Energy-Aware Secure Multicast Communication In Wireless Ad-Hoc Networks

Loukas Lazos, and Radha Poovendran

***Abstract*— In this paper we address the problem of efficiently securing multicast communication in energy constrained wireless ad hoc networks. We show that existing scalable key distribution techniques that rely on logical hierarchies, are energy inefficient. We show that great energy savings can occur when physical location information is considered in the key distribution scheme construction. We also show that even greater energy savings occur when routing and physical layer information is exploited. We develop a location-aware key distribution scheme, with the use of an iterative clustering algorithm. We formulate an optimization problem for the key distribution tree construction and maintenance and show that any candidate optimal solution does not scale with group size. We alternatively propose a heuristic routing-aware algorithm for efficiently constructing and maintaining the key distribution tree. We further improve our algorithm by incorporating batch rekeying instead of individual rekeying after every membership change. We present simulation studies that illustrate the improvements achieved using our algorithms.**

***Index Terms*— key management, multicast, ad hoc, logical key tree, batch rekeying, energy-constrained**

I. Introduction

The idea of many personal devices being able to communicate over a short range without the need for a pre-deployed network infrastructure has recently become feasible. These devices typically consist of embedded microprocessors and are constrained by their limited computational power and battery life. Wireless networks of such type have many civilian applications such as dynamic communication for emergency-rescue operations and disaster-relief efforts, patient monitoring and drug inventory management in hospitals, as well as military applications such as deployment of surveillance networks and real time information distribution to moving army units. Most of these applications rely on the capability of peer-to-peer collaboration in computation and/or communication. The extension of group communication applications that pre-existed in the wired or cellular wireless networks, into the infrastructureless wireless ad hoc networks needs to be performed in a secure and efficient manner.

When an identical message has to be sent to multiple receivers, multicast communication model reduces the sender overhead as well as the network communication cost. A variety of applications that can be modeled as single-sender multiple-receiver, can benefit from multicast communication mode. In most of them, securing the communication channel is an inherent critical requirement. Secure multicast communication can be realized through encrypting the information that is transmitted over the open wireless channel.

The use of symmetric key cryptography allows the sender to perform one encryption (in broadcast mode) and every user to perform one decryption per message, thus reducing the computational and communication overhead. However, the use of a single key known to all members, requires its update each time a membership change occurs, in order to provide backward and forward traffic protection [1]. Backward traffic protection ensures that new members will not have access to previously transmitted information, while forward traffic protection ensures that revoked members will not have access to any information transmitted in the future. Since every member holds the data encryption key also known as session encryption key (SEK), when a member leaves the group, a secure channel to reach the remaining valid members for the update of the SEK is required. Hence, the group has to have additional keys called Key Encrypting Keys (KEKs) for distributing the new SEK [1].

The *key management* problem is to ensure that only valid members hold the valid keys at any time. The key management problem can be reduced to the *key distribution* problem, which involves the secure and efficient distribution of the SEK and the KEKs to valid members. When the communication group is fairly large and dynamic, the key distribution solution has to be scalable with the group size. In the case of wired networks, the rooted tree based hierarchical key distribution schemes are known to have scalable performance in terms of communication and storage cost [1], [3]. The communication cost is measured as the number of messages required for updating the SEK in case of a membership change, while the storage cost is measured as number of keys that need to be stored by each user, and/or the group communication initiator which operates as a key server, also known as the Group Controller (GC).

In a wireless environment where portable electronic devices with limited battery resources are communicating in an ad hoc mode, the energy expenditure for sending a message depends on the distance between the sender and the receiver. Hence, messages with different recipients have variable cost in terms of energy cost. The communication overhead can no longer be measured as number of messages and the energy cost of each message contributes differently to the communication overhead.

A. Paper Contributions

In our paper we make the following contributions:

We show that the key tree based schemes are energy-inefficient, since they do not take into account the network topology. These schemes were evaluated for energy-

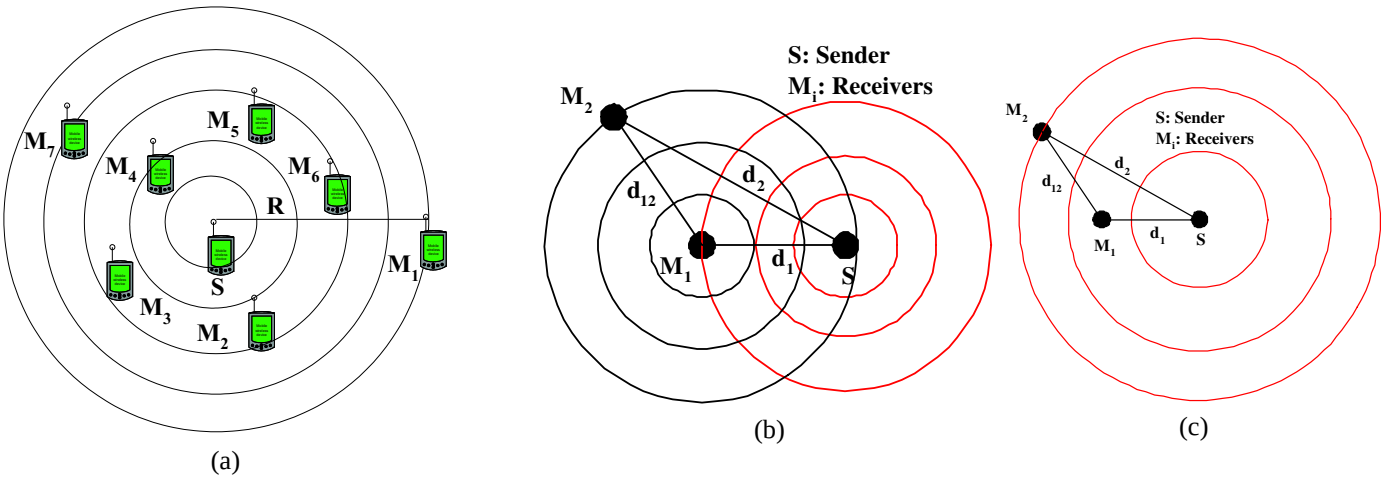


Fig. 1. (a) Broadcast advantage for members M_1-M_7. (b) Sender transmits an identical message to both receivers by transmitting to node M_1 and let M_1 relay the message to M_2. (c) Sender transmits an identical message to both receivers by transmitting to node M_2 and let M_1 receive the information for free.

constrained sensor networks in [2]. We formulate an optimization problem for performing energy efficient key distribution, when routing information is available. We map the problem into a non-bipartite weighted matching problem [22], whose optimal solution does not scale with the multicast group size. Hence, we indicate the necessity for heuristic algorithms.

We develop an energy-aware key distribution scheme that makes use of the geographical location information, when no other information is available. We also develop a simple heuristic routing-aware key distribution algorithm with scalable performance. Our algorithm combines physical and network layer information to achieve high energy efficiency. We argue that batch rekeying is suitable for combating high group dynamics and develop an energy-aware rekeying algorithm for performing batch rekeying based on the routing information.

The remainder of the paper is organized as follows. In Section II we describe the network model assumed and state the problem. In Section III we present the motivation behind our research topic. In section IV we present a routing-aware key distribution scheme. In Section V we develop a location-aware key distribution scheme. In section VI we develop an energy-aware batch rekey algorithm. In Section VII we provide simulation results and show the improvements achieved by our algorithms. In Section VIII we present related work. In Section IX we describe open problems and future work and in Section X we conlcude.

II. Problem Statement and Network Model

A. Network Model Assumptions

We assume that the network consists of N users randomly distributed in a specific area. We also assume single-sender multiple-receiver communication model. All nodes can act as relay nodes and therefore relay information to any node within their communication range defined by their transmission power. The nodes of the network are assumed to be static.

We assume that signal transmission is the major component of energy expenditure and therefore ignore any processing cost involved. We further assume that omnidirectional antennas are used for transmission and reception of the signal. The required power for reaching a receiver at a distance d is proportional to the γ^{th} power of that distance, $P_d \propto d^\gamma$ where $2 \leq \gamma \leq 4$. Without loss of generality we set the proportonality constant to be equal to 1. Hence, $P_d = d^\gamma$. This is a rather simple model that is used in cases of flat terrain with line of sight (LOS) transmission between the transmitter and the receiver. The algorithms that are developed in this paper are directly applicable when different propagation or antenna models are employed.

The omnidirectionality of the antennas used for signal transmission and reception results in a property unique in the wireless environment known as the *broadcast advantage* [16]. In Figure 1(a) the sender S is sending a message to member M_1. All members that lie within the circle of radius SM_1 will receive the broadcasted message for free. Hence, when an identical message needs to be sent to multiple receivers, the sender can significally reduce the energy expenditure by making one transmission to the furthest member.

B. Ad Hoc Environment Characteristics

In Figure 1(b), assume that $d_2 > d_1$ and that the sender S needs to transmit an identical message to nodes M_1 and M_2. One strategy would involve two unicast transmissions to each recipient with individual energies d_1^γ and d_2^γ and a total energy expenditure of $d_1^\gamma + d_2^\gamma$. However, the sender S can reduce the overall energy expenditure by taking advantage of the broadcast nature of the wireless medium. The sender can choose between one of the two following strategies: (a) transmit to M_1 and let M_1 relay the message to M_2. (b) transmit to M_2 and let M_1 receive the message for free, since $d_2 > d_1$ (due to broadcast). If $d_2^\gamma > d_1^\gamma + d_1 2^\gamma$ then the sender chooses the strategy (a), depicted in Figure 1(b), otherwise strategy (b), depicted in Figure 1(c), is preferred.

Although most of the existing routing algorithms for wireless ad hoc networks attempt to be energy efficient, we do not tie our work to any specific one. The network layer is responsible for constructing the routing paths, using criteria of energy efficiency, power efficiency or overall network lifetime. Our proposed work minimizes the energy expenditure due to the

overhead added to perform secure group communications and does not alter routing.

C. Problem Statement

Our goal is to perform key management for achieving secure group communications in a wireless ad hoc network environment. We assume that the network has been succesfully initialized, and initial cryptographic quantities (pairwise trust establishment) have been distributed. Several approaches about secure initialization in ad hoc networks with energy limitations are presented in [7], [8], [9]. Under these assumptions we develop key distribution schemes that satisfy the following requirements:

- Distribute and update the necessare cryptographic quantities for secure multicast in an energy efficient manner.
- Are scalable with group size and limit the required memory allocation.
- Limit the number of messages required to be exchanged for ensuring protection againsth unauthorized access when changes in group membership occur.

D. Efficiency Metrics

In wired networks, the communication cost of the key distribution is defined as number of messages required to be exchanged when a membership change occurs. The storage cost is defined as the number of keys each member needs to store. Depending on the key distribution scheme, there is a tradeoff between storage cost and communication cost.

For example, if key trees are employed as a key distribution method, a higher degree tree will result in lower storage, but higher communication overhead. For logical key hierarchies as proposed in [3] the communication cost is equal to $d \log_d N$ where d is the degree of the tree. The optimal degree for lowest communication cost in wired networks is $d = 3$ [3], [5]. For key trees that use one way functions the communication cost is equal to $(d-1)\log_d N$ and the optimal degree is $d = 2$ [4], [5].

In the wireless ad hoc environment, every message requires different energy expenditure depending on the location of the receipient. Hence, a lower number of transmitted messages does not guarantee lower energy expenditure. As it can be shown (see Appendix B), it is possible for a higher degree tree to have a lower energy cost, even if more messages need to be transmitted.

Although we are intrested to keep the number of messages exchanged low, our major design goal is to minimize the communication overhead in terms of energy expenditure. Therefore, our communication efficiency metric will be the *energy* required for the distribution of the cryptographic quantities, rather than the number of messages. The storage metric will still be number of messages need to be stored.

III. Motivation

Group key management ensures that only valid members hold proper keys at any time instance. Realization of group key management can be done through secure distribution and update of the cryptographic entities. The update of the cryptographic keys, also known as rekeying operation, is necessary due to group membership changes and/or key lifetime expiration.

When the group communication involves large dynamic groups where frequent membership changes occur, the key management/distribution scheme needs to be scalable with the group size. The most efficient group key management techniques that have been proposed for group communication in wired networks, are relying upon logical key trees [1], [3], [4], [5], [6]. The use of a hierarchical scheme such as a tree, mainly focuses on the scalability problem with respect to the communication cost and the storage requirements in both the sender and each individual member. Logical key tree based schemes reduce the complexity of rekeying operation from $O(N)$ *to* $O(logN)$. Additionally, new approaches attempt to bound the communication cost and processing overhead, and hence improve scalability and performance, by performing periodic or batched rekeying instead of rekeying on every membership change [17], [18], [19].

In the following sections we present the impact of the physical location, and physical, network layer information on the energy efficiency of the key distribution.

A. Impact of the Physical Location on the Efficiency of the Key Distribution.

We now demonstrate the need for consideration of the geographical location information in the construction of the key distribution scheme. Such information can be obtained using the Global Positioning System (GPS) [14]. In Figure 2(a), we represent a wireless ad hoc network of 7 nodes (network A), with S being the sender, and two intermediate nodes R_1, R_2 relaying traffic to four members of a multicast group M_1-M_4. The energy required for sending a message from the GC to the two relay nodes is set to one unit and the energy required for sending a message from the relay nodes to the receiving nodes is also set to one unit. Hence, the GC needs to perform only one transmission to reach R_1, R_2 and relay nodes R_1, R_2 need to perform one transmission each to reach $\{M_1, M_2\}$ and $\{M_3, M_4\}$, respectively.

Figure 2(b),(c) present two different key distribution strategies for the multicast group in Figure 2(a). The one in Figure 2(b) is built according to the available geographical location information (close-by members are placed adjacently into the key tree), while the one in Figure 2(c) is a result of a random placement of the members into the leaves of the tree. For key trees, a member is assigned keys that are along the path traced from the leaf node to the root [1]. For example M_1 is assigned keys $\{K_0, K_{1.1}, K_{2.1}\}$.

Let's assume that key K_0 has been compromised and needs to be replaced by the new key K_0'. For scheme in Figure 2(c), the GC generates encrypted messages and transmits them to relay nodes R_1 and R_2, respectively. Node R_1 performs one transmission to M_1, M_2 and R_2 performs one transmission to M_3, M_4. The total energy expenditure is 4 energy units. For scheme in Figure 2(c), the GC transmits two messages to both R_1, R_2. Both R_1 and R_2 need to transmit twice to reach nodes M_1, M_3 and M_2, M_4, since nodes that do not share common

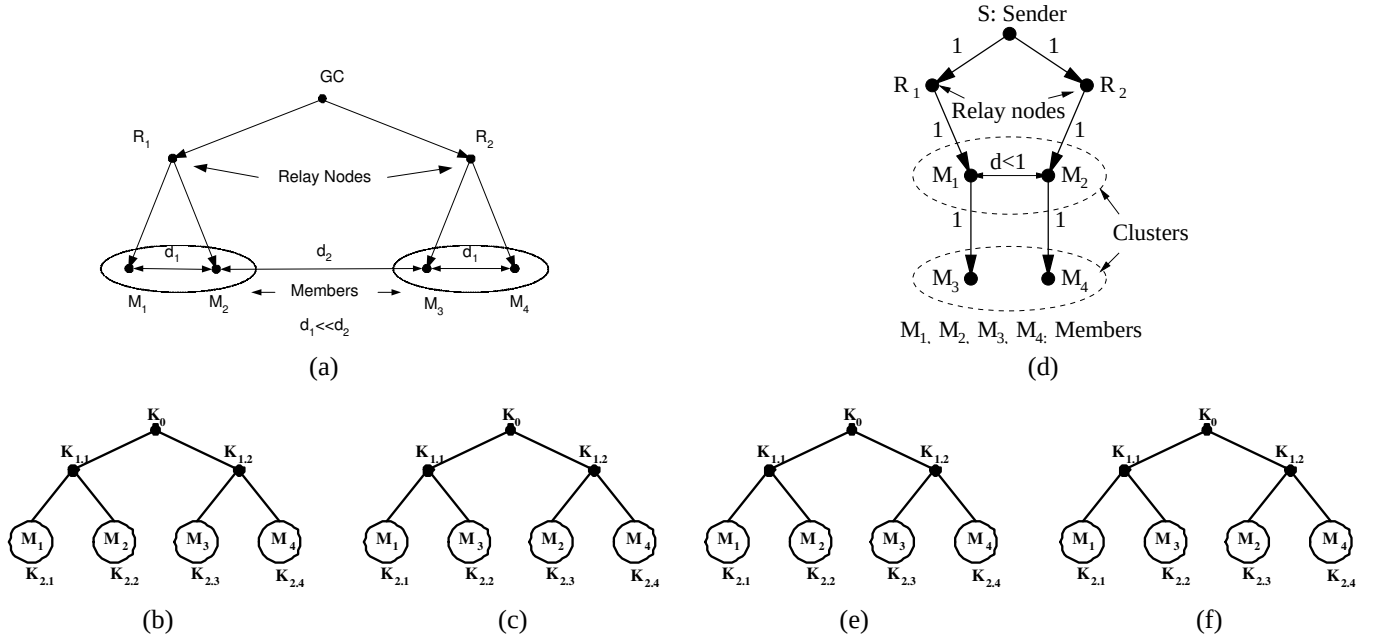


Fig. 2. (a) Routing tree for network A. (b) A location-aware key distribution scheme for network A. (c) A random logical hierarchical key distribution scheme for network A. (d) Routing tree for network B. (e) A routing-aware key distribution scheme for network B. (f) A location-aware key distribution scheme for network B.

keys cannot receive information with a single transmission. The scheme in Figure 2(c) requires 6 energy units. *Hence, for this example, the consideration of the physical location information in the realization of the key distribution scheme leads to energy savings of 33%.*

B. Impact of the physical and network layer on the efficiency of the key distribution.

When only the geographical location information is available, proper consideration in the key distribution tree construction can lead to significant energy savings. However, if physical and routing layer information is also available, a routing-aware key distribution shceme can be even more energy efficient. We demonstrate this with the following example.

In Figure 2(d), we represent a wireless network of 7 nodes, with S being the GC, and two intermediate nodes R_1, R_2 relaying traffic to four members of the multicast group M_1-M_4. The energy required for sending a message from the GC to the two relay nodes is set to one unit and the energy required for sending a message from the relay nodes to M_1, M_2 is also one unit. The energy required for sending a message from M_1, M_2 to M_3, M_4 respectively is one unit.

Figure 2(e),(f) present two different key distribution strategies for the multicast group in Figure 2(d). The one in Figure 2(e) is location-aware, while the one in Figure 2(f) is routing-aware. Let's assume that K_0 has been compromised and needs to be replaced by the new key K_0'. For scheme in Figure 2(e), the GC generates encrypts K_0' with $K_{1.1}$ and transmits it to relay nodes R_1 and R_2 respectively. Nodes R_1, R_2 are relaying the message to M_1, M_2 respectively. The GC also encrypts K_0' with $K_{1.2}$ and transmits it to relay nodes R_1, R_2 respectively. Nodes R_1, R_2 relay the message to M_1, M_2 and the latter forward the message to M_3, M_4. The total energy expenditure is 8 energy units.

For scheme in Figure 2(f), by following similar steps the energy expenditure for updating K_0 is only six units.

IV. Routing-Aware Key Distribution Scheme

We showed that the joint consideration of the physical and network layer in the design of the key distribution scheme for secure multicast in ad hoc networks, can lead to significant energy savings. In this section we formulate an optimization problem for constructing an energy-efficient logical tree. We show that the complexity of the optimal solution is prohibitive for large networks. We alternatively propose a hybrid method with significantly lower complexity for the construction of an energy-efficient routing-aware key tree. Without loss of generality (more for clarity), we try to construct a balanced binary tree. Extension to a d-ary tree is straightforward.

A. Problem Formulation

We assume that the probability of each user leaving the multicast group is uniform i.e. members have equal probability of leaving the muticast group. We also assume that the underlying routing is reliable and it can be used to compute the required energy to reach a group of members. We must note that in no way we attempt to alter the routing procedure. Our proposed scheme is independent from the routing process. However, we use the outcome of routing to achieve energy-efficient key distribution.

We define the following quantities:

- N: Multicast group size
- T: Key distribution tree
- l: Level of the key distribution tree T
- R: Underlying routing tree

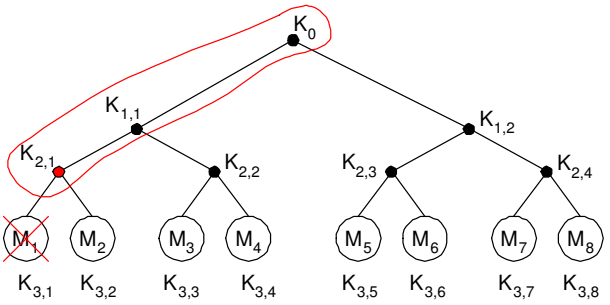


Fig. 3. M_1 leaves the multicast group

- S: The set of all members of the multicast group
- S_k^i: The i^{th} element of a partition of S into subsets consisting of k users. i.e. $\cup S_k^i = S$ and $S_k^i \cap S_k^j = \emptyset$ for $i \neq j$.
- $E_{S_k^i}(R)$: Energy required to broadcast a message to members of subset S_k^i according to the routing tree R.
- $E_{M_i}(R,T)$: Energy required for rekeying when the i^{th} member leaves the multicast group.
- $E_{TOTAL}(R,T)$: Total energy required for rekeying when every user leaves once the multicast group, according to the key distribution tree T and the routing tree R.

In Figure 3 we present a key distribution tree for a multicast group of $N = 8$ members. The energy expenditure for rekeying after a member leave varies with the location of the members that need to receive the updated keys. Therefore we calculate the total energy required for rekeying after each member leaves once the multicast group:

$$E_{TOTAL}(R,T) = \sum_{i=1}^{N} E_{M_i}(R,T) \tag{1}$$

We can compute the average energy required for rekeying after a member leave as $E_{TOTAL}(R,T)/N$, since all members have the same probability of leaving the group.

Suppose that member M_1 in Figure 3 leaves the multicast group. Keys K_0, $K_{1,1}$ and $K_{1,2}$ have to be updated to the appropriate user groups. Assuming that each member receives a message with all the new keys that it needs the energy required for updating the keys after M_1 left the multicast group is:

$$\begin{aligned} E_{M_1}(R,T) &= E_{\{M_2\}}(R) + E_{\{M_3,M_4\}}(R) \\ &\quad + E_{\{M_5,M_6,M_7,M_8\}}(R) \end{aligned} \tag{2}$$

Summing over all members M_i will give us the total energy $E_{TOTAL}(R,T)$:

$$E_{TOTAL}(R,T) = \sum_{i=1}^{8} E_{S_1^i} + 2\sum_{i=1}^{4} E_{S_2^i} + 4\sum_{i=1}^{2} E_{S_4^i} \tag{3}$$

Generalizing equation (3) for N users where for simplicity of the equations we have assumed that $N = 2^n, n \in Z^+$

$$E_{TOTAL}(R,T) = \sum_{j=0}^{\log_2 N - 1} \left((j+1) \sum_{i=1}^{\frac{N}{2^j}} E_{S_{2^j}^i} \right) \tag{4}$$

The total energy expenditure depends upon the routing tree R and the key distribution tree T, i.e. the members' position on the leaves of the key distribution tree. As stated earlier, our goal is not to alter the routing tree for minimizing the energy expenditure of the communication overhead that key management involves. We are interested in selecting the optimal tree T^* that minimizes the quantity in equation (5) given that routing is fixed and provided by the network layer.

$$T^* = \arg\min_T E_{TOTAL}(R,T) \tag{5}$$

B. Complexity of the problem

Solving the above optimization problem, involves minimizing the energy expenditure for transmiting keys located at different levels of the key distribution tree T. If the keys are located in the l^{th} level of the tree, the will be sent to groups of $N/2^l$ members. For level $l = 0$ (root key) the energy expenditure is constant and equal to broadcasting to the whole group. For level $l = \log_2(N)$ the energy expenditure is constant and equal to broadcasting to each individual member.

For level $l = \log_2(N) - 1$ we need to broadcast keys to groups of two members. We can obtain the optimal solution by borrowing an algorithm from graph theory. We can map the optimal partition into subsets of 2 members problem, into a minimum weight non-bipartite matching problem. Consider an undirected graph $G(V,E)$. A matching is a collection of edges $M \subseteq E$ such that no two edges in M are incident. Let $w : E \rightarrow \Re^+$ be a function which assigns a weight to each of the edges e_i of G. The weight $w(F)$ of a subset $F \subseteq E$ of the edges of G is defined as $w(F) := \sum_{e \in F} w(e)$. The minimim weight non-bipartite matching problem is to find a perfect matching M in G such that $w(M)$ is minimized.

Let the set of vertices V correspond to the members of the multicast group and let the weights $w(e)$ of the edges correspond to the energy expenditure assosiated for broadcasting a message to both members connected by the edge $e \in E$ according to the routing tree R. Finding the partition of M into subsets of two members that requires the least amount of energy for broadcasting a message to each subset, is equivalent to solving the minimum weight non-bipartite matching problem for the graph $G(V,E)$.

Though several algorithms have been developed for finding the optimal solution [20], [21], [22], [23], [24] the least complex technique requires the use of sophisticated data structures and has complexity of $O(n(m + nlogn))$ where $|V| = n$ and $|E| = m$, [23]. The complexity of the algorithm is prohibitive for large networks, since it does not scale with the group size N. Additionally, obtaining the optimal solution for level $l = log_2(N) - 1$ and combining its outcome for level $l = \log_2(N) - 2$, does not guarantee optimality for that level as it can be shown by example. Therefore, heuristic solutions have to be employed.

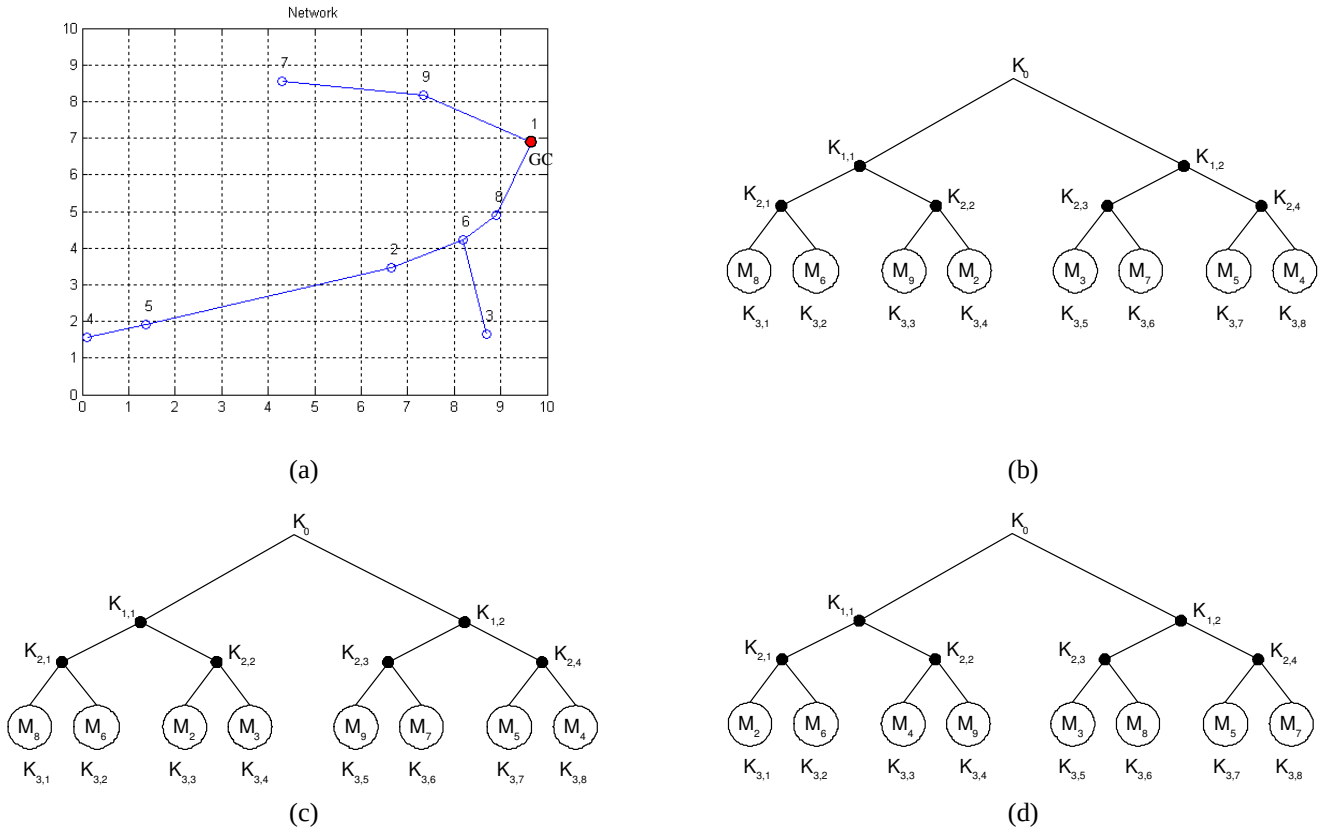


Fig. 4. (a) The routing paths of a wireless ad hoc network. (b) Key distribution tree built with the Routing-Aware key distribution algorithm. (c) Best possible Key distribution tree. (d) Worst possible key distribution tree

C. A Simple Heuristic Solution Based on Routing

Our heuristic solution relies on the routing tree R for constructing the key distribution tree T. We make the obseravation that the inner nodes (nodes closer to the source) are reached before the outer nodes (nodes away from the source). Inner nodes are either leaf nodes in which case the broadcast advantage is utilized and they receive a message for "free" when this message has to be sent to the outer nodes, or they act as relay nodes in which case they still receive a message for "free" when they have to relay it to outer nodes. Additionally, we note that the broadcast advantage is most beneficial when nodes are almost in the same radius from the source. In that case, no matter if the inner node is a leaf or a relay node, we save almost as much energy as we spend for sending a message to both the inner and outer node. Hence, grouping together nodes that are close to each other in a circular way will lead to an energy efficient key distribution tree.

Routing-Aware Key Distribution Scheme

Step 1: Compute $E_i(R) = E_{S_k^i}(R)$

Step 2: Sort $E = \{E_1, E_2, ..., E_N\}$ in ascending order.

Step 3: Add leaf nodes from left to right as in E.

D. Application of the Routing-Aware Algorithm to a Sample Network

We now illustrate the construct of the key tree for the nine-node network shown in Figure 4(a). The GC can communicate with each member of the multicast group by using the routing paths indicated. Sorting the energies for reaching each member of the multicast group gives $E_{\{M_8\}} < E_{\{M_6\}} < E_{\{M_9\}} < E_{\{M_2\}} < E_{\{M_3\}} < E_{\{M_7\}} < E_{\{M_5\}} < E_{\{M_4\}}\}$. The resulting key distribution tree is shown in Figure 1(b). The optimal key distribution tree, obtained by brute force searching, is shown in Figure 4(c). We can observe that the two trees are almost identical with only members M_4 and M_4 been intercahnged. The worst possible tree, also obtained through brute force is shown in Figure 4(d). We observe that this tree is drastically different than the optimal.

V. Location-Aware Key Distribution Scheme

We make the observation that members located close to each other can potentially be reached with broadcast, or use the same routing paths to receive data. If we represent members as points in the 2-dimentional plane, we can employ a clustering algorithm to cluster them into groups and construct a hierarchical key tree structure.

A. A Heuristic Location-Aware Algorithm.

Though one can use any suitable clustering algorithm, we have developed a variant of *K-means* (see Appendix B) algo-

rithm for creating appropriate clusters. K-means is used due to its ease of implementation and the ability to control the number of steps in which it terminates [15]. The goal of K-means is to create K clusters out of N points ($K < N$) such that a "loss" cost function is minimized with respect to a dissimilarity measure. K-means uses the squared Euclidean distance as a dissimilarity measure. If the coordinates of point i are $x_i = (x_{i_1}, x_{i_2})$, the Euclidean distance is equal to:

$$d\left(x_i, x_{i'}\right) = \sum_{j=1}^{2} \left(x_{ij} - x_{i'j}\right)^2 = \|x_i - x_{i'}\|^2 \qquad (6)$$

When the propagation loss factor is $\gamma = 2$, the Euclidean distance is proportional to the transmission power and is suitable for our clustering procedure. In case where $\gamma \neq 2$, we can modify the dissimilarity measure to be proportional to the transmission power as in (7).

$$d\left(x_i, x_{i'}\right) = \|x_i - x_{i'}\|^{\gamma} \qquad (7)$$

In our analysis we will focus, without loss of generality, on the case where $\gamma = 2$. When $\gamma \neq 2$ instead of K-means, K-medoids can be used to solve the optimization problem at the expense of increased computational complexity [15].

Using K-means, N members are assigned into 2 clusters. Since K-means does not guarantee that equal number of members will be assigned to each cluster or N might be odd, we adopt a refinement procedure for assigning equal number of members to each cluster. This refinement leads to a construction of a balanced key tree when $N = 2^n$ and forces a structure as close to the balanced as possible otherwise. At every following step all clusters are further divided into two new clusters and the refinement procedure for each pair of clusters is repeated. The algorithm terminates when we have created clusters of two members (after $\log_2 N$ splits when $N = 2^n$).

B. Description of the Location-Aware Key Distribution Algorithm

Step 1: Assign all points to an initial global cluster.

Step 2: Divide each cluster into two clusters using the K-means algorithm.

Step 3: Use the refinement procedure detailed below to balance the number of points that are assigned to each cluster, i.e. assign the same number of points to each cluster.

Step 4: Iterate step 2 and 3 until clusters of two or one points have been created.

Step 5: Merge single points, if possible, with the use of K-means for only single points.

Step 6: Map the cluster hierarchy into tree hierarchy.

If cluster C_1 has $|C_1|$ points and cluster C_2 has $|C_2|$ points with $|C_1| > |C_2|$, the refinement procedure will be moving $\lfloor (|C_1| - |C_2|)/2 \rfloor$ points from C_1 to C_2. The criterion by which the points are selected, is the minimum Euclidean distance of members belonging to cluster C_1 from the point expressing the mean vector m_{C_2} of cluster C_2.

Refinement Procedure

for $(k := 1;\ k \leq \lfloor (|C_1| - |C_2|)/2 \rfloor ;\ k++)$
find i* such that

$$d(x_{i^*}, m_{C_2}) = \min_{i \in C_1} \|x_i - m_{C_2}\|^2 \qquad (8)$$

and move it to cluster C_2. recalculate means m_{C_1}, m_{C_2}.
endfor

C. Illustration of the Location-Aware Algorithm

In Figure 5(b), the results of the application of the location-aware algorithm for the nine-node netowrk in Figure 5(a). Initially, clusters $C_1 = (M_1, M_2, M_3,\ M_4, M_5, M_6)$ and $C_2 = \{M_7, M_8\}$ are constructed. After the application of the refinement procedure, members M_5 and M_6 are moved from C_2 to C_1 in order to form two clusters of equal size. A second application of K-means results in the shown final four clusters and the tree in Figure 5(b) is built.

We must note that we prefer a top-down approach and split the group into two subclusters at each step, rather than a bottom-up approach that would involve creating $N/2$ clusters in the first step. The bottom-up approach is much more complex than the top-down, since k-means creates the desired number of clusters with unequal number of points within each cluster. Hence, the refinement procedure would have to be extremely complex in order to be able to energy efficiently move points between a great number of clusters for the creation of a balanced structure.

VI. Energy-Aware Rekeying

A. Batch Rekeying for High Group Dynamics.

There are several known rekeying schemes for wired netowrks that are scalable with group size N. The rekeying strategies in [3] are based on the partial update of all the necessary KEKs in every change of membership. This is known as individual rekeying. All KEKs are updated only when they expire.

Individual rekeying after each join and leave might prove expensive in communication overhead and hence energy expenditure, especially when the group dynamics are high, i.e., joins and leaves happen with a high frequency. Additionally, since the keys have to propagate through the network, the propagation delay might be comparable to the time between joins/leaves. In this case, there are keys that are generated and sent by the GC to the appropriate members, that will never be used since by the time they reach the members they have to be updated due to a new member leave/join.

In [18], batch rekeying is proposed as a scalable alternative to individual rekeying. In batch rekeying, the GC waits for a

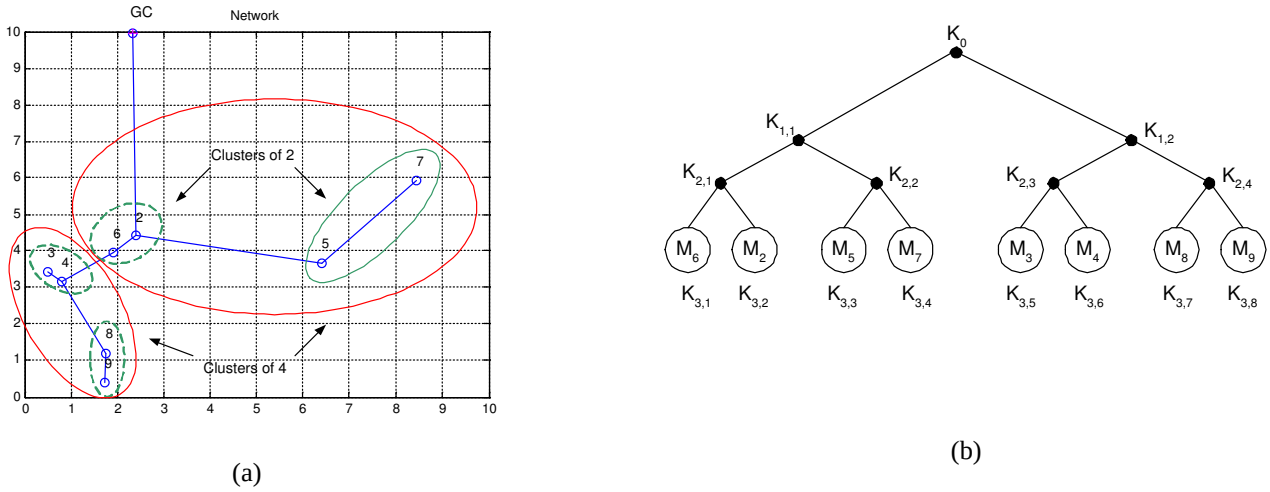


Fig. 5. (a) Application of the k-means clustering algorithm in an ad hoc network. (b) Resulting location-aware key distribution tree.

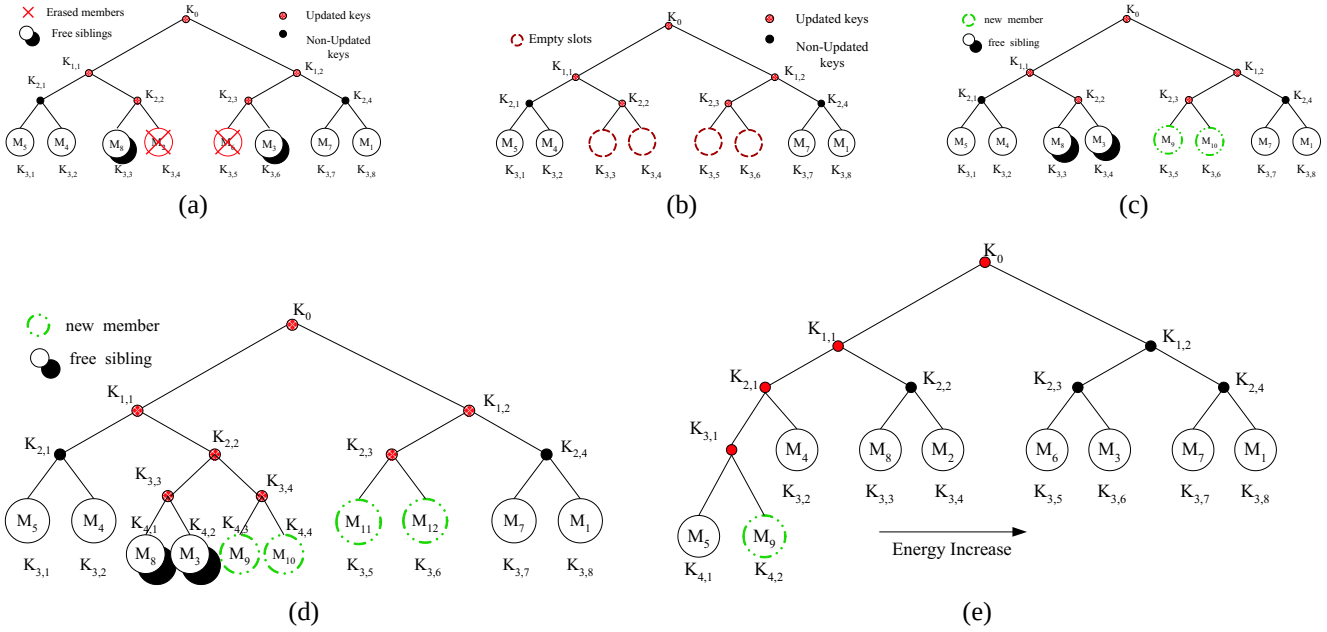


Fig. 6. (a) The routing paths of a wireless ad hoc network. (b) Key distribution tree built with the Routing-Aware key distribution algorithm. (c) Best possible Key distribution tree. (d) Worst possible key distribution tree.

certain period of time called a rekey interval, collects all the join/leave requests and updates all keys after the expiration of the rekey interval. A performance evaluation of both methods is provided in [17].

In [18], [19] it has been shown that batch rekeying decouples the frequency of key updates from the dynamics of the multicast group. It has a predictable communication cost and avoids problems of out-of-sync data delivery. Its main drawback is the increased latencies that members might experience in joining and leaving the communication group. Additionally, there is a tradeoff between the desired level of security and the rekey interval. The rekey interval is also known as vulnerability window since during this time, evicted members are still able to receive data, until the next key update takes place.

The reason why batch rekeying is preferable to other strategies is the increased frequency of rekeying, due to the nature of the ad hoc wireless networks. Apart from the regular need for rekey due to a membership change, there are several factors that increase the rekey frequency. New power-aware routing algorithms [25] [26] propose frequent update of the routing tree for extending the network lifetime and avoid rapid battery depletion. In such a case, routing information is outdated after a period of time even if the members are completely static. Additionally, node failures can alter either the membership status or the routing paths. In any case, we can interpret node failures as leaves and treat them ac-cordingly in the rekey operation.

The batch rekey algorithms that have been developed for the wired networks consider only the scalability issue and try to reduce the number of required encryption keys and/or maintain a balanced key tree structure. Our approach has an additional objective due to the different energy cost of a

message with different receipients. The rekey algorithm will primarily attempt to *update keys that require less energy to be delivered*. The objectives of 1) reducing the number of keys that need to be updated and 2) keeping a balanced tree structure,are also going to be preserved. The addition of the energy dimensionality does not allow a logical update distribution algorithm based on integer IDs as proposed in [17]. The new members have to be added into the tree structure according to their location and depending on the modified routing structure rather than a logically assigned integer ID. The proposed algorithm can be illustrated by the following examples.

Let:

- J: Set of joins collected in one rekey interval.
- L: Set of leaves collected in one rekey interval.
- B: Set of members with a free sibling.
- A: Set of members that will be added to the key tree (A is not necessarily equal to J).
- P: Set of empty slots in the key tree.

B. Illustration of the Energy-Aware Rekeying Algorithm

Example 1: Assume that the application of the routing-aware key distribution algorithm for a multiast group of 8 members results in the key distribution tree of Figure 6(a). Assume also that during the rekey interval, 2 leaves and 2 joins occur. Members M_2 and M_6 are leaving the multicast group, while members M_9 and M_{10} are joining the group. In Figure 6(b) we indicate the members that leave the group and the leaf nodes that are left without a sibling. For our case $J = \{M_9, M_{10}\}$, $L = \{M_2, M_6\}$ and $B = \{M_8, M_3\}$. Since all the keys that members of set B hold will be updated, the update procedure for members of set B is equivalent to the joining procedure for a new member except for the pairwise key establishment. Hence, we can consider that those members belong to set A which becomes $A = \{M_9,\ M_{10},\ M_8,\ M_3\}$. There are four empty slots created that can accomodate the members of set A.

The placement of members of set A into the key tree has to be routing-aware so as to be energy efficient. We follow the same principle as in Section V and order the set A according to ascending energy required to reach every member from the GC. We then place the members into the empty slots from left to right. In our example A was ordered as $A = \{M_8,\ M_3,\ M_9,\ M_{10}\}$ and the resulting key tree structure is shown in Figure 6(c).

In the case where more joins than leaves happen, leaf nodes have to be split and the height of our tree has to increase. Since we attempt to keep a balanced tree structure, we shall split all leaf nodes before further increasing the height of the tree. We choose to split the empty slots first and with a priority from left to right. The reason why empty slots have to be split first is that all keys from the empty slots to the root of the tree are going to be updated due to joins. So if we initially fill the empty slots and than split a leaf node that was not empty before, we cause additional key updates. Also, the nodes shall be split from left to right since our tree construction is such that higher energy is required to propagate keys to the right side of the tree. By splitting the leftmost empty slot we assign a greater number of keys to members that require the least energy to be reached by the GC.

Example 2: Consider again the key distribution tree of Figure 6(a) and assume that 2 leaves and 4 joins occur. Members M_2 and M_6 are leaving the multicast group, while members M_9, M_{10}, M_{11}, M_{12} are joining the group. In this example $J = \{M_9, M_{10}, M_{11}, M_{12}\}$, $L = \{M_2, M_6\}$, $B = \{M_8, M_3\}$ and $A = \{M_9,\ M_{10},\ M_{11},\ M_{12},\ M_8,\ M_3\}$. By ordering Awe get $A = \{M_8,\ M_3,\ M_9,\ M_{10},\ M_{11},\ M_{12}\}$ and choose to split the two leftmost empty slots. The resulting key distribution tree is shown in Figure 6(d).

Example 3: In Figure 6(e) we show the key distribution tree when only one join occurs during the rekey interval. Again we choose to split the leftmost member for the addition of the new member, since fewer keys will be sent to member that are further away from the GC.

A very important observation we shall make is that the key tree after an update due to membership changes does no longer correspond to the new underlying routing structure to the fullest. The deletion and addition of new nodes, changes the routing tree and consequently the energy required to reach each member from the GC. Therefore, the ordering of the leafs of the key tree is not in accordance with the ordering of the set containing the energies for reaching each member. However, if we were to reconstruct the tree according to the new routing tree and the resulting energies ordering, we would have to update all the keys of the key tree rather than a portion of them and hence spent more energy. The set E is partially ordered depending on what portion of keys was updated. The more keys are updated the closer the key tree will be to the optimal tree obtained if recosntruction according to the new routing tree took place.

C. Energy-Aware Rekeying Algorithm

Step 1: Collect joins and leaves during the rekey interval. J set of joins, L set of leaves

Step 2: Find the set of free siblings B. Set $A = \{J, B\}$.

Step 3: Order the members in set A in ascending energy required to be reached by the GC.

Step 4: If $|J| \leq |L|$ add the members in A into the empty slots from left to right. Add dummy nodes to unused slots. Set as dummy a parent that has dummy children.

Step 5: If $|J| > |L|$ calculate the nodes that need to be split. Split empty slots from left to right. If more nodes need to be split, split regular nodes from left to right.

VII. Simulation Results

A. Simulation Setup.

Simulation was performed in randomly generated network topologies confined in a 10x10 square grid region. After the

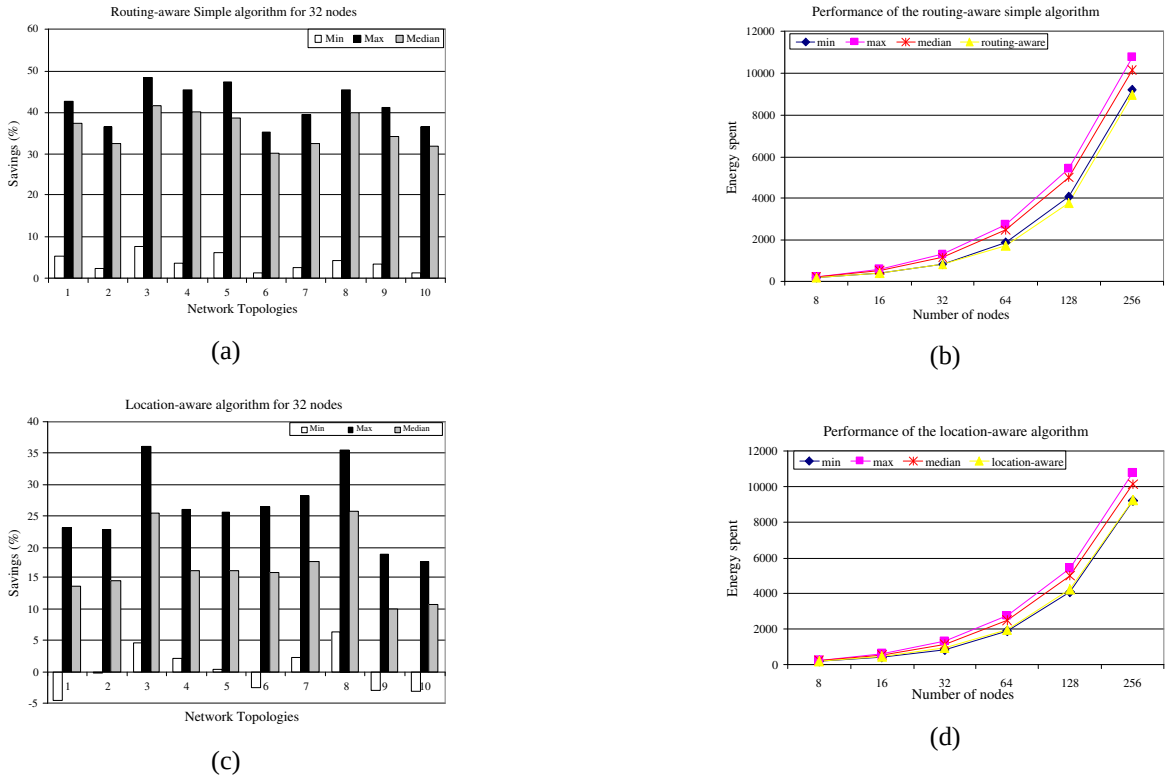


Fig. 7. (a) Performance of the routing-aware simple algorithm for 10 different network topologies of size 32. (b) Performance of the routing-aware simple algorithm for different network sizes, averaged over 100 different network topologies. (c) Performance of the location-aware algorithm for 10 different network topologies of size 32. (d) Performance of the location-aware algorithm for different network sizes, averaged over 100 different network topologies.

network generation, the routing paths are established according to the routing algorithm. Though any suitable algorithm can be applied to provide the routing paths, we used BIP [16]. The resulting routing tree is used to calculate the consumed energy for transmitting an updated key to each member or group of members. In our network model, members have the ability to transmit with infinite power and the propagation loss factor is set to $\gamma = 2$. Our network is static. Since there is no algorithm to provide the optimal solution for the key distribution tree construction, we performed exaustive search for small group sizes $N = 8, N = 16$. For larger group sizes, $N = 32, 64, 128$, we generated for each network instance, 10,000 different key tree structures and compared the performance of our algorithms with the key tree that requires the minimum, maximum and median energy for key update, out of the 10,000 tree structures. Further, we repeated the same comparison for 100 different network topologies and averaged the result. Although the best solution might not be included in the 10,000 generated trees, this still remains a valid comparison since it emulates the random logical assignment in the wired netwroks.

B. Experiment 1: Application of the Simple Routing-aware Algorithm.

In Figure 7(a), we can observe that the simple routing-aware algorithm gives significant savings compared to a tree structure that does not take into account the network topology. It can save up to 49% compared to the worst possible assignment, while it does almost as good as the best possible assignment out of $10,000$ trees. In Figure 7(b), the average savings for different network sizes are presented. The savings are averaged over 100 different network topologies for each value of N and $10,000$ trees are used for comparison for each topology. For network sizes of 8 and 16 nodes, where we could perform exaustive search our solution was no more than 4% off the optimal solution, while it did up to 43% better than the worst solution.

C. Experiment 2: Application of the Location-aware Algorithm.

In Figure 7(c) the savings for the location-aware key distribution tree are shown. In Figure 7(d) the average savings over 100 different network instances are shown. We can observe that the K-means variant algorithm, significantly out-performs most of the key structures. It provides 15% - 37% savings from the worst possible assignment, 15% - 26% savings from the median case and does almost as good as the best possible case out of $10,000$ trees.

In Figure 8(a) and Figure 8(b) the simple routing-aware algorithm is compared to the location-aware one. The simple routing-aware algorithm performs better by 2% - 9%. The difference relies on the fact that the location-aware algorithm fails to accurately exploit the circularity of the broadcast advantage. It only considers adjacent nodes, while in reality receiving nodes can be located in oposite directions relative to the transmitting node and still benefit from broadcast.

The performance of our location-aware key distribution algorithm relies also on the homogeneity of the environment

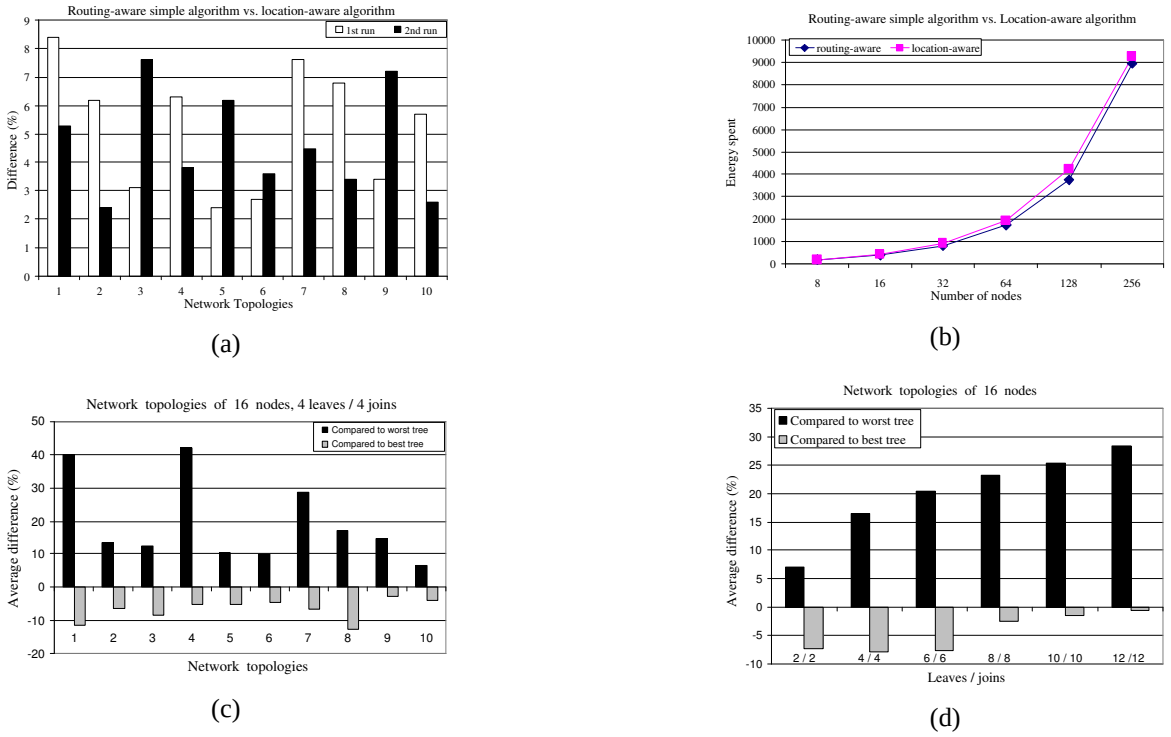


Fig. 8. (a) Comparison of the simple routing-aware algorithm vs. the location-aware algorithm for 10 different network topologies of size 32. (b) Comparison of the routing-aware simple algorithm vs. the location-aware algorithm for different network sizes, averaged over 100 different network topologies. (c) Performance of K-means variant for 10 different network topologies. (d) Average savings for different N. (e) Comparison of the greedy algorithm with K-means algorithm for 10 different network topologies. (f) Comparison of the greedy algorithm with K-means algorithm for different number of nodes.

where the ad hoc network is deployed. If the path loss model is identical throughout the network terrain, then the physical location of the nodes is directly connected with the energy spent for reaching them. However, when different path loss models must be assumed for the description of different regions, high spatial correlation between nodes does not necessarily imply the use of similar routing paths for directing traffic to those nodes.

Hence, we can partition the area where the network is deployed, by identifying regions that are described by the same path loss model. This partition is followed by the application of the K-means variant to each region, resulting in the construction of sub-clusters.

D. Experiment 3: Application of the Energy-aware Rekeying Algorithm.

In the simulation for the energy-aware rekeying algorithm, we created random network topologies and forced random joins and leaves. We measured the required energy for updating the appropriate keys when the member insertion was done according to the proposed algorithm and when the member insertion was performed in the worst possible way, a possible result of an assignment based on logical IDs.

We also compared the difference between the key tree which was updated with the use of the energy-aware batch rekeying algorithm, and the best possible key tree which is generated with the use of the routing-aware simple algorithm, given the new routing conditions. This difference is an indicator of how much does our tree deviates from the routing. The deviation depends on the number of joins/leaves in the network and the actual change in the routing paths. When the routing paths change drastically due to member deletion/addition and the number of joins/leaves is small compared to the total number of members of the multicast group, then the deviation is high. The routing information that was initially used to construct the key tree is completely outdated and the best key tree structure has significant difference from the modified one. On the other hand, when the drastic change in the routing information is caused by a great number of joins and leaves, the best tree structure that the routing-aware simple algorithm creates, is very close to the tree structure that the energy-aware batch rekey algorithm generates and the deviation is very low.

In Figure 8(c), we show the performance of our energy-aware batch rekeying algorithm , for 10 different network topologies of 16 nodes. randomly generating 4 joins and 4 leaves, we place the new members according to our algorithm described in Section VIII and then place them in the worst way, a possible result of a logical as-signment. We calculate the energy savings by comparing the two cases. We also reconstruct the key tree from the new routing information available and compare it with the result of our algorithm. We can observe that significant savings occur given that we use our rekey algorithm than any logical version. We also observe that the difference between our structure and the best possible structure that can be generated by a complete tree rearrangement can have a difference of up to 20%.

In Figure 8(d), we show the performance of our energy-aware batch rekeying algorithm for different number of joins

and leaves. For every different combination of joins and leaves we generate 100 random network topologies and randomly choose the joins and leaves. We average the results over the 100 networks. We can observe that the more the number of joins and leaves the biggest the savings that we achieve with our algorithm. This result expected since more joins and leaves mean that more members will receive new keys and we can achieve more savings by performing optimal member assignment.

In contrast the difference between the routing-aware key tree and the energy-aware batch rekeying tree decreases with the increase of the number of joins and leaves. This outcome is also expected, since the more members are involved in the key tree rearrangement, the highest the correspondance of the tree to the new routing paths.

VIII. Related Work

Providing secure communication in an ad hoc network environment is a fairly unexplored scientific area. Very few key management schemes have recenlty been proposed for such networks.

In [8], a key management scheme for Distributed Sensor Netowkrs (DNS) of ad hoc nature is proposed. The scheme is based on key pre-distribution and probabilistic key sharing among the nodes of the network and provides services of rekeying, key revocation, as well as node addition and deletion. By using arguments from random graph theory, the authors propose a key pre-distribution of a certain small amount of keys, randomly selected from a large pool of keys, to each node of the network. A link exists between two nodes only if nodes share a key. Additionally, two nodes that are not within each other's communication range can establish a pairwise key if a path key exists between them. However, all communication between the two nodes is exposed to the intermediate nodes that determine the path key. The distinct difference between our proposed schemes and the shceme in [8] is that the network topology is taken into account in the design of the key management scheme so that energy efficient secure links are established in contrast to a probabilistic determnination of links to be used. Also, we are mostly intrested in secure group communications rather than point-to-point secure link establishment.

In [10], [11], [12], public key cryptography is proposed as a key management scheme for for wireless ad hoc networks. In contrast to most wired network applications where a trusted centralized authority is present, in ad hoc network such an assumption is not valid. In [10] threshold cryptography is proposed for the distribution of trust in the key management. N servers are able to sign certificates and $t+1$ of them are required to perform a cryptographic operation. In [11] the servers that realize the threshold scheme are called Mobile Certificate Authorities (MOCA). In [13] public key cryptography is proposed for multicast application in ad hoc networks. However the proposed scheme is neither scalable nor energy efficient.

The use of public keys and digital signatures require increased computational power and communication resources. In an ad hoc environment mobile hosts often have low computational power and operate in a limited battery. Hence, our work is based on symmetric, rather than strong asymetric cryptography. None of the approaches we have seen so far for performing key management in ad hoc networks, takes the energy expenditure of the communication overhead into account. Additionally, most past research is focused on securing unicast communication, while we are proposing algorithms for performing group communication.

Our work is based upon performing key management by using key graphs structures and especially trees, although it can be directly extendable to any form of graph. Key trees were initially proposed for group communication for wired networks in [1], [3], mainly addressing the scalability issues associated with multicast. In [17], [18], [19] periodic or batched rekeying instead of rekeying on every membership change is proposed in an attempt to bound the communication cost. All proposed key management schemes are logical and in contrast to our schemes do not consider the underlying network structure.

IX. Future work and open problems

Optimization over the graph strucuture: In our present work we showed that the network topology has to be taken into acound when designing a key management scheme for group communications in a wireless ad hoc environment. We adopted the tree structure as a scalable solution, and developed algorithms to perform key management in an energy efficient way. We presented the aplication of our algorithms to binary trees, although they are directly applicable to a tree of any degree. What structure is most energy efficient to use though, remains an open problem. For example, if trenary trees are used instead of binary trees, fewer keys have to be stored, but more messages have to be transmitted for a key update. Since each message has different energy cost, a trenary tree might prove more energy efficient than a binary one, depending on the network topology.

Incorporation of mobility for the nodes of the network: Mobility can be seen as a series of successive snapshots of the network. If nodes are mobile, the network topology is dynamic and the underlying routing tree is adapting to it in a dynamic way. Since our algorithms are tightly dependent, either directly (if the location-aware key distribution scheme is employed) or indirectly (if the routing-aware key distribution scheme is employed), upon the location of the nodes, if that location changes our key tree structure is outdated and does no longer reflect the network topology. Hence, we have to develop a method to adjust the key tree structure to the dynamically changing network structure.

The issues involved with the algorithm to be developed, concern how frequenctly the adjustment has to take place in order to be energy efficient, how sensitive is our key structure to routing changes, how can we battle effects of oscillation and deal with problems of stabilization.

Secure traffic delivery to subgroups: When a message needs to be broadcasted securely to the entire group, the energy efficiency of the broadcasting is solely dependent upon the underlying routing tree. Hence, otpimizing the routing procedure will result in energy efficiency for traffic delivery. We

presented energy efficient schemes for minimizing the overhead of securing the traffic delivery to the multicast group.

When traffic needs to be delivered to a subgroup of the entire multicast group in a secure manner, optimizing the routing procedure does not result in high energy efficiency for data delivery. Since the traffic is encrypted, the subgroup has to share a common key, only known to its members, in order to be able to decrypt the transmitted information. If a common key is not shared among subgroup members, multicast essentially becomes unicast, since the same data has to be encrypted with different keys and transmitted multiple times.

Suppose we need to send the same traffic stream to both M_2 and M_5 on the network of Figure 4(a). Suppose also that key distribution is performed according to the key tree of Figure 4(b). Sicne M_2 and M_5 do not share a common key only known to them, the only way to deliver the same traffic stream to both in a secure manner, is by encrypting the data twice with keys $K_{3,4}$ and $K_{3,7}$ respectively. Hence, the energy expenditure for sending the stream to M_2 and M_5 is equal to $(Q(E_{S_1^2}(R) + E_{S_1^5}(R)))$, where Q is the number of messages of the traffic stream. If M_2 and M_5 share a common key only known to them, the energy expenditure for sending the traffic stream would be equal to $(Q(E_{S_1^5}(R)))$. Hence, *the energy savings would be* $(Q(E_{S_1^2}(R)))$.

If we assume that the number of active subgroups is small, it would be energy efficient to establish a SEK for every distinct subgroup receiving a data stream. The SEKs can be delivered by using appropriate KEKs from the key distribution tree, in order to reduce the overhead. For example, if members $\{M_8, M_6, M_3, M_7\}$ form a subgroup that has to receive identical information, keys $K_{1,1}$, $K_{2,3}$ can be used to encrypt the and transmit the SEK to all members of the subgroup, and the established SEK can be used to encrypt the data stream directed to the subgroup. If members of the subgroup already share a common key ($\{M_8, M_6, M_9, M_2\}$ share $K_{1,1}$), there is no need to establish an additional SEK.

We need to further investigate the impact of subgrouping in the energy expenditure of the identical traffic delivery.

Application of clustering in non-homogeneous terrain: The performance of the location-aware algorithm depends on the homogeneity of the area where it is applied. If an identical path loss model cannot be used throughout the area where the ad hoc network is deployed, power needed for communication is no longer proportional to the distance with the same proportonality constant. Hence, the location-aware algorithm has to be applied into homogeneous sub-regions, resulting in sub-clusters that have to be combined to provide the hierarchical key structure. An energy efficient method has to be developed to provide a systematic way for combining the sub-clusters, or even take into account the path loss model dissimilarity before any clustering.

X. Conclusion

We showed that scalable key management shemes that were developed for secure multicast in wired networks, are energy inefficient for wireless ad hoc networks. We showed that the network topology has to be taken into account into the design of the key distribuiton schemes. We developed a location-aware key distribution algorithm that uses geographical location information and exploits the spacial correlation of the nodes. We formulated an optimization problem for energy efficient key distribution when routing information is available, and showed that an optimal solution will not scale with group size. Hence, we proposed a simple energy efficient routing-aware heuristic shceme. We also developed an energy-aware sheme, to perform batch rekeying instead of individual rekeying in every membership change. We illustrated the application of our algorithms in binary trees and provided simulation results to show the improvements achieved. Finaly, we provided an energy efficient way to direct traffic to subgroups of the multicast group.

References

[1] D. M. Wallner, E. C. Harder and R. C. Agee, "Key Management for Multicast: Issues and Architectures," INTERNET DRAFT, Sep. 1998.

[2] D. Carman, P. Kruus and B. Matt, "Constraints and Approaches for Distributed Sensor Network Security," NAI Labs Technical Report #00-010 Sep. 2000.

[3] C. K. Wong, M. Gouda and S. Lam, "Secure Group Communications Using Key Graphs," IEEE/ACM Trans. On Networking Vol.8, No.1, pp. 16-31, Feb. 2000.

[4] D. Balenson, D. McGrew and A. Sherman, "Key management for large dynamic groups: One-way Function trees and amortized Initialization," INTERNET DRAFT, Feb. 1999.

[5] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor and B. Pinkas, "Multicast Security: A Taxonomy and Some Efficient Constructions," In Proc. of IEEE INFOCOM 99.

[6] A. Perrig, D. Song and D. Tygar, "ELK, a new protocol for efficient large-group key distribution," In Proc. of the IEEE Security and Privacy Symposium 2001, May 2001.

[7] F. Stajano and R. Anderson, "The resurrecting duckling: Security issues for ad-hoc wireless netowrks," In Security Protocols, 7^{th} International Workshop, 1999.

[8] L. Eschenauer and V. Gligor, "A key-management scheme for distributed sensor networks," In Proc. of the 9^{th} ACM Conference on Computer and Communications Security Washington D.C., USA, November 2002.

[9] H. Chan, A. Perrig and D. Song, "Random Key Predistribution Schemes for Sensor Networks," in IEEE Symposium on Security and Privacy, California, USA, May 2003.

[10] Z. Haas and L. Zhou, "Securing ad hoc networks," in IEEE Network Magazine, vol. 13, no. 6, pp. 24-30, Nov./Dec. 1999.

[11] S. Yi and R. Kravets, "Key management for heterogeneous ad hoc wireless networks," University of Illinois at Urbana-Champaign, Department of Computer Science Technical Report #UIUCDCS-R-2001-2241, UILU-ENG-2001-1748, July 2002.

[12] S. Capkun, L. Buttyan, and J. Hubaux "Self-Organized Public-Key Management for Mobile Ad-Hoc Networks," in IEEE Transactions on Mobile Computing (TMC) 2002.

[13] S. Mki, T. Aura, and M. Hietalahti, "Robust membership management for ad-hoc groups," In Proc. 5^{th} Nordic Workshop on Secure IT Systems (NORDSEC 2000), Reykjavik, Iceland, October 2000.

[14] Educational Observatory Institute GPS page, available via WWW at URL: http://www.edu-observatory.org/gps/gps.html.

[15] T. Hastie, R. Tisbshirani and J. Friedman, *The Elements of Statistical Learning, Data Mining, Inference and Prediction*, Springer Series in Statistics, NY, 2001.

[16] J.E. Wieselthier, G.D. Nguyen and A. Ephremides, "On the Construction of Energy Efficient Broadcast and Multicast Trees in Wireless Networks," in Proc. IEEE INFOCOM 2000, Tel-Aviv, Israel, pp. 586-594.

[17] Y. Yang, X. Li and S. Lam, "Reliable Group Rekeying: Design and Performance Analysis," in Proc. of IEEE ACM SIGCOMM 2001, San Diego, CA, USA, Aug. 2001.

[18] X. Li, Y. Yang, M. Gouda and S. Lam, "Batch Re-keying for Secure Group Communications," in Proc. of World Wide Web 10 (WWW10), Hong Kong, China, May 2001.

[19] S. Setia, S. Koussih and S. Jahodia, "Kronos: A Scalable Group Re-Keying Approach for Secure Multicast," In Proc. of the IEEE Security and Privacy Symposium 2000, Oakland, CA, USA, May 2000.

[20] J. Edmonds, "Maximum Matching and a Polyhedron with (0,1) vertices," in Journal of Research of the National Bureau of Standards, 69B, pp:125-130, April-June 1965.

[21] E.Lawyer, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, Ney York 1976.

[22] H. Gabow, *Implementation of Algorithms for Maximum Matching and Non Bipartite Graphs*, PHD Thesis, Stanford University, 1974.

[23] H. Gabow, "Data Structures for Weighted Matching and Nearest Common Ansestors with Linking," in Proc. of the First Annual ACM SIAM Symposium on Discrete Algorithms (SODA '90), pp:434-443, San Francisco, CA, USA, Jan 1990.

[24] W. Cook and A. Rohe, "Computing Minimum-Weight Perfect Matchings," Technical Report 97863, Forschungsinstitut für Diskrete Mathematik, Universität Bonn, 1997.

[25] I. Kang and R. Poovendran, "On the Lifetime Extension of Energy-Constrained Multihop Broadcast Networks," WCCI 2002, Hawaii, USA, May 2002.

[26] L. Tassiulas and J. Chang, "Maximum Lifetime Routing in Wireless Sensor Networks," in Proc. of the ARL Federated laboratory 4^{th} Annual Symposium, College Park, MD, USA, March 2000.

APPENDIX

The goal of K-means is to create K clusters out of N points ($K < N$) such that a "loss" cost function is minimized with respect to a dissimilarity measure. K-means uses the squared Euclidean distance as a dissimilarity measure. If the coordinates of point i are $x_i = (x_{i_1}, x_{i_2})$, the Euclidean distance is equal to:

$$d(x_i, x_{i'}) = \sum_{j=1}^{2} \left(x_{ij} - x_{i'j}\right)^2 = \|x_i - x_{i'}\|^2 \tag{9}$$

When the propagation loss factor is $\gamma = 2$, the Euclidean distance is proportional to the transmission power and is suitable for our clustering procedure. In case where $\gamma \neq 2$, we can modify the dissimilarity measure to be proportional to the transmission power as in (10).

$$d(x_i, x_{i'}) = \|x_i - x_{i'}\|^{\gamma} \tag{10}$$

The "loss" function that is minimized in K-means is:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} \|x_i - x_{i'}\|^2 \tag{11}$$

$$W(C) = \sum_{k=1}^{K} \sum_{C(i)=k} \|x_i - m_k\|^2 \tag{12}$$

where $C(i) = k$ symbolizes the assignment of the i^{th} member to the k^{th} cluster, m_k is the mean vector of cluster k and C is the resulting cluster configuration (N members assigned into K clusters). The "loss" function is minimized by assigning the N members to K clusters in such a way, that within each cluster the average dissimilarity (distance between points assigned to the same cluster and the cluster mean) is minimized.

$$C^* = \arg\min_{C} \sum_{k=1}^{K} \sum_{C(i)=k} \|x_i - m_k\|^2 \tag{13}$$

We can obtain C^* by solving the enlarged optimization problem

$$\min_{C, \{m_k\}_1^K} \sum_{k=1}^{K} \sum_{C(i)=k} \|x_i - m_k\|^2 \tag{14}$$

where m_k is the mean of the k^{th} cluster. The iterative algorithm for solving (14) is as follows [15]:

K-means Algorithm

Step 1: Have an arbitrary assignment C of points into the specified number of clusters K (initialization can be done by assigning the i^{th} point to the i mod K cluster). Compute the mean vector for each cluster.

Step 2: For the given assignment C, the cluster variance as expressed in (13) is minimized with respect to $\{m_1, \ldots, m_K\}$ yielding the means of the currently assigned clusters

Step 3: Given a current assignment of means $\{m_1, \ldots, m_K\}$, (14) is minimized by assigning each point to the closest cluster mean. That is,

$$C(i) = \arg \min_{1 \leq k \leq K} \|x_i - m_k\|^2 \tag{15}$$

Step 4: Iterate step 2 and 3 until the cluster memberships do not change.

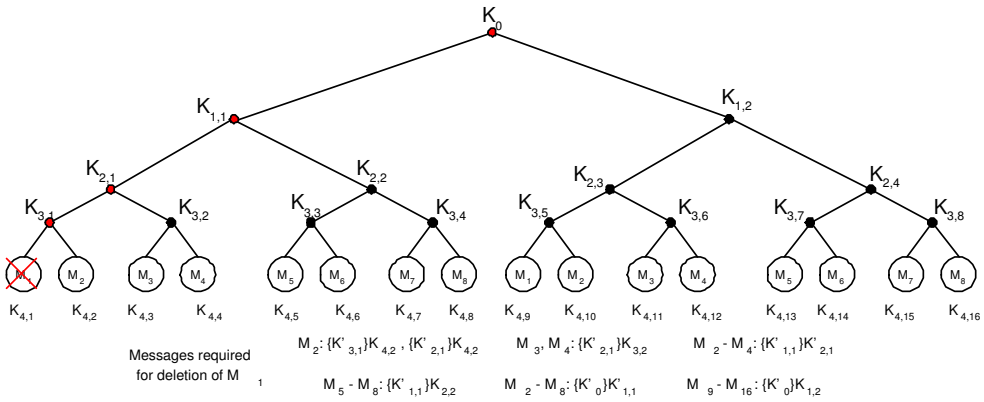
Assume we have an ad hoc network of $N = 17$ nodes forming a multicast group with S being the GC sending information to members $M_1 - M_{16}$. Assume also that sorting the energies for reaching each member from the GC results in E=$\{E_{\{M_1\}}, E_{\{M_2\}}, E_{\{M_3\}}, E_{\{M_4\}}, E_{\{M_5\}}, E_{\{M_6\}}, E_{\{M_7\}}, E_{\{M_8\}}, E_{\{M_9\}}, E_{\{M_{10}\}}, E_{\{M_{11}\}}, E_{\{M_{12}\}}, E_{\{M_{13}\}}, E_{\{M_{14}\}}, E_{\{M_{15}\}}, E_{\{M_{16}\}}\}$. Assume that we build 2 different key trees, one with degree $d = 2$ and one with degree $d = 8$ as shown in Figure 9(a) and Figure 9(b), and that member M_1 is leaving the multicast group. The key tree in Figure 9(a) needs total of 7 messages for deleting M_1, while the key tree of Figure 9(b) needs 8 messages. However, if:

$$2E_{M_2} + E_{\{M_3,M_4\}} + E_{\{M_2..M_4\}} + E_{\{M_5..M_8\}} + E_{\{M_9..M_{16}\}} > \sum_{i=2}^{8} E_{\{M_i\}} + E_{\{M_2..M_8\}} + E_{\{M_9..M_{16}\}} \tag{16}$$

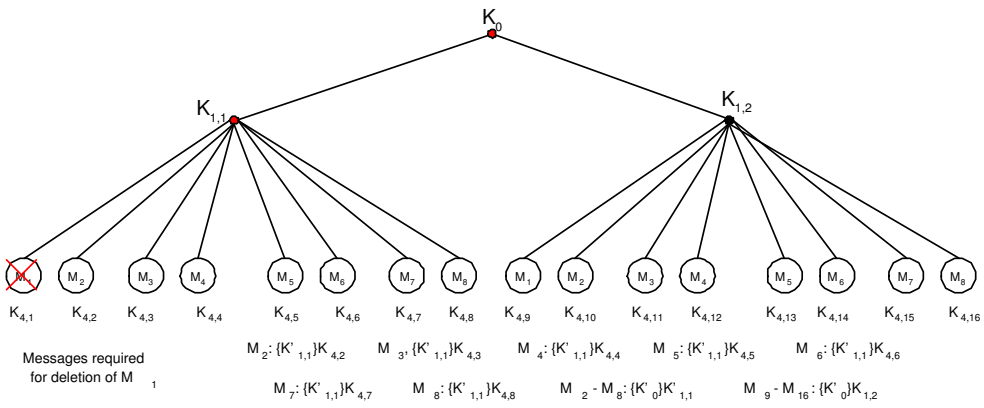
or equivalently

$$E_{\{M_2\}} + E_{\{M_3,M_4\}} + E_{\{M_2..M_4\}} + E_{\{M_5..M_8\}} > \sum_{i=2}^{8} E_{\{M_i\}} \tag{17}$$

the tree of degree $d = 8$ requires less energy to be rekeyed than the tree of degree $d = 2$ when M_1 leaves the multicast group.



(a)



(b)

Fig. 9. (a) Rekeying when $d = 2$ and member M_1 leaves the group. (b) Rekeying when $d = 8$ and member M_1 leaves the group.