. The *stateless property* is highly desirable in wireless networks, which allows a mobile node to recover the current group key without requesting it from the *manager*. Several protocols that support this property [30, 28, 31] have been proposed in previous approaches.

We plan to integrate the self-healing scheme into the proposed mechanism to improve its performance in highly mobile environments. The t-degree polynomials will be protected by the masking functions and the wireless nodes with suitable keys will be able to recover the lost secrets without interacting with the *managers*.

Conducting Performance Investigation

The proposed mechanism will be evaluated through simulation. A group of experiments will be conducted to investigate the performance and overhead under different node mobility models and communication scenarios. We plan to adopt the trust evaluation methods proposed in [40] for the group management and various user behavior patterns [41] will be studied. The simulation results will help us to identify the features that impact the performance the most and provide guidelines for future improvement.

7. CONCLUSIONS

Secure multicast has become an important component of many applications in wireless networks. In this paper, we focus on key distribution and update for secure inter group communication. Different from the previous approaches that depend on asymmetric encryption methods, we adopt polynomials to support the distribution of personal key shares and employ flat tables to achieve efficient key refreshment. The proposed mechanism reduces the computation overhead to process the multicast packets and improves the power usage at the wireless nodes. It becomes more difficult for an attacker to impersonate another entity in the network. The additional storage and communication overhead caused by the proposed mechanism has also been justified.

We plan to integrate the self-healing property into the proposed mechanism. Additional research is also required to study the impacts of group changes and traffic patterns on its performance. The results will lead to a more robust and efficient key distribution protocol for secure intra and inter group communication in wireless networks.

8. ACKNOWLEDGEMENT

The authors would like to thank the valuable comments from the committee members and the workshop chairs. This research is supported in part by NSF ANI 0219110 and NSF IIS 0242840. The authors would also like to thank the seeding fund from ITTC, CERIAS, and CISCO.

9. REFERENCES

[1] S. Yi, P. Naldurg, and R. Kravets. Security-aware ad hoc routing for wireless networks. In *Proc. of ACM International Symposium on Mobile Ad Hoc Networking & Computing*, 299–302, 2001.

[2] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner. The VersaKey Framework: Versatile group key management. *IEEE JSAC Special issue on middleware*, 17(9), 1614–1631, 1999.

[3] I. Chang, R. Engel, D. Kandlur, D. Pendarakis, and D. Saha. Key management for secure internet multicast using boolean function minimization techniques. In *IEEE INFOCOM*, 689–698, 1999.

[4] T. Hodes and R. Katz. Composable ad hoc location based services for heterogeneous mobile clients. *Wireless Networks*, 5(5):411-427, 1999.

[5] J. Agre, A. Akinyemi, L. Ji, R. Masuoka, and P. Thakkar. A layered architecture for location based services in wireless ad hoc networks. In *Proc. of IEEE Aerospace Conference*, 2002.

[6] H. Harney and C. Muckenhirn. Group Key Management Protocol (GKMP) Architecture. RFC 2094, 1999.

[7] D. Wallner, E. Harder, and R. Agee. Key Management for Multicast: Issues and Architectures. RFC 2627, 1999.

[8] C.K. Wong, M.G. Gouda, and S.S. Lam. Secure group communications using key graphs. *IEEE/ACM Transactions on Networking*, 8(1), 16-30, 2000.

[9] D.A. McGrew and A.T. Sherman. Key establishment in large dynamic groups using oneway function trees. Tech. Rep. No. 0755, Network Associates, Inc., 1998.

[10] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast Security: A Taxonomy and Some Efficient Constructions. In *IEEE INFOCOM*, 708-716, 1999.

[11] R. Canetti, T. Malkin, and K. Nissim. Efficient communication-storage tradeoffs for multicast encryption. In *Advances in Cryptology – EUROCRYPT*, 459-474, 1999.

[12] A. Perrig, D. Song, and J.D. Tygar. ELK, a new protocol for efficient large-group key distribution. In *Proc. of IEEE Symposium on Security and Privacy*, 2001.

[13] S. Mittra. Iolus: A framework for scalable secure multicasting. In *ACM SIGCOMM*, 277-288, 1997.

[14] L. Dondeti, S. Mukherjee, and A. Samal. Scalable secure one-to-many group communication using dual encryption. *Computer Communications*, 23(17), 1681-1701, 1999.

[15] R. Molva and A. Pannetrat. Scalable multicast security in dynamic groups. In *Proc. of ACM CCS*, 101–112, 1999.

[16] S. Rafaeli and D. Hutchison. Hydra: A decentralized group key management. In *Proc. of IEEE International Enterprise Security Workshop*, 2002.

[17] B. Briscoe. MARKS: Multicast key management using arbitrarily revealed key sequences. In *Proc. of International Workshop on Networked Group Communication*, 1999.

[18] S. Setia, S. Koussih, and S. Jajodia. Kronos: A scalable group re-keying approach for secure multicast. In *Proc. of IEEE Symposium on Security and Privacy*, 2000.

[19] R. Pietro, L. Mancini, Y. Law, S. Etalle, and P. Havinga. LKHW: A Directed Diffusion-Based Secure Multicast Scheme for Wireless Sensor Networks. In *Proc. of IEEE International Conference on Parallel Processing Workshops*, 2003.

[20] T. Kaya, G. Lin, G. Noubir, and A. Yilmaz. Secure multicast groups on ad hoc networks. In *Proc. of ACM workshop on security of ad hoc and sensor networks*, 94 – 102, 2003.

[21] L. Lazos and R. Poovendran. Location-Aware Secure Wireless Multicast in Ad-Hoc Networks under Heterogeneous Pathloss. Technical Report UWEETR-2003-0012, University of Washington, 2003.

[22] L. Lazos and R. Poovendran. Energy-Aware Secure Multicast Communication in Ad-hoc Networks Using Geographic Location Information. In *Proc. of IEEE International Conference on Acoustics Speech and Signal Processing*, 2003.

[23] S. Mäki, T. Aura, and M. Hietalahti. Robust Membership Management for Ad-hoc Groups. in *Proc. of Nordic Workshop on Secure IT Systems*, 2000.

[24] A. Yasinsac, V. Thakur, S. Carter, and I. Cubukcu. A Family of Protocols for Group Key Generation in Ad Hoc Networks. In *Proc. of IASTED International Conference on Communications and Computer Networks*, 183–187, 2002.

[25] B. DeCleene, L. Dondeti, S. Griffin, T. Hardjono, D. Kiwior, J. Kurose, D. Towsley, S. Vasudevan, and C. Zhang. Secure Group Communications for Wireless Networks. In *IEEE MILCOM*, 2001.

[26] D. Bruschi and E. Rosti. Secure multicast in wireless networks of mobile hosts: protocols and issues. *Mobile Networks and Applications*, 7(6), 503–511, 2002.

[27] M. Moharrum, R. Mukkamala, and M. Eltoweissy. CKDS: An Efficient Combinatorial Key Distribution

Scheme for Wireless Ad-Hoc Networks. In *Proc. of IEEE International Conference on Performance, Computing, and Communications*, 631–636, 2004.

[28] S. Zhu, S. Setia, S. Xu, and S. Jajodia. GKMPAN: An Efficient Group Rekeying Scheme for Secure Multicast in Ad-Hoc Networks. In *Proc. of International Conference on Mobile and Ubiquitous Systems: Networking and Services*, 42–51, 2004.

[29] A. Shamir. How to share a secret. *Communications of the ACM*, 22, 612–613, 1979.

[30] J. Staddon, S.Miner, M. Franklin, D. Balfanz, M. Malkin, and D. Dean. Self-Healing Key Distribution with Revocation. In *Proc. of IEEE Symposium on Security and Privacy*, 2002.

[31] D. Liu, P. Ning, and K. Sun. Efficient self-healing group key distribution with revocation capability. In *Proc. of ACM conference on Computer and communications security*, 231–240, 2003.

[32] D. Liu, P. Ning, and R. Li. Establishing Pairwise Keys in Distributed Sensor Networks. In *ACM Transactions on Information and System Security*, 8(1), 41–77, 2005.

[33] S.M. More, M. Malkin, J. Staddon, and D. Balfanz. Sliding-window self-healing key distribution. In *Proc. of ACM workshop on Survivable and self-regenerative systems*, 82–90, 2003.

[34] R. Pickholtz, D. Schilling, and L. Milstein. Theory of spread spectrum communications – a tutorial. *IEEE Trans. Comm.*, 1982.

[35] V. Gupta, S. Krishnamurthy, and M. Faloutsos. Denial of service attacks at the MAC layer in wireless ad hoc networks. In *Proc. of Milcom*, 2002.

[36] P. Bjorklund, P. Varbrand, and D. Yuan. Resource optimization of spatial TDMA in ad hoc radio networks: A column generation approach. In *IEEE INFOCOM*, 2003.

[37] M. Steiner, G. Tsudik, and M. Waidner. Key Agreement in Dynamic Peer Groups. *IEEE Transactions on Parallel and Distributed Systems*, 11(8):769–780, 2000.

[38] S.T. Redwine Jr.. A Logic for the Exclusion Basis System. In *Proc. of the Hawaii International Conference on System Sciences (HICSS)*, 2004.

[39] L. Morales, I.H. Sudborough, M. Eltoweissy, and M.H. Heydari. Combinatorial Optimization of Multicast Key Management. In *Proc. of the Hawaii International Conference on System Sciences*, 2003.

[40] B. Bhargava and Y. Zhong. Authorization Based on Evidence and Trust. In *Proc. of Data Warehouse and Knowledge Management Conference (DaWak)*, 2002.

[41] Y. Zhong. Formalization of Dynamic Trust and Uncertain Evidence for User Authorization. Ph.D. Thesis, Dept. of CS, Purdue University, 2005.

[42] N. Potlapally, S. Ravi, A. Raghunathan, and N. Jha. Analyzing the energy consumption of security protocols. In *Proc. of International symposium on Low power electronics and design*, 30–35, 2003.

[43] P. Ni and Z. Li. Energy Cost Analysis of IPSec on Handheld Devices. *Microprocessors and Microsystems, special issue on Secure Computing Platform*, 28(10), 585–594, 2004.

[44] X. Chen and J. Wu. Multicasting techniques in mobile ad hoc networks. In *The handbook of ad hoc wireless networks*, 25–40, CRC Press, Inc., 2003.

[45] L. Ji and M. Corson. Differential destination multicast - a MANET multicast routing protocol for small groups, In *Proc. of IEEE INFOCOM*, 2001.

[46] L. Ji and M. Corson. Explicit multicasting for mobile ad hoc networks. In *Mobile Networks and Applications*, 8(5), 535–549, 2003.

[47] K. Chen and K. Nahrstedt. Effective Location-Guided Tree Construction Algorithms for Small Group Multicast in MANET, In *Proc. of IEEE INFOCOM*, 1180–1189, 2002.

[48] C. Gui and P. Mohapatra. Efficient Overlay Multicast for Mobile Ad Hoc Networks, In *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, 2003.

[49] S. Vasudevan, B. DeCleene, N. Immerman, J. Kurose, and D. Towsley. Secure Leader Election Algorithms for Wireless Ad Hoc Networks. In *Proc. of IEEE DARPA Information Survivability Conference and Exposition (DISCEX)*, 2003.