Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2011, Article ID 382701, 15 pages
doi:10.1155/2011/382701

Research Article

Efficient Key Agreements in Dynamic Multicast Height Balanced Tree for Secure Multicast Communications in *Ad Hoc* Networks

Hua-Yi Lin[1] and Tzu-Chiang Chiang[2]

[1] *Department of Information Management, China University of Technology, Hsing-Chu 30301, Taiwan*
[2] *Department of Information Management, Tunghai University, Taichung 41349, Taiwan*

Correspondence should be addressed to Tzu-Chiang Chiang, steve312kimo@thu.edu.tw

Received 1 June 2010; Revised 14 September 2010; Accepted 21 November 2010

Academic Editor: Damien Sauveron

Alterations and unpredictability of the network topology in mobile *ad hoc* networks (MANETs) are less capable of ensuring the security of multicast data transmissions than in conventional networks. Despite the recent development of many key agreement protocols for MANETs, to our knowledge, only a few secure multicast data transmissions have been integrated into the key agreement. This study proposes a dynamic multicast height balanced group key agreement (DMHBGKA) that allows a user in a multicast group to efficiently and dynamically compose the group key and securely deliver multicast data from a multicast source to the other multicast group users in wireless *ad hoc* networks. The hierarchical structure of the proposed key agreement partitions the group members into location-based clusters capable of reducing the cost of communication and key management when member joins or leave networks. Moreover, based on elliptic curve Diffie-Hellman (ECDH) cryptography key management, the proposed scheme not only provides effective and efficient dynamic group key reconstructions and secures multicast data transmissions but also fits the robustness of the wireless networks and lowers overhead costs of security management.

1. Introduction

As an emerging paradigm of wireless communication for mobile nodes, *ad hoc* networks have received considerable attention in recent years due to a rapid expansion of wireless devices and the interest in mobile communications. In an *ad hoc* network [1–3], mobile nodes want to communicate with each other, but have no fixed links like a wire infrastructure network. While acting as a router, each node is responsible for discovering dynamically other nodes in a transmission range [4]. The emergence of *ad hoc* networks poses a challenge for maintaining the security of a group multicast since mobile *ad hoc* networks differ from conventional wired networks. Security is thus a priority concern in wireless networks, especially for security-sensitive applications. Computer security attributes of confidentiality, integrity, availability, authentication, and nonrepudiation are crucial to protect communications in *ad hoc* networks. Moreover, the network topology of an *ad hoc* network changes frequently and unpredictably, explaining why security is extremely challenging in routing and multicasting. In practice, establishing a trusted entity referred to as a certification authority (CA) by using a single node in *ad hoc* networks is a rather complex task. For an unavailable or compromised CA due to a vulnerable network structure, the entire secure communication cannot access the public keys of other nodes [5–8].

Many security protection schemes have been developed for an individual multicast group. Some schemes address single-security-level group communications by using Diffie-Hellman algorithm extending contributory key management and logical key hierarchy [9]. While describing how a multicast group user can compose a group key, this study presents a hierarchical group key management to multicast data from a multicast source to the remaining multicast members securely. We hypothesize that capable of acquiring the measures, that is, latitude, longitude, and altitude, from global positioning system (GPS) mobile nodes have a hierarchical structure. Additionally, group members are partitioned into location-based clusters to reduce the cost of key management. Moreover, encryption and decryption operations are presented for secure multicast communications.

The rest of this paper is organized as follows. Section 2 introduces the related security aspects of secure multicast communications in *ad hoc* networks. Section 3 then presents a secure multicast key agreement. Next, Section 4 introduces the proposed dynamic multicast height balanced group key agreement (DMHBGKA) scheme and the process of rekeying for participating and departing nodes. Additionally, Section 5 discusses secure multicast communication operations. Section 6 summarizes the simulation and analytical results for the proposed scheme. Conclusions are finally drawn in Section 7, along with recommendations for future research.

2. Related Security Aspects of Secure Multicast Communications in *Ad Hoc* Networks

The role of multicasting as a scalable solution for group communication in MANETs has ushered in the development of many group key management approaches. While those schemes normally focus on improving security and reducing the size of group keys, forward and backward confidential information should also be provided for multicast applications whenever a user joins or leaves the system. Kim et al. [7] developed a tree-based group key agreement scheme by using a binary tree infrastructure to compute and update a group key efficiently. That study also completed secure and distributed protocols by exploiting the group Diffie-Hellman (GDH) key exchange. Vasudevan and Sukumar [10] developed a scalable secure multicast algorithm by using a multiserver approach when the data encryption key (DEK) had to be changed. To minimize the rekeying cost, their schemes utilize the dynamic split and merge with a low overhead cost, where a physical server splits and merges its traffic into multiple groups, with each group served by a dedicated server. Wang et al. [11] developed a hybrid group key management scheme with a two-level structure where the group users are subdivided into clusters, subsequently reducing the rekeying cost as key updating. While developing a scheme that ensures key and data authenticity among group members, Chiang and Huang [12] demonstrated the data confidentiality of group messages with the properties of forward and backward confidential information. The group key is established collaboratively by combining the keys of all authenticated members, which assists in maintaining the communication and computation transparency among group members. Chaddoud et al. [13] divided group members into several operation units to perform microkey management. Compared with the logical key hierarchy (LKH), the above schemes can more significantly reduce the overload of the key server and provide more efficient key management for a secure wireless multicast. However, the above schemes lack efficient key management mechanisms for members to participate in or leave MANETs dynamically.

Despite the considerable attention paid to grouping or clustering issues for reducing traffic overhead and broadcast storm problems of MANETs, reducing the rekeying costs in key updating and increasing the key management efficiency have seldom been addressed in group key management schemes that focus on clustering issues. Clustering algorithms for MANETs have been developed to reduce communication costs between mobile nodes. Even numerous mobile nodes require only a few cluster headers to manipulate wireless communications.

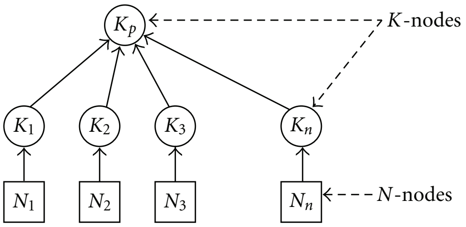


Figure 1: An acyclic key graph.

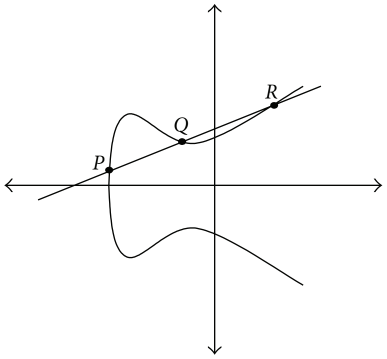


Figure 2: A elliptic curve with $P + Q + R = O$.

Our previous work developed a key-distribution graph model by using the Prüfer decoding algorithm for secure multicast communications in MANETs [12]. A key graph is a directed acyclic graph G with two nodes, that is, leaf nodes (N-nodes) representing multicast-user nodes and K-nodes representing keys [13]. Each N-node representing a multicast-user node has one outgoing edge associated with the individual key of each user node. Each K-node has one or more incoming and outgoing edges. If only having incoming edges and no outgoing edge, a K-node is a root of the key graph. K_p-node denotes a group key held by each user in N. Moreover, a key-distribution graph specifies a secure group (N, K, P) as follows:

(1) each multicast-user node in G corresponds to a unique N-node,

(2) each individual key corresponds to a unique K-node,

(3) the group key K_p has a direct path from all K-nodes.

For instance, the key graph in Figure 1 specifies the following secure group:

$$N = \{N_1, N_2, N_3, \dots, N_n\},$$
$$K = \{K_1, K_2, K_3, \dots, K_n\},$$
$$P = \{K_p\}.$$

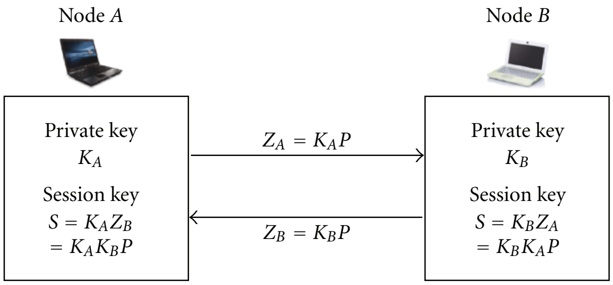


Figure 3: ECDH key agreement protocol.

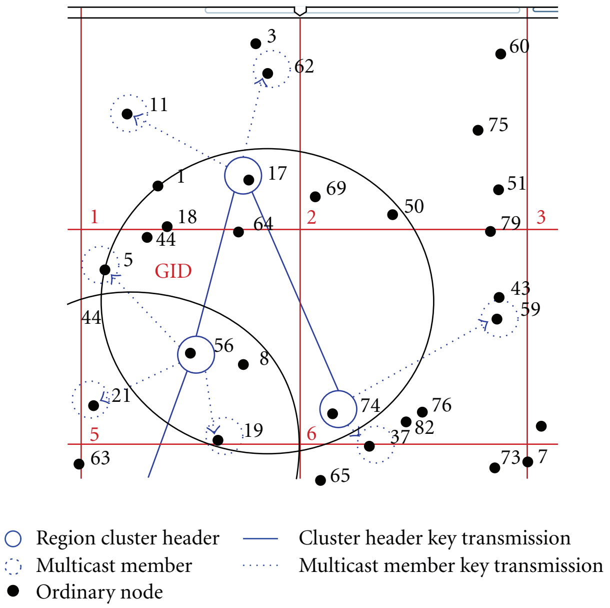


Figure 4: Graph for notations.

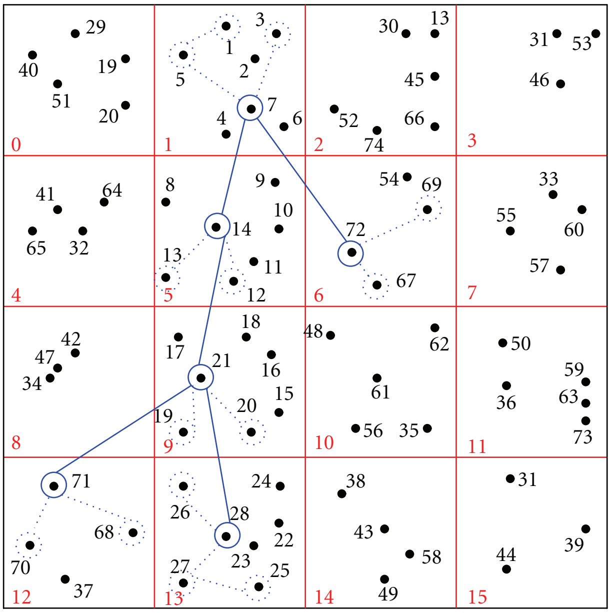


Figure 5: A multicast group in MANETs.

3. Secure Multicast Key Agreement

This section introduces multicast key management schemes and maintenance concepts that provide location-based multicast secure communications by using elliptic curve Diffie-Hellman agreement and geographic position information to deliver packets to multicast trees securely.

3.1. Elliptic Curve Diffie-Hellman Key Management Agreement. Since MANETs have limited resources, many security schemes provide high security level functions, such as asymmetric key and public key infrastructure (PKI), but they need a lot of resources; therefore, mobile networks cannot perform the security functions very well. To date, several studies have adopted elliptic curve Diffe-Hellman-(ECDH-) based security methods for networks, such as the studies by Sklavos and Zhang [14], Szczechowiak et al. [15], and Liu and Ning [16]. Sklavos and Zhang developed a hardware design and architecture for elliptic curve cryptography (ECC). Szczechowiak et al. investigated the ECC boundary and proved that public key cryptography was practical for wireless networks. Liu and Ning generated an implementation library and an executable package for ECC.

This session briefly introduces the ECC and ECDH schemes [17] for implementation in this study. Table 1 compares the security levels of common cryptographic key lengths. Smaller key size 160-bit in the ECC performs comparable security levels to 1024-bit RSA. The ECC has efficient operation and is indeed practicable for wireless networks with limited resources.

An elliptic curve is topologically equivalent to a torus over a finite field GF (a Galois field of order p), as shown in Figure 2 and comprises a set of finite points (x_i, y_i), where coordinates x_i, y_i are integers and satisfy

$$y^2 + a_1 xy + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6. \quad (1)$$

The coefficients a_i are elements in GF(p), since the field GF(p) ($p \in$ prime) is generally adopted in cryptographic applications, such that the elliptic curve in (1) can be translated into $E_p(a, b)$

$$y^2 = x^3 + ax + b \ (\text{mode } p), \quad (2)$$

where a and b belong to GF(p). Considering two points on curve $P = (x_1, y_1)$ and $Q = (x_2, y_2)$, and a point at infinity O, where $P \neq Q \neq O$, points P, Q and O satisfy the following rules:

(1) $P + O = O + P = P$, $P + (-P) = O$,

(2) $(x_1, y_1) + (x_1 - y_1) = P + (-P) = O$,

(3) $P + Q = R = (x_3, y_3)$ on the curve, where $x_3 = \lambda_2 - x_1 - x_2$, $y_3 = \lambda(x_1 - x_3) - y_1$, where

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q, \\[2ex] \dfrac{3x_1^2 + a}{2y_1} & \text{if } P = Q. \end{cases} \quad (3)$$

Table 1: Comparison of key length for ECC and RSA.

Security level	Symmetric key length (bits)	ECC key length (bits)	RSA/DH/DSA key length (bits)	ECC/RSA key size ratio	MIPS years time to break key
2^{80}	80	160	1024	1/6	10^{12}
2^{112}	112	224	2048	1/9	10^{24}
2^{128}	128	256	3072	1/12	10^{28}
2^{192}	192	384	7680	1/20	10^{47}
2^{256}	256	512	15360	1/30	10^{66}

However, given points P and Q on the curve, if the discrete logarithm of Q to base P, denoted as K, is large, then calculating the value of K where $PK = Q$ is infeasible. The ECC requires the elliptic curve discrete logarithm problem being simple to solve.

The elliptic curve Diffie-Hellman (ECDH) is a variant of the Diffie-Hellman (DH) key agreement protocol, using elliptic curve cryptography that allows two parties to establish a shared secret key (session key) over an insecure channel. Two parties then exploit this key to encrypt subsequent communications using a symmetric key scheme. The ECDH with 160-bit key lengths provides the same security level to a 1024-bit DH secret sharing protocol [15, 16]. However, the original DH protocol needs a key of at least 1024 bits to achieve adequate security; therefore, it requires high CPU and memory capabilities to perform exponential operations. Unfortunately, mobile nodes with limited resources have insufficient power to handle the overhead. Therefore, ECDH is quite suited for MANETs.

Consider the case in ECDH, where mobile node A wants to establish a shared key with node B, as shown in Figure 3. The public parameters (a prime p, a base point P as a generator in Diffe-Hellman, coefficients a and b, and elliptic curve $y^2 = x^3 + ax + b$) must first be set. Additionally, each party must have an appropriate key pair for elliptic curve cryptography, comprising an ECC private key K (a randomly selected integer) and a public key Z (where $Z = KP$). Let a node key pair of A denote (K_A, Z_A), and a node key pair of B denote (K_B, Z_B). Each party must have the other party's public key. Node A calculates $Z_A = K_AP$, while node B calculates $Z_B = K_BP$. Both parties calculate the shared key as $S = K_AZ_B = K_AK_BP = K_BK_AP = K_BZ_A$. The protocol is secure because it reveals nothing (except public keys, which are not secret), and because no party can calculate the private key of the other unless it can solve the Diffie-Helman problem (DHP) [18]. ECDH scheme is suited for *ad hoc* networks with limited resources. Each node only needs a few operations to achieve compatible security levels on RSA or Diffee-Hellman. This study exploits ECDH on group-based key managements and secure data transmission mechanisms and proposes a dynamic multicast height balanced group key agreement to achieve effective and efficient key synchronization, even though nodes dynamically participate in and depart from the wireless network.

3.2. The Clustering Scheme for Choosing Cluster Head. This section describes the selection steps for cluster headers in the location-based multicluster architecture shown in Figures 4 and 5. The clustering scheme partitions a large group into a hierarchy of recursively organized subgroups based on a distributed geographic hashing method. A mobile node wanting to join a multicast group takes x and y coordinates as inputs of a hash function and then outputs a unique region ID. This node subsequently sends a HELLO message, including the region ID, x and y coordinates. In the same region, the fact that each node with a unique ID realizes x and y coordinates of its one-hop neighbors allows it to determine which one has the shortest distance to the center of the wireless network area. The node with the shortest distance is selected as a cluster head and then broadcasts a cluster message to the remaining nodes. Following the clustering phases, the system determines 16 clusters in this system, that is, $0, 1, \ldots, 15$. Each cluster head subsequently exploits the proposed DHBGKA scheme to generate a group key (GK_i) for each cluster member to ensure secure multicast communications.

4. Dynamic Multicast Height Balanced Tree

In *ad hoc* networks, mobile nodes join or leave networks dynamically, necessitating that the system performs group key reconstructions frequently. This work presents a dynamic multicast height balanced group key agreement (DMHBGKA) to achieve dynamic multicast key management. The DMHBGKA tree has the following attributes.

(1) DMHBGKA tree is a special binary search tree in which the subtrees of each node differ in height by at most one. Additionally, each subtree is a DMHBGKA tree, as shown in Figure 6.

(2) Balance factor (BF) denotes the height difference of left and right subtrees, while $\mathrm{BF} = |H_L - H_R| \leqq 1$, where H_L denotes the height of a left subtree, and H_R denotes the height of a right subtree.

(3) A node joining or leaving networks leads to a tree unbalance. The proposed DMHBGKA scheme adjusts procedures to rebalance the tree. The procedures are classified into categories of left rotation (LL), left-right rotation (LR), right rotation (RR), and right-left (RL) rotations. The procedure is adjusted as follows.

Step 1. According to the binary search tree rule, place (or remove) the new joining (or leaving) node in (or from) the correct place, depending on its ID (MAC or IP address).

Step 2. Calculate the BF of each node, which belongs to $(0, -1,$ or $1)$. If not, the DMHBKA tree loses balance.

Step 3. Adopt LL, RR, LR, and RL mechanisms to perform unbalanced adjustments.

Step 4. Reconstruct the balanced DMHBGKA tree.

A node joins or leaves the networks. The time complexity associated with searching the target node is $O(\log n)$ (n denotes the number of nodes); the system only needs to modify the link point of the data structure and thus takes $O(1)$ time complexity. As the DMHBGKA tree is unbalanced, in a worst case scenario, the adjusting procedure must move a leaf node from the bottom to the root position and at most takes $O(\log n)$. Given that the DMHBGKA tree is effective and efficient for dynamic mobile networks, this study exploits DMHBGKA to manage the dynamic group as described in detail in the following.

4.1. Dynamic Multicast Height Balanced Group Key Agreement—DMHBGKA. First, based on the node's ID (MAC or IP), this study utilizes the binary search tree algorithm to locate the node in the DMHBGKA tree, as shown in Figure 7. The system then performs ECDH key management agreement from leaf nodes to the root node. Initially, leaf nodes 1 and 3 perform ECDH to obtain the session key K_1K_3P. Nodes 1 and 3 as well as their parent node 2 then calculate the subgroup key $K_1K_2K_3P$ cooperatively. Next, nodes 5 and 7 perform ECDH to obtain the session key K_5K_7P. Nodes 5 and 7 as well as their parent node 6 calculate the subgroup key $K_5K_6K_7P$ cooperatively. By using the same procedure, nodes 2 and 6 obtain $K_1K_2K_3K_5K_6K_7P$ and then deliver it to node 4. Root node 4 then determines the group key $K_1K_2K_3K_4K_5K_6K_7P$ for this tree. Consequently, the root node 4 unicasts securely the group key to each node.

Figure 8 shows that new nodes 8 and 9 join the system. According to the DMHBKA agreement, nodes 8 and 9 are located in the right subtree, and then the adjusting procedure is performed to maintain the tree balance. The group key is subsequently calculated as $K_1K_2K_3K_4K_5K_6K_7K_8K_9P$. If the root node leaves, as shown in Figure 9, the system selects the largest ID node from either the left subtree or the smallest ID node from the right subtree to replace the root node. Nodes 1 and 2 subsequently leave, with the system performing the adjustment procedure and recalculating the new group key as $K_3K_5K_6K_7P$ in Figure 9.

The proposed mechanism identifies the joining or leaving node in $O(\log n)$, and only needs to recalculate the key value from the part of the joining (leaving) node subtree without recalculating the entire tree, thus saving a tremendous amount of operational time. The proposed approach is effective and efficient, and the DMHBGKA algorithm is shown in Algorithms 1 and 2 :

4.2. Interregion Key Exchange Agreement. As the multicast data cross different regions, this study proposes a region key mechanism to secure the transmitted data between regions. For instance, the multicast path is region 1 → region 5 → region 9 → region 13, as shown in Figure 10. Each pair of root nodes must calculate the interregion key between them using the ECDH agreement. Following calculations, this study derives the interregion key $K_1K_2K_3 \cdots K_{11}K_{12}K_{13}K_{14}P$ for CH_7 in region 1 and CH_{14} in region 5. The interregion key $K_8K_9K_{10} \cdots K_{19}K_{20}K_{21}P$ is for CH_{14} in region 5 and CH_{21} in region 9. The interregion key $K_{15}K_{16}K_{17} \cdots K_{26}K_{27}K_{28}P$ is for CH_{21} in region 9 and CH_{28} in region 13. Subsequently, the source node and destination node exploit the group and interregion keys to perform secure multicast communications.

5. Secure Multicast Communications

This section describes the secure operations for multicast communications in MANETs. Figure 10 presents a multicast group and tree, and a multicast source node N_1 allocated on region 1 is assumed here to want to transfer data to all multicast members which are drawn in dotted circles. For simplicity, a description is made of the encryption and decryption operation of secure multicast from multicast source node N_1 to destination node N_{25}, that is, one of the multicast members in region 13.

This work assumes that the multicast tree is generated by the multicast source and the path from multicast source to destination node N_{25} is known. To distinguish between the cluster groups, this study transfers the entire range of the wireless network into a geographical position. The cluster headers are responsible for the secure multicast backbone transmission.

When the multicast source node N_1 wants to transfer multicast data to the destination node N_{25}, N_1 is located in region 1 and belongs to the cluster header node N_7. First, secure communications must be ensured between node N_1 and the cluster header node N_7 belonging to the multicast backbone network.

To ensure data integrity issues, this study adopts hash message authentication code (HMAC) functions to generate HMAC(data) and aggregate HMAC(data) with original multicast data as [data|HMAC(data)]. Secure multicast communication procedures are described in detail as follows:

$$N_1 \longrightarrow N_5$$
$$\mathrm{EK}_{\mathrm{GK1}}[\mathrm{EK}_{K1K25P}[\mathrm{data}|\mathrm{HMAC}\ (\mathrm{data})]].$$

Initially, N_1 and N_{25} cooperatively calculate their session key $K_1K_{25}P$ along the multicast backbone, and then N_1 encrypts the [data|HMAC(data)] using $K_1K_{25}P$ as EK_{K1K25P}[data|HMAC(data)]. Additionally, the member node is located in the same region and has the same group key $\mathrm{GK}_1 = K_1K_2K_3K_4K_5K_6K_7P$. Therefore, the multicast resource node N_1 located in region 1, as shown in Figure 10, encrypts $[\mathrm{EK}_{K1K25P}$[data|HMAC(data)]] using GK_1 to ensure the security of transmitted data in region 1. This operation ensures that the nodes in adjacent regions 0, 2, 4, 5, and 6 cannot decrypt the encrypted data from the multicast source node .

Subsequently, N_1 sends the encrypted data to the next node N_5. After receiving the data, N_5 decrypts the encrypted

```
{
  if( current == null ) // insert null node
  { current = new DMHBGKA_Node(value, null, null); }
  else if(value < current.value) // less than current node value/
  { current.left = DMHBGKA _Insert(value, current.left);
    if(height(current.left) - height(current.right) == 2)
    // unbalance occurs//
    {if(value < current.left.value)
      {current = Rotate_Left_Child(current); //LR
      Groupkey_Reconstruction( rootnode, current) //from current node to root }
    else
      {current = Doublerotate_With_Left_Child(current); //LL
      Groupkey_Reconstruction( rootnode, current) //from current node to root//}
    }
  }
  else if(value > current.value)// it is greater than current node//
  {current.right = DMHBGKA _Insert(value, current.right);
    if(height(current.right) - height(current.left) == 2)
    // there is an imbalance//
    { if(value > current.right.value)
      {current = Rotate_Right_Child(current); //RL
      Groupkey_Reconstruction( rootnode, current) //from current node to root// }
    else
      {current = doublerotate_Right_Child(current); //RR
      Groupkey_Reconstruction( rootnode, current) //from current node to root// }
    }
  }
  current.height = Math.max(height(current.left),
  height(current.right)) + 1;
  return current;
}
```

Algorithm 1: DMHBGKA_Insert (int value, point current).

data $\mathrm{EK}_{\mathrm{GK}1}[\mathrm{EK}_{K1K25P}[\mathrm{data}|\mathrm{HMAC}(\mathrm{data})]]$ using GK_1. Subsequently, N_5 encrypts the $[\mathrm{EK}_{K1K25P}[\mathrm{data}|\mathrm{HMAC}(\mathrm{data})]]$ using GK_1 again, and deliveries them to the cluster head CH_7

$$N_5 \longrightarrow \mathrm{CH}_7$$
$$\mathrm{EK}_{\mathrm{GK}1}[\mathrm{EK}_{K1K25P}[\mathrm{data}|\mathrm{HMAC}(\mathrm{data})]].$$

Once the encrypted data are received, since N_5 and CH_7 are located in the same region and have the same group key GK_1. CH_7 can decrypt the encrypted data. Subsequently, CH_7 must deliver the data to cluster head CH_{14} in region 5. Since the transmission data cross different regions, thus CH_7 and CH_{14} cooperatively calculate the interregion key $\mathrm{RK}_{1,5} =$

```
{
if( node == null)
{System.out.println(del_val +“Not found in DMHBGKA Tree\n”); return null;}
else
{ // search for del_val to be deleted//
if(node.value < del_val)
{node.right = DMHBGKA _Remove( del_val, node.right); }
else if( node.value > del_val)
{ node.left = DMHBGKA _Remove( del_val, node.left);}
// del_val found, delete if a descendant is null //
else if( node.left == null)
{ node = node.right;}
else if( node.right == null)
{node = node.left; }
//no descendant is null, rotate on heavier side//
else if( height( node.left ) > height( node.right ))
{ node = Rotate_Right_Child(node); //RL
node.right = DMHBGKA _Remove( del_val, node.right );
Groupkey_Reconstruction( rootnode, current) //from current node to root//}
else
{ node = Rotate_Left_Child( node ); //LR
node.left = DMHBGKA _Remove( del_val, node.left );
Groupkey_Reconstruction( rootnode, current) //from current node to root//}
//reconstruct weight information //
if( node != null )
{node.height = height( node.left ) + height( node.right );}
}
return node;
}
```

Algorithm 2: DMHBGKA_Remove (int del_val, point node).

$K_1K_2K_3\cdots K_{14}P$, and then CH_7 encrypts $[\mathrm{EK}_{K1K25P}[\text{data} \mid \text{HMAC(data)}]]$ using $\mathrm{RK}_{1,5}$ as $[\mathrm{EK}_{\mathrm{RK}1,5}[\mathrm{EK}_{K1K25P}[\text{data} \mid \text{HMAC(data)}]]]$. Subsequently, CH_7 sends the encrypted data to the backbone cluster head CH_{14}

$$\mathrm{CH}_7 \longrightarrow \mathrm{CH}_{14}$$

$$\mathrm{EK}_{\mathrm{RK}1,5}[\mathrm{EK}_{K1K25P}[\text{data} \mid \text{HMAC(data)}]].$$

CH_{14} receives the transmitted data and decrypts them using $\mathrm{RK}_{1,5}$. Subsequently, CH_{14} encrypts the received data using $\mathrm{RK}_{5,9} = K_8K_9K_{10}\cdot\cdots\cdots K_{19}K_{20}K_{21}P$ as $[\mathrm{EK}_{\mathrm{RK}5,9}[\mathrm{EK}_{K1K25P}[\text{data} \mid \text{HMAC(data)}]]]$, and sends them to the next cluster head CH_{21}. By repeating the above procedures, the encrypted data are transmitted to cluster head CH_{28} allocated in region 13.

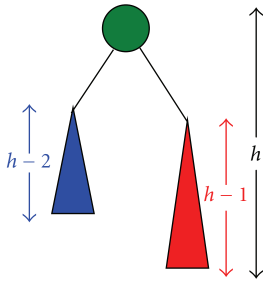


Figure 6: Recursive definition of height balanced binary search tree.

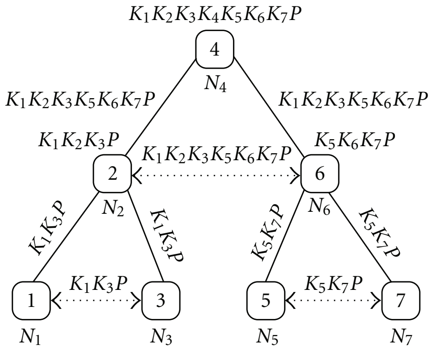


Figure 7: Group key exchange agreement.

The cluster head CH_{28} is responsible for transmitting the received data to the destination node N_{25}. Since CH_{28}, N_{27} and N_{25} are located in the same region; therefore, they have the same group key. CH_{28} adopts the group key $\text{GK}_{13} = K_{22}K_{23}K_{24} \cdots K_{28}P$ to encrypt the received data and transmits the encrypted data to the intermediate node N_{27}

$$\text{CH}_{28} \longrightarrow N_{27}$$
$$\text{EK}_{\text{GK13}}[\text{EK}_{K1K25P}[\text{data}|\ \text{HMAC}(\text{data})]].$$

Upon receiving the transmitted data, N_{27} decrypts the encrypted data using GK_{13}, encrypts the results using GK_{13}, and sends the encrypted data to the destination node N_{25}

$$N_{27} \longrightarrow N_{25}$$
$$\text{EK}_{\text{GK13}}[\text{EK}_{K1K25P}[\text{data}|\text{HMAC}(\text{data})]].$$

Upon receiving the encrypted data, N_{25} decrypts them using GK_{13} and session key $K_1K_{25}P$, and verifies the integrity of HMAC(data). If any changes take place during the transmissions, the receiving node detects the modifications immediately by verifying the HMAC. Thus, the proposed secure multicast schemes satisfy the following security analyses.

(1) Confidentiality and Authentication. During the data transmission, this study exploits the group and interregion keys to encrypt the multicast data. Only the node with the same group or interregion keys can decrypt the transmitted data. The other nodes are not aware of the group and interregion keys; therefore, cannot decrypt the encrypted data. Thus, the scheme can ensure that the data transmission is confidential and authentic.

(2) Data Integrity and Accuracy. This study employs message authentication code (HMAC) to verify the integrity of transmitted data. During the transmission, each node calculates HMAC, and the receiver verifies the integrity of HMAC. Since HMAC is an irreversible operation, given a random number y, no ways can compute x such that $H(x) = y$. Moreover, when $a \neq b$, then$H(a) \neq H(b)$. Therefore, if any nodes modify the transmitted data during transmissions, the receiver detects the unmatched HMAC instantly and recognizes the tampered data.

6. Analyses

6.1. Communication Cost Evaluation. The communication cost of *ad hoc* networks is an immensely complex problem [19]. The main complicity arises when attempting to consider irregular geographical distribution and any sources of interference (such as maintaining clusters, bandwidth, CPU, memory, and network traffic). This study adopts hop counts to evaluate communication costs, because this is the most widely used measure. In the proposed cluster-based models, it is logical to assume that *ad hoc* networks have $m \times n$ mobile nodes and are located on a 2D coordinate. These mobile nodes are allocated on the intersections as shown in Figure 11. This study attempts to compute the min-hop-count for any two nodes in the proposed model, for simplicity the following terms are defined.

N_{ab}: denotes a mobile node allocated on coordinate (a, b).

$\text{Min}_{hop}(N_{ab}, N_{cd})$: represents the minimal hop count between node N_{ab} and N_{cd}.

$\text{AVMin}_{\text{hop}}$: is the average minimal node-hop-count for any two nodes in this model.

$\text{AVCBMin}_{\text{hop}}$: denotes the average minimal cluster-hop- count for any two clusters in this model.

A: is a set containing $\{1, 2, \ldots, m\}$ or $A = \{1, 2, \ldots, m\}$.

B: is a set containing $\{1, 2 \ldots, n\}$ or $B = \{1, 2, \ldots, n\}$

$$\begin{aligned} V &= \sum_{a,c \in A; b \in B} \underset{\text{hop}}{\text{Min}}(N_{ab}, N_{cb}), \\ H &= \sum_{b,c \in B; a \in A} \underset{\text{hop}}{\text{Min}}(N_{ab}, N_{ac}), \\ R &= \sum_{a,c \in A; b,d \in B} \underset{\text{hop}}{\text{Min}}(N_{ab}, N_{cd}), \quad a \neq c,\ b \neq d. \end{aligned} \tag{4}$$

B_{ab}: represents the number of $a \times b$ grids in the model, where $a, b \neq 1$.

Generally, V represents the sum of Minhop between two nodes, which is parallel to the Y-axis. Meanwhile, H represents the sum of Minhop between two nodes, which is parallel to the X-axis. Furthermore, Bab represents the number of grids $a \times b$ in an $m \times n$ model. Finally, R is the

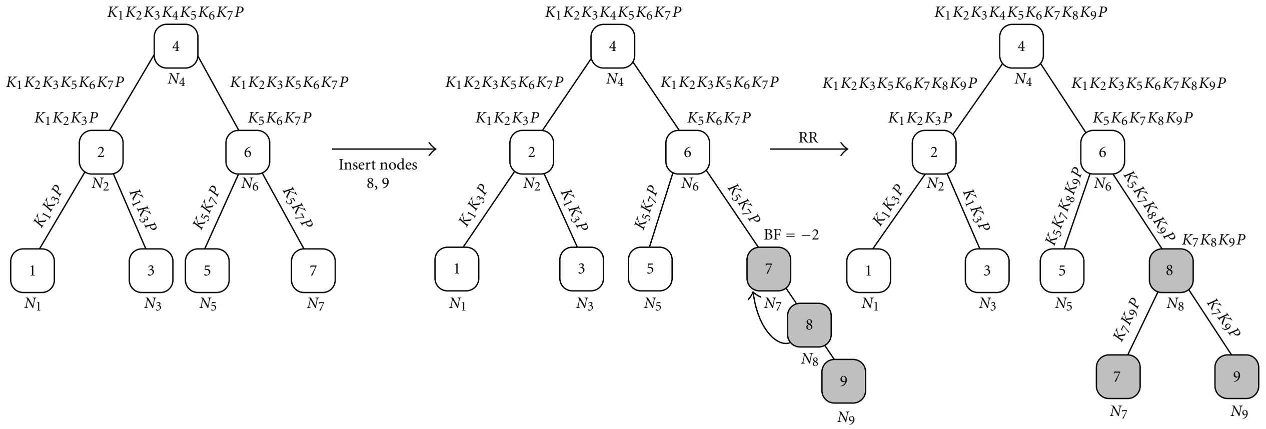


Figure 8: Nodes 8 and 9 join the network and perform RR adjusting procedure to recalculate the group key.

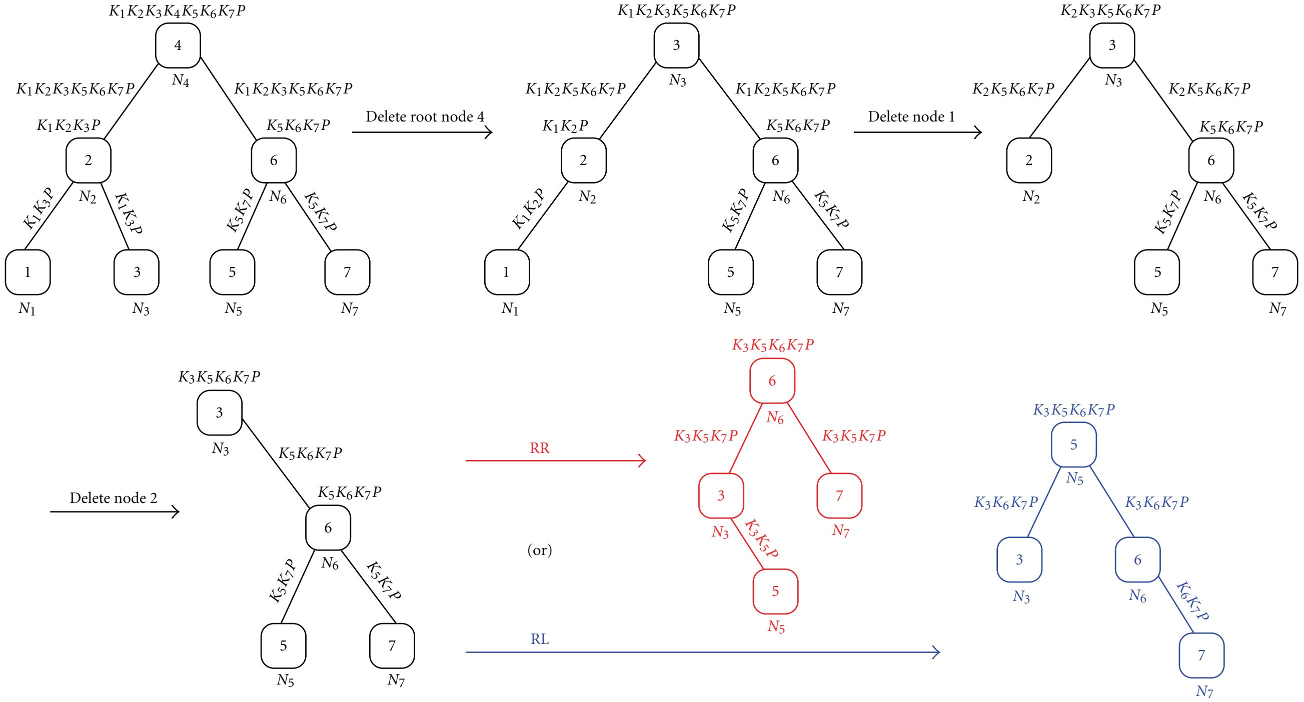


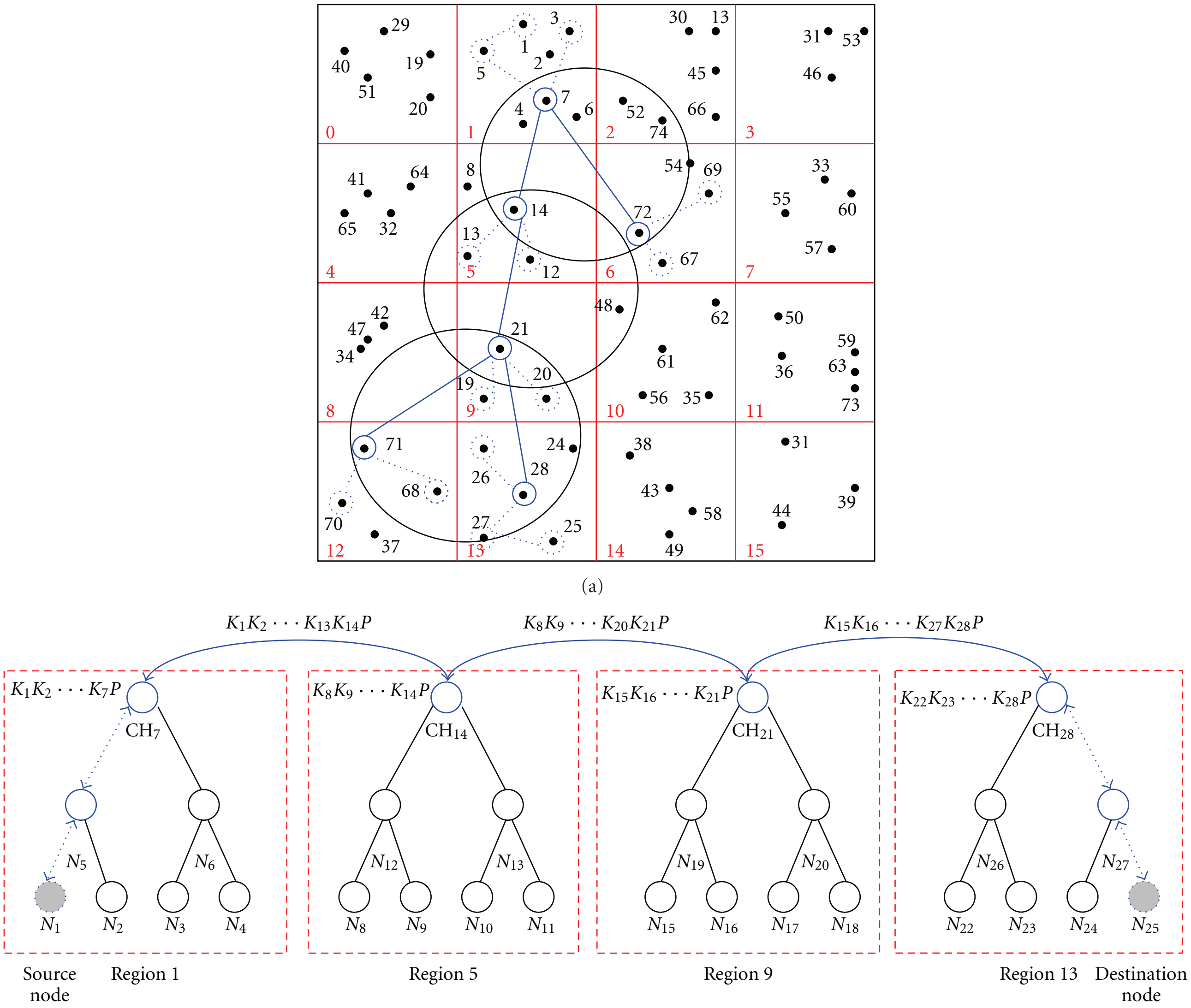
Figure 9: When root node, nodes 1 and 2, leave, the system performs adjusting procedures and recalculates the group key.

Minhop sum of all diagonal line pair nodes belonging to a grid $a \times b$. From the aforementioned terminologies, the following equations are established:

$$\begin{aligned} V &= n \cdot \sum_{i=1}^{m-1} (m-i)i = \frac{n(m-1)m(m+1)}{6}, \\ H &= m \cdot \sum_{i=1}^{n-1} (n-i)i = \frac{m(n-1)n(n+1)}{6}, \\ B_{ab} &= (m-a+1)(n-b+1), \\ R &= 2 \cdot \sum_{b=2}^{n} \sum_{a=2}^{m} B_{ab} \cdot (a+b-2) \\ &= \frac{n(n-1)m(m-1)(n+m+2)}{6}. \end{aligned} \tag{5}$$

From (5), The AVMin$_{\text{hop}}$ is determined to be $(m+n)/3$ as follows:

$$\begin{aligned} \text{AVMin}_{\text{hop}} &= \frac{(V+H+R)}{C_2^{nm}} \\ &= \left(\frac{nm(m^2-1)}{6} + \frac{nm(n^2-1)}{6} \right. \\ &\qquad \left. + \frac{n(n-1)m(m-1)(n+m+2)}{6} \right) / C_2^{nm} \\ &= \frac{(nm(n+m)(nm-1)/6)}{C_2^{nm}} \\ &= \frac{nm(n+m)(nm-1)/6}{nm(nm-1)/2} = \frac{m+n}{3}. \end{aligned} \tag{6}$$



(a)

(b)

Figure 10: Secure multicast data communication between nodes 1 and 25.

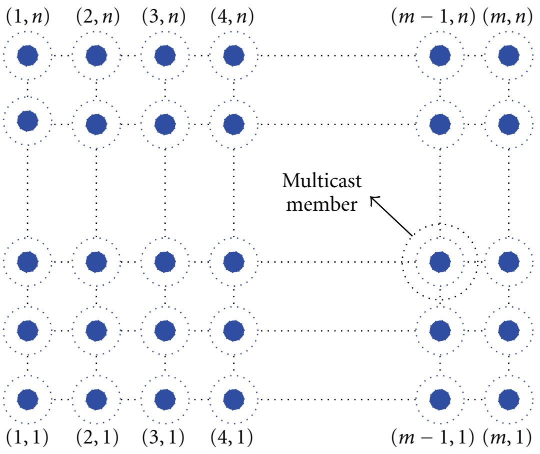


Figure 11: The mobile *ad hoc* networks model.

Applying (6), the cluster distribution model determined and the $\text{AVCBMin}_{\text{hop}}$ can be calculated for any two clusters. The cluster model is assumed to be denoted as an $a \times b$ model, and moreover to satisfy three conditions.

(1) Each cluster domain has the same number.

(2) Every cluster domain has $a \cdot b$ nodes; these nodes lie on $a \times b$ grid, and every intersection is only allocated a node.

(3) The gateways between two clusters are located on the boundary lines.

For clarity, an example is presented for explanatory purposes. Let the *ad hoc* networks be represented by a 7×5 model, while the cluster domain is a 4×2 model, each cluster allocates eight nodes, and the double bold lines represent one cluster domain as shown in Figure 12. The above model shows that the system is divided into eight cluster domains. This study realizes that cluster Ch1 comprises of node set $= \{N_{ij} \mid i = 1,2,3,4.\ j = 1,2\}$, and each cluster Ch$_i$ ($i = 1,2,3,\ldots,8$) has eight nodes. The gateway nodes between clusters Ch1 and Ch3 are the node set $= \{N_{12}, N_{22}, N_{32}, N_{42}\}$, which indicates the boundary between two rectangles. In an $m \times n$ network model with an $a \times b$ cluster model, $(a-1)$ divides $(m-1)$ and $(b-1)$ divides $(n-1)$. Since each cluster can be treated as a node, which represents a cluster head, then the set of all clusters can be represented as a $((m-1)/(a-1)) \times ((n-1)/(b-1))$ network model, and AVCBMin$_{\text{hop}}$ can be computed the same as AVMin$_{\text{hop}}$. From (6), AVCBMin$_{\text{hop}}$ $= ((m-1)/(a-1) + (n-1)/(b-1))/3$. For a 31×21 network model and a 4×3 cluster model, then from (6), AVMin$_{\text{hop}}$ $= (m+n)/3 = (31+21)/3 = 17.333$, and the average minimum cluster-hop-count for any two clusters AVCBMin$_{\text{hop}}$ is

$$\begin{aligned}&\frac{((m-1)/(a-1)+(n-1)/(b-1))}{3}\\ &\quad=\frac{((31-1)/(4-1)+(21-1)/(3-1))}{3}\\ &\quad=\frac{10+10}{3}=\frac{20}{3}=6.666.\end{aligned} \tag{7}$$

The ratio of the communication cost between any two cluster heads and any two nodes is AVCBMin$_{\text{hop}}$/AVMin$_{\text{hop}}$ $= 6.666/17.333$. Generalizing this equation under the $m \times n$ network model and the $a \times b$ cluster model, and it becomes

$$\begin{aligned}\frac{\text{AVCBMin}_{\text{hop}}}{\text{AVMin}_{\text{hops}}}&=\frac{((m-1)/(a-1)+(n-1)/(b-1))/3}{(m+n)/3}\\ &=\frac{((m-1)/(a-1)+(n-1)/(b-1))}{(m+n)},\end{aligned} \tag{8}$$

where $a, b \neq 1$. Generally, (8) is lower than 1. This calculation result implies that the cluster-based average minimum cluster-hop-count is below the normal-based minimum node-hop-count. That is, the cluster-based model outperforms the normal node-based model.

Considering the additional communication costs of node-based AVMin$_{\text{hops}}$ and cluster-based AVCBMin$_{\text{hop}}$, the node-based communication costs of two neighboring nodes are assumed to be λ, in which case the costs of cluster-based two neighbor clusters will be $\omega \cdot \lambda$. The ω denotes the nodes passing through between two neighbor clusters, which generally is $1 \leqq \omega \leqq \text{Max}(a,\ b)$. Since two clusters could pass through a gateway node or even multiple nodes when communicating with each other, the ω depends on the position of the cluster head and the cluster topology. Equation (8) then is generalized to be equivalent to

$$\begin{aligned}R&=\frac{((m-1)/(a-1)+(n-1)/(b-1))\cdot\omega\lambda}{(m+n)\cdot 1\lambda}\\ &=\frac{((m-1)/(a-1)+(n-1)/(b-1))\cdot\omega}{(m+n)}.\end{aligned} \tag{9}$$

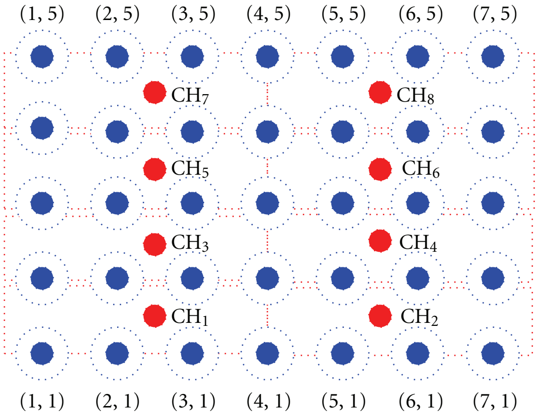


FIGURE 12: 7×5 ad hoc networks, 4×2 cluster domain.

If $a \geqq b \geqq 2$, then regardless of the values of m and n

$$\begin{aligned}R&\leq\frac{((m-1)/(b-1)+(n-1)/(b-1))\cdot\omega}{(m+n)}\\ &=\frac{((m+n-2)/(b-1))\cdot\omega}{(m+n)}<\frac{\omega}{(b-1)}.\end{aligned} \tag{10}$$

Furthermore, where $m, n \gg 1$ $(a, b \geqq 2)$, (9) is

$$R \cong \left(\frac{m(b-1)/(m+n)+n(a-1)/(m+n)}{(a-1)(b-1)}\right)\cdot\omega. \tag{11}$$

(1) When $m = n \gg 1$. Equation (11) is $R \cong (a+b-2)/(a-1)(b-1) \times \omega/2$.

(2) When $m \gg n,\ 1$. Then $n/(n+m) \rightarrow 0$, and $m/(n+m) \rightarrow 1$, (11) is $R \cong \omega/(a-1)$.

(3) When $n \gg m,\ 1$. Then $n/(n+m) \rightarrow 1$, and $m/(n+m) \rightarrow 0$, (11) is $R \cong \omega/(b-1)$.

From the result of the derivation, R (ratio of (AVCBMin$_{\text{hop}}$ − −communicative− −costs)/(AVMin$_{\text{hop}}$ − −communicative− −costs)) shows that the cluster-based approach outperforms the normal node-based one, at least in situations where the proposed model achieves better communication costs.

6.2. Computational Costs of DMHBGKA.

This section evaluates the performances of DMHBKA, tree-based group Diffie-Hellman (TGDH) and group Diffie-Hellman (GDH). Table 3 summarizes the computational costs for several operations.

Key Operations. As a node leaves or joins the multicast tree dynamically, the entire system must reconstruct the group key. However, the operation only affects a specific portion of the subtree of the joining (leaving) node. Therefore, in a worst case scenario, for a leaf node joining or leaving the tree, the system recalculates the group key at most h times, where h denotes the height of the tree. The system normally recalculates the new group key for i times, where i denotes the level of the joining (leaving) node. Generally, the system recalculates the group key from the inserted (deleted) node

Table 2: Simulation parameters.

CPU/Memory	3.2 GHz/1 G
Operation system	XP
Wireless protocol	802.11 b/g
Transfer rate	10 Mb/sec
Transmission range	100 m
Frequency	2.4 GHz
Simulation time	600 s
Max speed	20 m/s
Region area	1500 m × 1500 m
A point multiplication (SEC-160)	0.18 s
ECDH session	5.2 s

position at level i to the root node. Therefore, the number of exponential operations is $(1 + i)\ i/2$. However, in the GDH scheme, each node must recalculate its partial group key, thus taking more key operations than DMHBKA and TGDH do.

Node Operations. The average search time for the joining (or leaving) node in the DMHBGKA tree is$[\sum_{i=1}^{n} \text{level}(i)]/n$. Moreover, the system must only modify the data structure of the link point for a joining (or leaving) node, and therefore, it takes $O(1)$ time complexity. Additionally, the number of multicast member nodes in the DMHBGKA tree is at most $2^h - 1$ and the minimum is $\text{Fab}(h+2) - 1$. Nevertheless, GDH identifies a specific node at most n and the minimum is 1.

Communications. Data communication for a group key requires i rounds, while a node joins because each node sends one message from its level i to the root node. The root node then broadcasts the group key to each node in the tree. The total number of messages is $n + i + 1$. Although DMHBGKA broadcasts more rounds than GDH and TGDH do, the message size is significantly lower than that of GDH and TGDH.

6.3. Group Key Agreement Performance Evaluation. In this study, each node only conserves a session key to encrypt/decrypt data, and exploits a hash function, such as HMAC-160 or RIPEMD-160, to verify the integrity of transmitted data. However, hash functions and session key operations consume few resources during secure data transmissions. Consequently, the plain operations are highly suited for MANETs.

Additionally, several analyses of various key cryptosystems are performed. The experiments are implemented on a Windows XP platform with a 3.2-GHz Celeron(R) CPU, 1-GB of memory, and a GNU C/C++ Library. The nodes move within a fixed region area 1500 m × 1500 m, and node mobility is simulated based on a random waypoint mobility model. Each node moves toward a randomly selected location at a speed uniformly distributed between 0 and *max_speed* and then pauses for a configured time, before selecting another random location and repeating the process. The mobile nodes pause time has value of (0, 30 s, 60 s, 120 s, 300 s, and 600 s), with 0 representing constant mobility and 600s indicating a stationary network. In a simulated network of nodes (25, 50, 75, 100, and 150), simulations are for constant node speeds of (0 m/s, 5 m/s, 10 m/s, 15 m/s, and 20 m/s), using the above pause time, and where every node has performed several ECDH operations. The mobile node calculates a point multiplication in 0.18 s on SEC-160 curve [20] and 5.2 s to establish a session key on ECDH in the prime field. Key synchronization time and data transmission time on interregion infrastructures are evaluated. Moreover, the proposed ECDH secure multicast data transmission session is also simulated to measure the performance on interregion mode. Table 2 lists the simulation parameters.

The key length of a security level is expressed as the pair $\langle x, y\rangle$, where x denotes the private key length in DMHBGKA, and y denotes the private key length in DH/RSA [21]. DMHBGKA requires only multiplication operations whereas DH and GDH [22, 23] must perform exponential operations. Therefore, DMHBGKA takes much less computing time than DH and GDH. Results in Figure 13 indicate that DMHBGKA needs a shorter synchronization time than DH and GDH in $\langle 160\,\text{bits}, 1024\,\text{bits}\rangle$ and $\langle 224\,\text{bits}, 2048\,\text{bits}\rangle$. Additionally, Figure 13 describes the estimated synchronization time of the group key for DMHBGKA, DH and GDH schemes. According to this figure, DMHBGKA takes less time than DH and GDH to achieve group key convergence. Figure 14 compares the two regions in terms of the synchronization time of the interregion key. Simulation results indicate that the DMHBGKA schemes outperform GDH and DH ones. This advantage is owing to that each cluster head is responsible for its cluster member key controller. Therefore, each pair of the cluster heads must only exchange their own group keys, making it possible to synchronize the new interregion key between two clusters without recalculating keys of all members in two clusters. Figure 15 indicates that cluster schemes outperform noncluster schemes in terms of synchronization time of the group key. This study divides all members into cluster-2, cluster-4, and cluster-8 structures. The simulation increases the number of cluster members from 1 to 32 sequentially, which is accompanied by implementing a group key. According to Figure 15, the convergent time increases with the number of cluster members. Moreover, cluster-8 performs the best in terms of reconstructing the group key.

Figure 16 shows the resynchronization time of the group key for adding or removing a node. Simulation results indicate that the DMHBGKA scheme takes less resynchronization time for nodes joining or leaving than DH and GDH since the point multiplication takes considerably less time than exponential operations. Furthermore, DMHBGKA must only recalculate the partial key from the place of a joining or leaving node to the root node, yet it does not need to recalculate keys of all members, subsequently achieving synchronization efficiently.

Figure 17 compares DMHBGKA, DH, and GDH schemes in terms of secure multicast data transmission time. Simulation results indicate that the DMHBGKA scheme performs better than other ones in terms of passing through nodes.

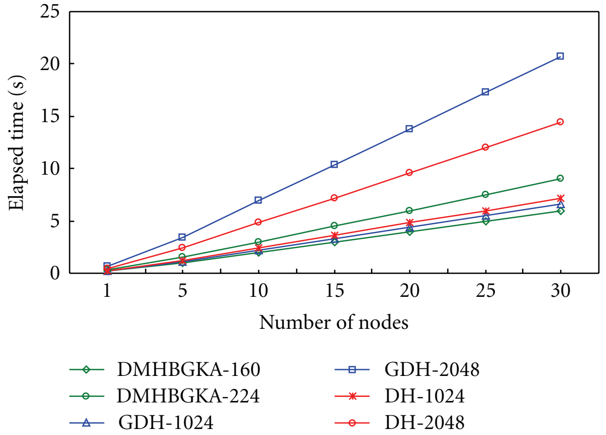


Figure 13: Group key synchronization time on DMHBGKA, GDH, and DH.

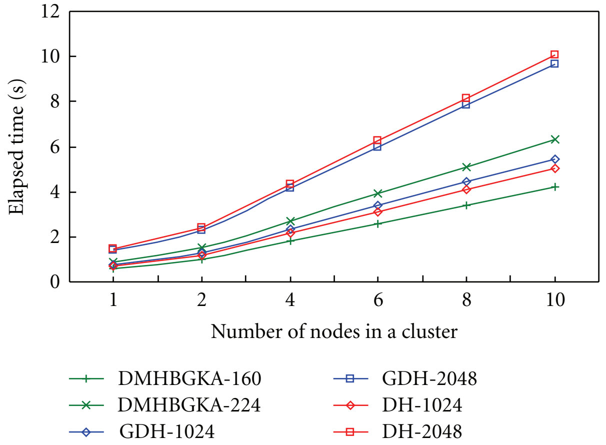


Figure 14: Synchronization time of the interregion key between two regions.

7. Conclusions and Future Work

This study presents a dynamic height balance group key agreement (DMHBGKA) scheme to ensure secure multicast data transmissions. The proposed scheme achieves the same security level as RSA and Diffe-Hellman do with shorter keys. Additionally, the proposed scheme performs very well for dynamic nodes joining or leaving. Simulation results indicate that in addition to consuming less system key synchronization time than other methods do, the proposed DMHBGKA scheme is also highly feasible for implementing constrained environments such as *ad hoc* networks.

Additionally, this study provides resilient and scalable mechanisms for dynamic group key management. The proposed scheme replaces exponential operations with point multiplications when performing ECDH, thus reducing the CPU overhead significantly. Therefore, the proposed scheme is highly promising for dynamic key operations in large-scale *ad hoc* networks. This study also presents a node in the multicast group, capable of exploiting the interregion key to deliver multicast messages securely from

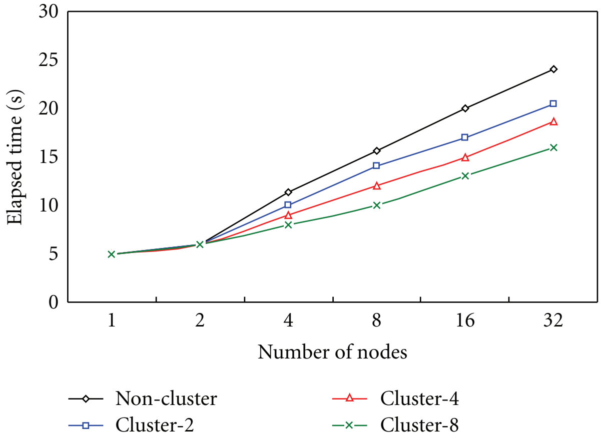


Figure 15: Convergent time of the group key for clusters versus nonclusters scheme.

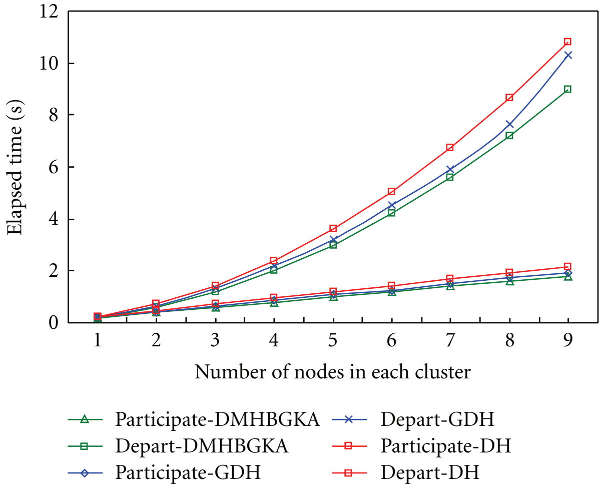


Figure 16: Resynchronization time of the group key for nodes joining or leaving.

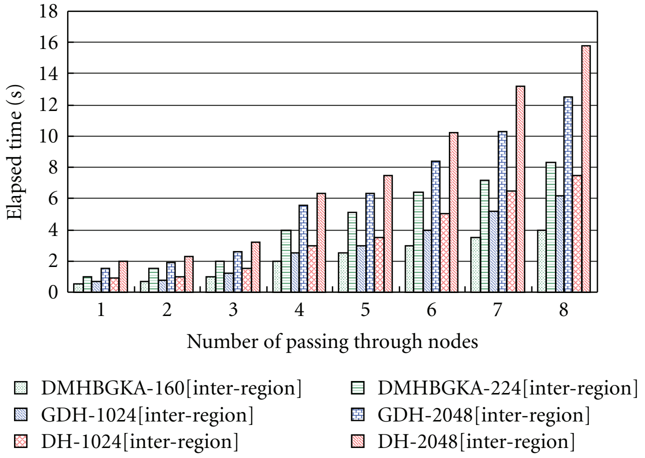


Figure 17: Comparison of secure multicast data transmission time.

Table 3: Computational costs.

		Key operations		Node operations			Communication		
N nodes height h Level i		Group key reconstruction	Exponentiations	Node traversal	Node joining/delete	The number of nodes	Rounds	Msgs sent per node	Total Msgs
DMHBGKA	Node joining	Max $= h$ Min $= i$	0	Average $= [\sum_{i=1}^{n} \text{level}(i)]/n$	Joining $= O(\log n)$	Max $= 2^h - 1$ Min $=$ Fab$(h+2) - 1$	i	1	$n + i + 1$
	Node leaving	Max $= h$ Min $= i$	0	Average $= [\sum_{i=1}^{n} \text{level}(i)]/n$	Delete $= O\ (\log n)$	Max $= 2^h - 1$ Min $=$ Fab$(h+2) - 1$	i	1	$n + i - 1$
GDH	Node joining	Max $= n$ Min $= 1$	$(n^2 + 3n)/2 - 1$	1	Joining $= O\ (1)$	Max $= n$ Min $= 1$	2	1	$n + 1$
	Node leaving	Max $= n - 1$ Min $= 1$	$(n^2 + 3n)/2 - 1$	1	Delete $= O(1)$	Max $= n$ Min $= 1$	1	1	$n - 1$
TGDH	Node joining	Max $= h$ Min $= i$	6 to $3h - 3$	i	Joining $= O(\log n)$	Max $= 2^h - 1$ Min $= h$	2	3	$2n + 1$
	Node leaving	Max $= h$ Min $= i$	3 to $3h - 3$	i	Delete $= O(\log n)$	Max $= 2^h - 1$ Min $= h$	1	1	$n - 1$

the multicast source to the internodes of the multicast tree. In addition to integrating the DMHBGKA scheme into secure multicast data transmissions, the proposed scheme provides secure communications among regions. Furthermore, the proposed approach utilizes ECDH key operations and a rapid hash function for secure multicast data transmission and data integrity verification, respectively, thus eliminating the requirement for complex operations on mobile nodes. Analytical results indicate that the proposed DMHBGKA scheme outperforms other methods in terms of rekeying performances, computation and communication costs, and overhead. Importantly, the proposed schemes are efficient and scalable for numerous mobile nodes in *ad hoc* networks.

References

[1] S. M. Das, H. Pucha, and Y. C. Hu, "Distributed hashing for scalable multicast in wireless ad hoc networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 3, pp. 347–362, 2008.

[2] B. Rong, H. H. Chen, YI. Qian, K. Lu, R. Q. Hu, and S. Guizani, "A pyramidal security model for large-scale group-oriented computing in mobile ad hoc networks: the key management study," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 1, pp. 398–408, 2009.

[3] W. He, Y. Huang, R. Sathyam, K. Nahrstedt, and W. C. Lee, "SMOCK: a scalable method of cryptographic key management for mission-critical wireless ad-hoc networks," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 1, pp. 140–150, 2009.

[4] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and R. Talebi, "On-demand location aware multicast (OLAM) for ad hoc networks," in *Proceedings of IEEE Wireless Communications and Networking Conference*, pp. 1323–1328, September 2000.

[5] N. C. Wang and S. Z. Fang, "A hierarchical key management scheme for secure group communications in mobile ad hoc networks," *Journal of Systems and Software*, vol. 80, no. 10, pp. 1667–1677, 2007.

[6] P. Yang and S. Zheng, "Security management in hierarchical ad hoc network," in *Proceedings of the International Conferences on Info-Tech and Info-Net*, pp. 642–649, October 2001.

[7] Y. Kim, A. Perrig, and G. Tsudik, "Tree-based group key agreement," *ACM Transactions on Information and System Security*, vol. 7, no. 1, pp. 60–96, 2004.

[8] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 16–30, 2000.

[9] M. Manulis, "Contributory group key agreement protocols, revisited for mobile ad-hoc groups," in *Proceedings of the 2nd IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS '05)*, pp. 811–818, November 2005.

[10] V. Vasudevan and R. Sukumar, "Scalable secure multicast using multi server approach for wireless environments," in *Proceedings of the International Conference on Control Automation, Communication and Energy Conservation (INCACEC '09)*, June 2009.

[11] Y. Wang, P. D. Le, and B. Srinivasan, "Hybrid group key management scheme for secure wireless multicast," in *Proceedings of the 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS '07)*, pp. 346–351, July 2007.

[12] T. C. Chiang and Y. M. Huang, "Group keys and the multicast security in ad hoc networks," in *Proceedings of the International Conference on Parallel Processing Workshops*, pp. 385–390, October 2003.

[13] G. Chaddoud, I. Chrisment, and A. Schaff, "Secure multicast survey," in *Proceedings of the 16th Word Computer Congress*, pp. 49–56, August 2000.

[14] N. Sklavos and X. Zhang, *Wireless Security and Cryptography: Specifications and Implementations*, CRC Press, Boca Raton, Fla, USA, 2007.

[15] P. Szczechowiak, L. B. Oliveira, M. Scott, M. Collier, and R. Dahab, "NanoECC: testing the limits of elliptic curve cryptography in sensor networks," in *Proceedings of the 5th European Conference on Wireless Sensor Networks (EWSN '08)*, pp. 305–320, February 2008.

[16] AN. Liu and P. Ning, "TinyECC: a configurable library for elliptic curve cryptography in wireless sensor networks," in *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN '08)*, pp. 245–256, April 2008.

[17] W. Haodong, S. Bo, and L. Qun, "Elliptic curve cryptography-based access control in sensor networks," *International Journal of Security and Networks*, vol. 1, no. 3-4, pp. 127–137, 2006.

[18] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[19] S. Basagni, I. Chlamtac, and A. Farago, "A generalized clustering algorithm for peer-to-peer networks," in *Proceedings of the Workshop on Algorithmic Aspects of Communication*, pp. 1–15, July 1997.

[20] Certicom Research, "Standards for Efficient Cryptography–SEC 1," *Recommended Elliptic Curve Domain Parameters*, September 2000.

[21] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, pp. 586–597, March 2004.

[22] G. Ateniese, M. Steiner, and G. Tsudik, "New multiparty authentication services and key agreement protocols," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 4, pp. 628–639, 2000.

[23] Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik, "On the performance of group key agreement protocols," in *Proceedings of the 22nd International Conference on Distributed Systems*, pp. 463–464, July 2002.