Efficient Hybrid Key Agreement Protocol for Wireless Ad Hoc Networks

Xiang-Yang Li* Yu Wang* Ophir Frieder*

Abstract— **Secure and efficient communication among a set of mobile nodes is one of the most important aspects in ad-hoc wireless networks. To ensure the security, several cryptography protocols must be implemented. Due to the resource scarcity in wireless networks, the protocols must be communication efficient and need as less computational power as possible. To secure the group broadcasting in wireless networks, often a group key is needed so that efficient conventional encryption, such as DES and AES, can be used. Several group key management protocols have been proposed. However, not all of them are communication efficient when applied to wireless ad-hoc networks. In this paper, we propose a key agreement protocol that is communication efficient by using connected dominating set concept to set up subgroups among all wireless nodes. We also show how to manage the group efficiently in a mobile environment.**

Keywords— **Wireless networks, network security, key agreement, connected dominating set.**

I. Introduction

Wireless ad hoc networks have attracted significant attentions recently due to its wide applications in different areas. In a wireless ad hoc network, there exists no fixed infrastructure such as switching centers or base stations. Mobile nodes that are within the communication range of each other can communicate directly whereas, the nodes that are far apart have to rely on intermediary nodes (routers) to relay messages. The mobility of a node in the wireless ad hoc network can cause changes to the network topology. Military operations remain the main application of ad hoc networks even today. Ad hoc networks can also be used for emergency, law enforcement and rescue missions. Since, the cost to set up an ad hoc network is low, it is a very attractive option for commercial uses. However, little has been done on designing secure protocols that are suitable for wireless ad hoc networks, especially designing communication efficient group key agreement protocols for group broadcasting in wireless ad hoc networks.

Security is an important issue for wireless ad hoc networks. Same to the wired networks, there are five main attributes of security for a wireless ad hoc network: *availability*, *confidentiality*, *integrity*, *authentication* and *non-repudiation*. Availability is to ensure that the network services survive despite denial of service attacks. A denial of service attack could be implemented at any level of the ad hoc network, e.g., jamming the frequency to interfere with communication. The key management service could be a target of this attack. Confidentiality ensures that information is passed only to authorized members of the network. Integrity is to guarantee that a message is transferred without getting corrupted. Authentication is to enable a node to identify the identity of the peer node it is communicating with. This is a useful property to detect isolated or compromised nodes. Non-repudiation is to ensure that a node cannot deny having sent/received the message.

There are many ways that an ad hoc network can be attacked upon. The use of wireless links gives ample opportunity for link attacks ranging from passive eavesdropping to active message replay, impersonation and distortion. Other than the above mentioned external attacks on the ad hoc network, there are possibilities of attacks from within the network by compromised nodes. An ad hoc network is dynamic because of frequent changes in both its topology and its membership (i.e., nodes frequently join and leave the network). Thus, any security solution with a static configuration is not enough. It is desirable for our security protocols to adapt to these changes. In the near future, an ad hoc network may consist of hundreds or even thousands of nodes. Security protocols should be scalable to handle such a large network.

Almost all cryptography protocols are based on private keys or public keys. Public key based protocols have some inherent advantages over the private key protocols. However, it is well-known that the private key based encryption protocols (such as DES, AES) is much faster than the public key based protocols (such as RSA, ElGamal). In this paper, we will concentrate on how to build a common private key for a group so they can communicate securely. Several group key management protocols [1], [2], [3], [4], [5] have been proposed for wired networks. However, not all of them are communication efficient when applied to wireless ad-hoc networks. In this paper, we propose a hybrid key agreement protocol that is communication efficient by using connected dominating set concept to set up subgroups among a group of wireless nodes. We apply some existed group key agreement protocols both for each subgroup and for the backbone. As the selected existing protocols used in our hybrid protocol have been shown to be secure, thus, our hybrid protocol also is secure.

The rest of the paper is organized as follows. In Section II, we give some preliminaries needed to describe our new protocol. Then we briefly review several previous key agreement protocols in Section III. Section IV presents our new hybrid key agreement protocol for wireless ad hoc networks based on CDS and some existing protocols. In addition, in Section V we show how our hybrid protocol can be easily adapted to mobile environment. We conclude our paper in Section VI.

II. Preliminaries

We now present some definitions and general terminologies used in this paper. They have been adapted from [2], [6]. A *key agreement protocol* is a key establishment method in which, a shared secret key is derived by two or more specified parties as a function of information contributed by, or associated with, each of these, such that no party can predetermine the resulting value. A key agreement protocol is *contributory* if each party *equally* contributes to the key and guarantees its freshness. A key establishment protocol is *distributory* if there is a party (called *trusted third party*) that generates the key and then distributes the key securely to all the other parties. Let $\mathcal{P}$ be a n-party key agreement protocol, and $\mathcal{M}$ be the set of members in the protocol and let S_n be a secret key generated as a result of $\mathcal{P}$. The protocol $\mathcal{P}$ is said to provide *implicit key authentication* if each $M_i \in \mathcal{M}$ is assured that no party $Q \notin \mathcal{M}$ can learn the key S_n. Then this protocol is called an *authenticated key agreement*. The protocol $\mathcal{P}$ provides *key confirmation* if any

*Department of Computer Science, Illinois Institute of Technology, Chicago IL 60616, USA. Email: xli@cs.iit.edu, wangyu1@iit.edu, ophir@cs.iit.edu

member is assured that its peer(s) do in fact, possess the particular key S_n. A key agreement protocol offers *perfect forward secrecy* if the compromising of a long-term key S_n cannot result in the compromising of the keys generated before S_n. On the other hand, a key agreement protocol is said to be vulnerable to *known key attacks* if the compromising of past keys allows a passive adversary to get future group keys, or an active adversary to impersonate one of the protocol members. For more detailed discussions of the above definitions, see [2], [7], [8].

Then we present some of the definitions and conventions used in a wireless ad-hoc environment. Assume that all wireless nodes are given as a set V of n nodes in a two-dimensional space. Each node is assumed to have some computational power and an omni-directional antenna. A message sent by a node can be received by all nodes within its transmission range. Here we assume every node has the same maximum transmission range which is normalized to one unit. All these nodes induce a *unit disk graph* UDG(V), in which, there is an edge between two nodes if and only if the distance between them is at most one unit. In other words, two nodes can communicate directly if and only if they are connected in UDG(V). If the receiver is not within the transmission range of the sender, the message is forwarded by some other intermediate nodes. The UDG(V) is always assumed to be a connected graph. All the nodes within a constant k-hop neighborhood of a node $u \in V$ are the *k-local nodes* or *k-hop neighbors* of u, represented by $N_k(u)$ hereafter. All nodes are assumed to be almost static for a reasonable period of time.

The *connected domination set* (CDS) was used as a virtual backbone for routing in wireless as hoc networks. It also plays an important role in our hybrid group key agreement protocol. A subset V' of V is a *dominating set* if each node in $u \in V$ is either in V' or is adjacent to some node $v \in V'$. A subset V' of V is a *connected dominating set* if V' is a dominating set and V' induces a connected subgraph. The nodes in a connected dominating set V' can communicate with each other using only nodes in V'. A dominating set with minimum size is called the *minimum dominating set* (MDS). A connected dominating set with minimum size is called the *minimum connected dominating set* (MCDS).

III. Protocols Review

In general, the key establishment protocols can be classified into two types: key distribution protocols and key agreement protocols [5]. The key distribution protocols, sometimes called as *centralized* key distribution protocols, are generally based on a trusted third party (TTP). The key agreement protocols, on the other hand, do not use a TTP but rely on the group members for a general key agreement. The centralized method has the following disadvantages: (1) The TTP that generates and distributes the key for a large group is a single point of failure. (2) The TTP is also a most attractive target for all kinds of adversaries and attacks. (3) To allow a single party to generate the keys for a whole group might not be acceptable in all cases. For example, consider a case of rival or competing groups in an existing group.

Even though the general argument of ours is in favor of distributed and contributory key agreement protocols, the necessity of a controlling member for any group is recognized. The role of TTP is sometimes useful in synchronization of membership operations like, addition and deletion and thus is generally a policy decision. In addition, when the number of group members is large, it is often expensive to use contributory key agreement protocols. Most of the proposed key agreement protocols assume that the communication cost between any pair of group members is one unit, which is not true in wireless ad hoc networks.

It is easy to have a secure centralized key distribution protocol: the trusted party or the group header constructs a secure communication channel with each of its members and then distributes the generated group key to each member. The secure communication channel can be built using the Diffie-Hellman protocol [9] or its variations. Thus, hereafter, we will concentrate on contributory key agreement protocols.

A. Two-party Key Agreement

The following notation is used throughout the paper:

n	number of members in the protocol
i, j, k	indices of members (range $[1, n]$)
M_i	i-th group member
q	order of an algebraic group $\mathcal{G}$
α	exponential base delimited by q
r_i	random exponent generated by M_i
S_n	Group key shared among n members

Here, we have two members M_1 and M_2. Member M_1 sends α^{r_1} to M_2 and M_2 sends α^{r_1} to M_1. M_1 computes the key $K = (\alpha^{r_2})^{r_1}$ and vice-versa for M_2. The security of this protocol is based on the assumption of the difficulty of the discrete log-arithmetic and the Diffie-Hellman Decision problem. This classic Diffie-Hellman protocol is vulnerable to the so called man-in-the-middle attack. Several remedy protocols were proposed. See [10] for a more detailed discussion.

B. Protocol GDH

The generic N-party Diffie-Hellman key exchange protocol [4] is very similar to the 2-party case. Let p be a prime and q be a prime divisor of $p-1$. Let $\mathcal{M} = \{M_1, ..., M_n\}$ be a set of users wishing to generate a group key. All participants $M_1, ..., M_n$ agree a *priori* on a cyclic group, $\mathcal{G}$, of order q, and a generator α, of this group $\mathcal{G}$. For each key exchange, each member M_i chooses a random value $r_i \in \mathcal{G}$. Let S_n be the computed group key at last.

Two practical protocols GDH.2 and GDH.3 were presented. The GDH.2 protocol tries to minimize the total number of protocol messages. GDH.3, on the other hand, tries to minimize the computational costs. Although the discussion below focuses on GDH.2, all of the techniques applied to GDH.2 can be adapted to GDH.3. For the completeness of the presentation, we briefly review the GDH.2 protocol below.

Algorithm 1: Group Diffie-Hellman GDH.2
Round i $(1 \le i \le n-1)$:
1. Member M_i selects a random integer $r_i \in Z_q^*$.
2. M_i sends M_{i+1}: $\alpha^{\Pi_{k=1}^{i} r_k}$ and $\alpha^{(\Pi_{k=1}^{i} r_k)/r_j}$, $\forall 1 \le j \le n$.

Round n:
1. Member M_n selects a random $r_n \in Z_q^*$.
2. M_n sends each M_i a number $y_i = \alpha^{(\Pi_{j=1}^{n} r_j)/r_i}$.
3. Each member M_i then computes the final key as

$$S_n = y_i^{r_i} = \alpha^{\Pi_{j=1}^{n} r_j}.$$

The GDH.2 protocol executes n rounds. In the first stage (the first $n-1$ rounds), contributions are collected from each individual group member M_i. The numbers collected by member M_i from M_{i-1} is necessary to compute the numbers that M_i has to send to M_{i+1}. Then, in the second stage (the n-th

round), the group keying material y_i is sent to member M_i. In [4], Steiner *et al.* also discussed adding and deleting members for GDH.2. For more detail, see [4].

C. Authenticated GDH.2 protocol

An extension of the GDH.2 protocol (Algorithm 1) to provide implicit key authentication was presented in [2]. For the completeness of presentation, Algorithm 2 briefly reviews this protocol. The assumption made is that M_n shares (or is able to share) a distinct key $K_{i,n}$ with each member M_i.

Algorithm 2: Authenticated GDH.2 (A-GDH.2)
ROUND i ($1 \leq i \leq n-1$): Same steps from GDH.2
ROUND n:
1. M_n selects a random integer $r_n \in Z_q^*$.
2. For each M_i, M_n sends M_i: $\{y_i = \alpha^{K_{i,n} \cdot \prod_{1 \leq j \leq n, j \neq i} r_j}\}$.
3. After receiving the above message, each M_i computes the group key as $S_n = y_i^{r_i \cdot K_{i,n}^{-1}}$.

In this protocol, each group member obtains an authenticated shared key with M_n. A slight modification of the A-GDH.2 is that M_n sends each member M_i the final key encrypted by some symmetric encryptions using key $K_{i,n}$ instead of sending them some number and let M_i compute the key. In this variation, the member M_n basically acts as the trusted third party (TTP).

D. Tree Based Approach

The application of the tree structure to key agreement protocols has been proposed earlier [11], [12]. In [13], [14], they used one-way functions to enhance the security in protocols based on tree-liked structures. A group of members, participating in the key agreement protocol, has a group manager who maintains a binary tree. Each node x has two cryptographic keys, a *node key* k_x and a *blinded node key* $b_x = g(k_x)$, i.e., the blinded node key b_x is computed from the node key k_x using the one-way function g. The key is blinded in the sense that an adversary with limited computational capability can know b_x but cannot find k_x. The application of key trees to a distributed environment has been proposed in the protocol *Tree Group Diffie-Hellman* (TGDH) [3] key agreement protocol. TGDH combines a binary tree structure with the group Diffie-Hellman technique.

The TGDH protocol, uses the hierarchy in a binary tree to its advantage. The root is at the topmost level, given a value of 0 and all the leaves are at the lowest level h. Since the tree is a binary tree, each node is a leaf or a parent of two nodes. Each leaf node in the tree represents a group member M_i. The internal nodes are used for the key management and do not represent any individual member. Each node of the tree is represented by (l, v), where l is its level in the tree and v is the index of this node in level l. The key associated to node (l, v) is $k_{(l,v)}$ and its blinded key $b_{(l,v)} = \alpha^{k_{(l,v)}} \mod p$. See [3] for detail. Each group member contributes equally to the group key. In other words, for each internal node (l, v), its associated key $k_{(l,v)}$ is derived from its children's keys. In [3], they defined $k_{(l,v)} = b_{(l+1,2v+1)}^{k_{(l+1,2v)}}$, which is essentially $\alpha^{k_{(l+1,2v+1)} \cdot k_{(l+1,2v)}}$. The group key is a result of contributions by the current members. The final group key is calculated at the root $(0, 0)$ and it is shared by all the members.

E. Hypercube Based Approach

An extension of the Diffie-Hellman key exchange protocol to multi-members was proposed in [1], [15] by arranging the group members in a d dimensional hypercube. The idea behind the hypercube key agreement approach is shown in the Figure 2 with four members A, B, C, D who want to agree on a key.

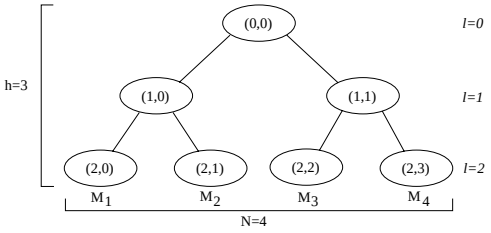

Fig. 1. Tree based representation of members.

Each of them is given a two bit address 00, 01, 10, 11 respectively. Let the contribution by each member to a 2-party DH be S_A, S_B, S_C and S_D. In the first round, illustrated by Figure 2(a), A and B engage a 2-party Diffie-Hellman protocol and calculate the key $S_{AB} = \alpha^{S_A S_B}$; C and D similarly calculate the key $S_{CD} = \alpha^{S_C S_D}$. In the next round, illustrated by Figure 2(b), A and C participate in the Diffie-Hellman protocol using S_{AB} and S_{CD} as random numbers instead of selecting new random numbers. In other words, they will generate a key $\alpha^{S_{AB} S_{CD}}$. Similarly, B and D also participate the Diffie-Hellman protocol to get the key $\alpha^{S_{AB} S_{CD}}$. Thus, the final key calculated by all members is $S_{ABCD} = \alpha^{S_{AB} S_{CD}}$.

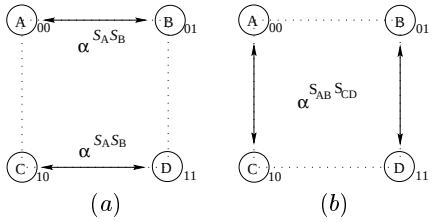

Fig. 2. (a) Round 1: pairwise exchange in a d-cube. (b) Round 2: pairwise exchange in a d-cube.

For simplicity, we assume that the number of members is $n = 2^d$. Each member is assigned a vertex and a unique d-bit address from the set Z_n. The protocol runs for d rounds. In the j^{th} round, neighbors along the j^{th} dimension of the hypercube participate in a 2-party Diffie-Hellman protocol. After d rounds all members share the same key.

Becker and Wille [15] gave a detailed study of the communication complexities of the hypercube based protocol. They also discussed a protocol named *octopus*, in which members are divided into four disjoint subgroups, and each subgroup has a member as the header. Let A, B, C, D be such four group headers. Member A first builds a secure communication channel with each of its subgroup member A_i. Then A_i generates a random k_i and sends it to A. A calculates $S_A = \prod_{A_i \in G(A)} k_i$, where $G(A)$ is the subgroup headed by A. Headers B, C, and D also generate S_B, S_C, and S_D. Then A, B, C, D perform a hypercube key agreement protocol described above to get a common key $K = \alpha^{S_{AB} S_{CD}}$. Then A sends each of its subgroup members A_i the following values: $x_i = (\alpha^{S_B})^{S_A / k_i}$ and $y = \alpha^{S_{CD}}$. Notice that A knows S_A, k_i, α^{S_B} and $\alpha^{S_{CD}}$. Then member A_i first calculates $S_{AB} = x_i^{k_i}$ and the final key as $y^{S_{AB}} = \alpha^{S_{AB} S_{CD}}$, which is same for all group members. Our hybrid key agreement protocol will apply similar approach, but we allow more group members and we utilize the geometric properties to partition the subgroups to achieve communication efficiency.

F. Discussions

Most of the previously proposed protocols inherently assume that there is an ordering among all members and a leader is

elected among all members before a key agreement protocol is executed. In addition, most of the protocols assume that the communication cost by sending a message from M_i to some M_j is one unit. For example, GDH protocols [2] assume that the communication cost between M_i and M_{i+1} is one unit; the protocols based on hyper-cube [1] assume that the communication cost between two neighbors in a hyper-cube is one unit. However, these assumptions are neither available (e.g., ordering among members) nor true (e.g., communication costs between some specific nodes are one unit) in wireless ad hoc networks. The communication cost of two nodes is one unit in wireless ad hoc networks if they are within the transmission range of each other. We are interested in designing communication efficient key agreement protocols for wireless ad hoc networks.

IV. Hybrid Approach

In a wireless ad-hoc environment, the number of members could be very large. This presents to us a scenario in which we could divide all members into multiple groups: each group performs some key agreement or distribution protocols; the group leaders perform some key agreement or distribution protocols among them to get a consensus among all these groups.

A. General Method

In our proposed protocol, the first round is a clustering method that divides the entire set of nodes into subgroups based on the geometric locations of nodes. Each of these subgroups selects a leader (also called dominator). This selection process is done by generating a connected dominating set (CDS) [16], [17], [18], [19] from the set of wireless nodes. Once the CDS has been constructed, the set of dominators of the CDS form the *subgroup leaders*; each dominator and the set of its dominatees form an individual subgroup. We use the term subgroup leader and dominator interchangeably hereafter. We could apply the GDH.2 protocol (Algorithm 1) or some other key agreement protocols to the set of dominators and the key is generated as a contribution from all the dominators (and connectors). On success of the key agreement protocol over the dominators, each dominator and the set of its group members follow the key distribution protocol if the same key for each subgroup is required. The afore mentioned steps make sure that all the nodes share the same key, and an overall key agreement is reached. We first outline our hybrid key agreement protocol that is suitable for wireless ad hoc networks.

Algorithm 3: Hybrid Key Agreement Protocol
1. Wireless nodes construct a CDS distributively.
2. The contributory key agreement protocol is applied among the set of computed dominators and connectors.
3. Each dominator distributes the computed key to all its dominatees if the same key is required. Otherwise, each subgroup performs its own key agreement protocol.

Here we prefer the subgroup key be different from the key for backbone. This difference adds more freedom of managing the group members such as joining and leaving. The motivation of using CDS is the presence of group leaders for efficient management of the group membership activities. In addition, using CDS will potentially save the communication cost as we will analyze later. It is critical to reduce communication costs in wireless ad hoc networks as the wireless nodes are often powered by batteries only.

Although we will not elaborate in detail, we require that two nodes need authenticate each other when they want to communicate with each other for key agreement. This will prevent some nodes outside of the group from impersonating the node in the group.

B. Connected Dominating Set

Algorithms of building a connected dominating set typically consist of two stages: the first one is the clustering or the group formation and the second one is to find connectors to make the cluster heads connected. The clustering algorithm works as follows: it finds a subset of nodes such that the rest of the nodes are visible to at-least one of the cluster heads (*dominators*). The node that is not a cluster head is called a *dominatee*.

Assume that each node has a distinct identity (ID). A node is called *white* if its status is not determined. We then briefly review some clustering methods [16], [19]:

Algorithm 4: Clustering
- A white node claims itself to be a dominator if it has the smallest ID among all of its white neighbors if there is any, and broadcasts IamDominator to its 1-hop neighbors.
- A white node receiving IamDominator message marks itself as dominatee and broadcasts IamDominatee to its 1-hop neighbors.

Our assumption is that each node knows the IDs of all its 1-hop neighbors, which can be achieved by asking each node to broadcast its ID to its 1-hop neighbors initially. The set of dominators generated by the above method is actually a maximal independent set. So far, each node only has to broadcast twice to its 1-hop neighbors: one for telling its ID and one for notifying its dominator/dominatee status.

We continue to review the method of finding connectors. The connectors are from the set of nodes among all dominatees that connect the dominators. The connectors and the dominators together form the *connected dominating set*. Several communication efficient algorithms [16], [17], [19], [20] for finding connectors have been proposed.

Algorithm 5: FindingConnectors
- Every dominatee node w broadcasts IamDominatee(w,v) for each dominator v present in Dominators.
- When node u receives a message IamDominatee(w,v) where $u \neq v$, and there is no pair ($*$,v) in Connector2HopsPath, the u adds (w,v) to Connector2HopsPath and broadcasts message 2HopsPath(u,w,v) to its 1-hop neighbors. Here $*$ is a node ID.
- In the event where, a node w receives a 2HopsPath(u,w,v) message it marks itself as a connector.
- When a dominator x receives a message 2HopsPath(u,w,v) where $x \neq w$, if there is not triplet ($*$,$*$,v) in Connector3HopsPath then x adds (u,w,v) to Connector3HopsPath and broadcasts 3HopsPath(x,u,w,v) to its 1-hop neighbors.
- In the event where, a node u receives a 3HopsPath(x, u,w,v) message it marks itself as a connector.

The messages used are (1) IamDominator(u): Node u informs its 1-hop neighbors that u is a dominator. (2) IamDominatee(u,v): Node u informs its 1-hop neighbors that u is a dominatee of v. (3) 2HopsPath(u,v,w): Node u informs its 1-hop neighbors that u has 2-hops path uvw and w is the unique node selected by u to connect u, v. (4) 3HopsPath(x,u,v,v): Node x informs its 1-hop neighbors that x has a 3-hops path $xuwv$and u and w are the nodes selected by x and u respectively, to connect x and v.

In order to save memory cost at each wireless node, the following structures were designed: (1) Dominators: Stores all the dominator's of u, if any. If the node u is a dominator nothing is assigned. (2) Connector2HopsPath: For each dominator v that are 2-hops apart from u, node u stores (w,v), where w

is selected by u to connect to v. (3) Connector3HopsPath: For each dominator v that are 3-hops apart from u, node u stores (w,x,v), where w is the node selected by u and x is selected and w to connect to v.

It has been shown in [16], [19], that the number of connectors found is bounded by a constant. It is also known that the size of the CDS formed is within a constant factor of the minimum size. Figure 3 show an example in practice.

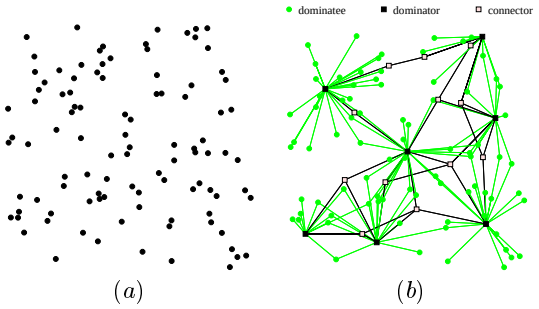

Fig. 3. (a) A set of wireless nodes. (b) constructed CDS.

C. Analysis

In [1], [2], [3], [4], they already showed their key agreement protocols are secure, which guarantees that our hybrid key agreement protocol is secure. Therefore, in this subsection, we concentrate on the analysis of the communication complexity of our hybrid key agreement protocol.

Notice that, in the original key agreement algorithm, they assume an existing order defined in the group members by requiring each member M_i send a message to M_{i+1}. In addition, they assume the communication cost between any two members M_i and M_{i+1} is always one unit. However, these assumptions do not hold anymore in this wireless ad hoc environment. The CDS nodes on the backbone are not connected to each other in any specific order, and a direct communication exists only between a node M_i and its 1-hop neighbors on the backbone. If M_{i+1} is not the 1-hop neighbor of M_i, then the communication from M_i to M_{i+1} is relayed by other intermediate nodes. However, we are able to give a method such that the total communications among the backbone is still $\Theta(n)$ although, for some pairs of M_i and M_{i+1}, the minimum hops to connect them is not bounded by a constant. Notice that, we can also use the tree-based key agreement protocol TGDH instead of GDH to generate the key for the backbone.

Assume that there are total n wireless nodes. After clustering process (Algorithm 4), there are g subgroups $G_1, G_2, ..., G_g$ with $n_1, n_2, ..., n_g$ members. In addition, assume Algorithm 5 finds k connectors to connect the dominators. It has been shown in [16], [19] that the construction of CDS costs $\Theta(n)$ total messages and the number of connectors k is $\Theta(g)$. Thus, the communication cost of the first step of algorithm 3 is $\Theta(n)$.

We first discuss the communication complexity of the third step of our hybrid key agreement protocol as it is simpler to analyze. Notice that in the third step, we could apply the contributory key agreement protocols or the key distribution protocols. Since the subgroup header (dominator) can reach all its subgroup members (dominatees) in 1-hop, the total communication cost of the third step is $\Theta(n)$.

We then study in detail the communication complexity of the second step in our hybrid key agreement protocol. We always assume that there is a leader selected already among all backbone nodes.

If TGDH protocol is used, we first construct a spanning tree for all backbone nodes. The tree can be constructed as follows. The leader sends a JOIN message to all of its 1-hop neighbors asking them to join the tree. Any node received the JOIN message for the first time replies and joins the tree, then sends a JOIN message to all of its 1-hop neighbors asking them to join the tree. Obviously, in this spanning tree, the communication between any parent node and its children is one unit as they are neighbors of each other. Consequently, if TGDH is used for key agreement among the backbone nodes, then the total communications of the third step is also $\Theta(n)$.

If the GDH protocol is used for the key agreement among backbone nodes, we first have to order the backbone nodes as $M_1, M_2, \cdots, M_{g+k-1}, M_{g+k}$, such that the total communication cost from M_i to M_{i+1}, $1 \leq i < g+k$, is linear. Notice that there are $g+k$ backbone nodes totally. For simplicity, let $c(M_i, M_{i+1})$ be the communication cost from M_i to M_{i+1} in the backbone, i.e., the number of hops connecting them. As using TGDH protocol, we first construct a spanning tree for all backbone nodes. We then order the backbone nodes using a postorder tree walk of the spanning tree, i.e., order the root after ordering all the subtrees rooted in its children. This ordering can be completed using $\Theta(n)$ messages by the following procedure. The root sends a message ORDERING(s) to a child u asking it to order the subtree rooted at u starting order with number s. Then node u returns the last number t used for ordering in that subtree to the root. The root sets $s = t$ and repeats the above procedure for other children if there is any.

We analyze the total communication cost $\sum c(M_i, M_{i+1})$ for the ordering derived by the above postorder method. More precisely, we show that it is at most $2(g+k)$. This is based on the following observation. The cost $c(M_i, M_{i+1})$ in the spanning tree is exactly the number of links visited by the postorder tree walk from M_i to M_{i+1}. It is well-known that the postorder tree walk visit every link of the spanning tree twice. Thus, the total communication cost $\sum c(M_i, M_{i+1})$ is 2 times the number of links in the spanning tree, which is exactly $g+k-1$. The above argument also applies to pre-order tree walk and in-order tree walk of the spanning tree. In other words, we can use pre-order, in-order, or postorder tree walk to order the backbone nodes.

V. Dynamic Maintenance

So far, we assumed that the wireless nodes are static or can be viewed static. This is not true in several applications. Wireless nodes will move around. The movement of wireless nodes makes it very difficult to design efficient protocols for various applications such as routing, backbone construction, and so on. In this subsection, we study in detail how our hybrid key agreement protocol can be easily adapted to node's movement.

If the node's movement does not cause the change of the network topology, it is obvious that no maintenance is necessary. Hereafter, we always assume that the network topology is changed due to node mobility. For simplicity, we assume that only one node is moving during a small time interval. We study in detail what is the effect of this node's movement.

A. Adding Node

We first show that our protocol is efficient in adding a new node u to the wireless ad hoc networks.

If u has a 1-hop neighbor v which is a dominator, u sends a message to join the subgroup of v, and marks itself as a dominatee. Member addition [4] is performed to get new group key.

Otherwise, u first sends a message to its 1-hop neighbors to claim itself as a new dominator. Then it will run Algorithm

5 to find some connectors to connect itself to the CDS. Apply member addition to add the new dominator and connectors to get the new key for the backbone. The subgroup of node u also needs run Algorithm 2 to agree upon a contributory key for this subgroup headed by u.

B. Removing Node

Removing a wireless node u from the ad hoc networks is also easy in our protocol. We assume that the removing of node u will not disconnect the network.

If u is a dominatee of node v, then v just apply member deletion [4] to get new group key for the subgroup of v.

Assume u is a connector of dominator v. Notice that u may be a connector for many pairs of dominators, but from the proof in [19] we know the number of the dominators v that u is connected to is bounded by a constant, which makes the linear communication cost remain. Every dominator v applies member deletion to get the new group key for its subgroup. Also it will run Algorithm 5 to find new connectors for itself. Applying member addition/deletion to add the new/old connectors to get the new key for CDS, if contributory key is always required for the backbone. Otherwise, dominator nodes can just tell the newly found connectors the common key of the backbone.

Assume node u is a dominator. Its 1-hop neighbors need to reselect their dominators by running the Algorithm 1 on these dominatees solely dominated by u. Then these new dominators will run Algorithm 5 to find new connectors. Apply member addition/deletion to add the new/old dominators and connectors to get the new key for CDS.

C. Node Movement

After discussing the adding and removing cases, we can easily deal with the situation in which a wireless node move from one cluster group to another cluster group. We can combine the first two cases: first remove that node from the original group then add it in the new cluster group.

VI. Conclusion

In this paper, we proposed an efficient hybrid key agreement protocol that is suitable for wireless ad hoc networks. Our protocol partitions the wireless nodes into a set of subgroups using the connected dominating set concept. One common key is generated for all backbone nodes and one common key is generated for each subgroup headed by the corresponding dominator. We separate the subgroup keys from the backbone key in favor of the easy maintenance of the group in mobile environment, and dynamic group membership. The communication cost of this protocol is $\Theta(n)$ when n is the number of wireless nodes. It is easy to maintain the group in a mobile environment. The cost of adding and removing a member could be as low as just a constant communications, and at most $O(\log n)$ communications. No update is needed if the network topology is not changed even though the nodes are moving.

References

[1] N. Asokan and Philip Ginzboorg, "Key-agreement in ad-hoc networks," in *Nordsec'99*, 1999, (To appear) in Computer Communication Review, 2000.

[2] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik, "Authenticated group key agreement and friends," in *ACM Conference on Computer and Communications Security*, 1998, pp. 17–26.

[3] Yongdae Kim, Adrian Perrig, and Gene Tsudik, "Tree-based group key agreement," Cryptology ePrint Archive, Report 2002/009, 2002.

[4] Michael Steiner, Gene Tsudik, and Michael Waidner, "Diffie-hellman key distribution extended to group communication," in *ACM Conference on Computer and Communications Security*, 1996, pp. 31–37.

[5] Michael Steiner, Gene Tsudik, and Michael Waidner, "CLIQUES: A new approach to group key agreement," in *International Conference on Distributed Computing Systems*, 1998, pp. 380–387.

[6] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[7] M BURMESTER, "On the risk of opening distributed keys," in *In Advances in Cryptology – CRYPTO '94*, 1994, Y. G. Desmedt, Ed., vol. 839 of Lecture Notes in Computer Science, Springer-Verlag, Berlin Germany, pp. 308–317.

[8] Yvo Desmedt and Mike Burmester, "Towards practical proven secure authenticated key distribution," in *Proceedings of the 1st ACM conference on Computer and communications security*. 1993, pp. 228–231, ACM Press.

[9] Whitfield Diffie and Martin E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, 1976.

[10] Douglas R. Stinson, *Cryptography: Theory and Practice*, CRC press, 1995.

[11] Raph C. Merkle, "Secrecy, authentication, and public-key cryptosystems," Technical Report No. 1979-1, Information Systems Laboratory, Stanford University, 1979.

[12] D. Wallner, E. Harder, and R. Agee, "Key management for multicast: Issues and architecture," IETF draft, July 1997. RFC 2627, June 1999.

[13] Amos Fiat and Moni Naor, "Broadcast encryption," in *CRYPTO, LNCS 773*, 1993, pp. 480–491.

[14] David A. McGrew and Alan T. Sherman, "Key establishment in large dynamic groups using one-way function trees," Manuscript, 1998, Submitted to IEEE Transactions on Software Engineering.

[15] Klaus Becker and Uta Wille, "Communication complexity of group key distribution," in *Proceedings of the 5th ACM conference on Computer and communications security*. 1998, pp. 1–6, ACM Press.

[16] Peng-Jun Wan, Khaled M. Alzoubi, and Ophir Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," in *INFOCOM*, 2002.

[17] Khaled M. Alzoubi, *Virtual Backbone in Wireless Ad Hoc Networks*, Ph.D. thesis, Illinois Institute of Technology, 2002.

[18] Khaled Alzoubi, Peng-Jun Wan, and Ophir Frieder, "Message-optimal connected-dominating-set construction for routing in mobile ad hoc networks," in *3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'02)*, 2002.

[19] Yu Wang and Xiang-Yang Li, "Geometric spanners for wireless ad hoc networks," in *Proc. of 22nd IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2002, To appear.

[20] S. Basagni, I. Chlamtac, and A. Farago, "A generalized clustering algorithm for peer-to-peer networks," in *Workshop on Algorithmic Aspects of Communication*, 1997.