Power Proximity Based Key Management for Secure Multicast in Ad Hoc Networks

Loukas Lazos and Radha Poovendran

Network Security Lab, Dept. of EE,

University of Washington, Seattle, WA 98195-2500

{l_lazos, radha}@ee.washington.edu

Abstract

As group-oriented services become the focal point of ad hoc network applications, securing the group communications becomes a default requirement. In this paper, we address the problem of group access in secure multicast communications for wireless ad hoc networks. We argue that energy expenditure is a scarce resource for the energy-limited ad hoc network devices and introduce a cross-layer approach for designing energy-efficient, balanced key distribution trees to perform key management. To conserve energy, we incorporate the network topology (node location), the "power proximity" between network nodes and the path loss characteristics of the medium in the key distribution tree design. We develop new algorithms for homogeneous as well as heterogeneous environments and derive their computational complexity. We present simulation studies showing the improvements achieved for three different but common environments of interest, thus illustrating the need for cross-layer design approaches for security in wireless networks.

Index Terms

ad hoc networks, key management, multicast, security, energy efficiency

I. Introduction

The idea of many wireless devices being able to collaborate via direct communication over a short range without the need for a pre-deployed network infrastructure, has recently become feasible. Wireless ad hoc networks have many civilian applications such as on-demand network deployment for emergency-rescue operations and disaster-relief efforts, patient monitoring and drug inventory management in hospitals, home networking, as well as military applications such as deployment of surveillance networks, target monitoring and real time information distribution to mobile military units. Almost all of these applications rely on the capability of peer-to-peer collaboration in computation and/or communications.

A salient feature of the wireless devices is the limited battery capacity on which they operate. Hence, these devices are constrained in both computational power and communication capability. Advances in VLSI technology continue to reduce the energy consumed during the execution of instruction sets [18]. However, when battery operated devices are wirelessly networked, the energy expenditure due to communication—a function of the propagation medium—has been noted as the dominant factor in battery depletion [6], [18]. In fact, studies in [18] show that the ratio of the energy required to transmit one bit of information compared to the energy required to execute one instruction is on the order of 1,500 to 2,200 for the microsensors in [29]. To

extend the lifetime of the devices, it is critical that both the communication model and the algorithms used, reduce the energy requirements for communication.

Multicast is used on several critical network applications such as: routing, neighbor discovery, key distribution and topology control, as well as many ad hoc network applications, which require identical data to be sent from a single sender to multiple receivers. For the single-sender multiple-receiver model, adopting the multicast communication model reduces the network traffic and allows the sender to conserve energy consumed in data processing. Moreover, in wireless communication mode, due to the broadcast nature of the wireless medium, multicasting has the potential to further reduce the network traffic in number of messages compared to wired multicast, and hence, reduce the network energy expenditure. A single broadcast transmission will reach any receiver within the communication range [1] of the source. However, anyone in range can listen to the information being broadcasted over the wireless medium. For applications requiring information privacy, it is important to ensure that only the intended receivers have access to the group communication at any given time.

Encrypting the information that is transmitted over the open wireless channel is the most common technique for securing the multicast communication[2]. When the amount of data to be transmitted is large or when the devices involved in communication do not have the capability to perform computationally intense exponentiations (public key cryptography), symmetric key cryptography is used to secure the communications. However, the use of a symmetric key requires all valid receivers to hold the decryption key for decrypting a multicast message. This shared decryption key is called Session Encryption Key (SEK) or Traffic Encryption Key (TEK). In order to preserve the secrecy of the multicast data, the SEK needs to be updated each time a member joins/leaves the multicast group [30]. Since everyone shares the SEK, members need to hold additional Key Encryption Keys (KEK) that are used to securely distribute the SEK to each valid member. The *key management* problem is to ensure that only legitimate members of the multicast group hold valid keys at any time during a multicast session. In the presence of group members that join or leave the multicast group, the key management problem can be reduced to the problem of finding efficient mechanisms to assign and distribute cryptographic keys.

Key distribution in wired networks is a well studied problem [20], [27], [30], [32]. For wired networks, the amount of rekey message overhead, the rekeying time delay under a member deletion and to some extent, the user key storage are the main problem constraints. The solutions that have been developed in the literature yield lower number of key update overhead messages, while not requiring prohibitive user storage [20], [27], [30], [32]. The key tree approach to key distribution that were introduced in [27], [30] has now been adopted with modifications in the Internet Community [2] for secure multicast. It has also been shown that the key trees indeed reduce the average number of update messages and a lower bound of $\mathcal{O}(\log_{\alpha} N)$ for a tree of degree α with group size N has been derived [23]. Hence, key trees are presented as a bandwidth-efficient [3] solution for secure multicast. However, the key distribution solutions for wired networks neither incorporate energy expenditure of the relay nodes as a parameter nor do they consider the network topology as part of the design. Instead, the key distribution trees are chosen randomly. i.e. independently of the node placement and network topology.

[1]The communication range is defined as the maximum distance from the transmitter, to a receiver, so that the Signal to Noise Ratio (SNR) is above the required threshold for communication.

[2]Additionally, cryptography can also support group access control for dynamic multicast groups through secure management of the cryptographic keys [4].

[3]In key distribution, the bandwidth requirement is defined as the number of messages sent by the group controller for updating keys.

In ad hoc wireless networks where the nodes have limited battery capacity, the energy spent by each node transmitting the data is a valuable resource and has to be efficiently used for network management operations such as key distribution. Hence, the key trees have to be designed incorporating energy as a design metric. Direct application of key distribution schemes of wired networks prove to be energy inefficient for the wireless ad hoc environment [12], [13].

In this paper, we study the problem of designing energy-efficient key distribution trees for secure wireless multicast, while preserving the bandwidth and storage efficiency exhibited by the key trees used in wired networks. Balanced key trees require an identical number of key update messages regardless of who is being deleted [23]. In addition, since a balanced tree assigns identical number of keys to every member, the key storage requirement is the same for every member. Hence, we address the question of designing balanced key trees that incorporate energy as a design parameter. In order to reduce the energy expenditure (physical layer parameter) of the key distribution (application layer operation), we propose a cross-layer design approach that incorporates, (a) the network topology (location of the nodes) and, (b) the propagation medium characteristics (physical layer). To our knowledge, our work is the first scheme, design or algorithm, that explicitly accounts for the impact of the energy consumption due to communication in the design of the secure multicast protocol. We also believe ours is the first cross-layer design approach in wireless security[4]. The figure 1 shows the type of cross-layer interaction used in the design of the key trees.

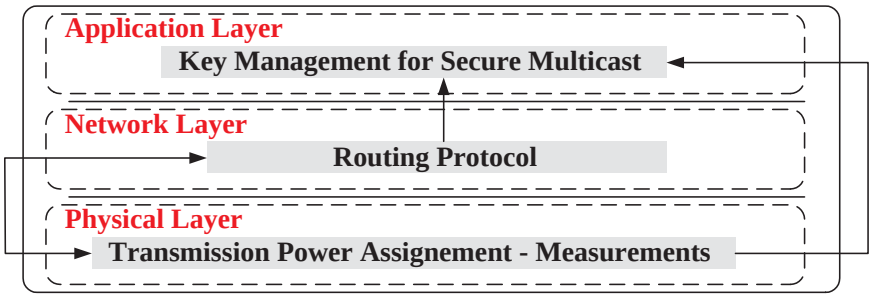


Fig. 1. Schematic of the type of cross-layer interaction that is used in the energy-efficient key tree design.

The following are our contributions in this paper:

- *Metric for Energy Efficiency:* Since the energy required to transmit the updated keys to a member depends on the network topology and path loss parameters of the medium, we introduce the average key update energy as a metric to evaluate the energy efficiency of key trees. We show that under the new metric, the communication overhead of a key distribution scheme is tied to the network topology (relative node location). This approach is significantly different from the approach taken by others for energy-constrained wireless networks [6], [8], [25], [26], [33]. We also study the properties of the average key update energy in terms of the network size, key tree degree and medium path loss model, and derive an upper bound of the metric in terms of these parameters. We optimize the degree of the key tree to derive the lowest upper bound. We also interpret related work in [30] and [4].
- *Incorporation of the Energy in Balanced Key Tree Design:* Observing that energy savings occur when an identical message is delivered to a set of nodes reached by overlapping routing paths, we define and use the idea of "power proximity" to group nodes in the key tree and reduce the network energy expenditure.

[4]Preliminary results of our work appear in IEEE ISWC'02 [12], IEEE ICASSP'03 [13] and in IEEE ICC'04 [14].

- *Homogeneous Medium with Constant Path Loss:* We also make the observation that when the transmission medium is homogeneous with constant attenuation factor, "power proximity" property is a monotonically increasing mapping to physical proximity. Hence, we replace "power proximity" with physical proximity by using Euclidean distance. This, in turn leads to a low complexity algorithm for constructing energy-efficient key trees that make use of location information alone. We then show that the computational cost for optimally grouping nodes based on physical proximity grows prohibitively high with the multicast group size and hence the sub-optimal solution is adopted.
- *Heterogeneous Medium with Variable Path Loss:* When the medium is heterogeneous, we note that due to varying path loss parameter, "power proximity" is no longer a monotonic mapping to physical proximity. In this case, we directly incorporate "power proximity" by considering the transmission power in grouping nodes in the key tree. We develop two algorithms, that combine node location with path loss medium characteristics to generate energy-efficient balanced key trees.

The remainder of the paper is organized as follows. In Section II, we present related work. In Section III, we describe the network model assumed and the desired features of an energy-efficient key management scheme for ad hoc networks. In Section IV, we introduce the average key update energy, a new metric for evaluating the energy efficiency of key distribution. In Section V we analyze the impact of "power proximity" on the key distribution, under the new metric. In Section VI, we present our physical proximity based key distribution scheme for the homogeneous medium. In Section VII, we present our "power proximity" based key distribution scheme for networks deployed in a heterogeneous medium. In Section VIII, we provide simulation results and show the improvements achieved by our algorithms. The Section IX presents conclusions.

II. Related Work

A. Notation

Before we present related work, we provide the notation that will be used through the rest of the paper.

GC	Group Controller.	$d_{i,j}$	Distance between nodes i,j.
N	Multicast group size.	γ	Path loss attenuation factor.
SG	A sub-group of MG, i.e. $SG \subseteq MG$.	$P(d_{i,j})$	Transmit power needed for commu-
MG	Multicast Group.		nication between nodes i,j.
M_i	The i^{th} member of the multicast group.	$diss(i,j)$	Dissimilarity between nodes i,j.
T	Key distribution tree.	$K_{l.j}$	Key assigned to the j^{th} node of the
α	Degree of the tree T.		l^{th} level of the tree T.
$\{m\}_{K_{l.j}}$	Message m encrypted with key $K_{l.j}$.	$A \rightarrow S : m$	A sends m to members of S.

B. Related Work

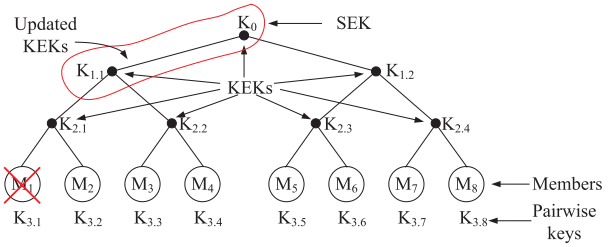
Providing secure multicast communications in an ad hoc network environment is a fairly unexplored research area. Very few key management schemes have recently been proposed for such networks. In [11], the authors propose a key management scheme for secure multicast in Mobile Ad Hoc Networks (MANET). All members of the multicast group belong to a physical security tree and each member is assumed to have GC privileges. Hence, it can act as an intermediary to allow the join and execute the revocation of any member attached to it. Joining members are attached to the best one hop node that minimizes the message

exchange for a join operation. Revocation is done periodically through the flooding of a certificate revocation list. Two significant differences between [11] and our work is that in [11], the authors assume the availability of mobile devices that are capable of performing asymmetric cryptographic operations, and do not consider the energy expenditure of the communication overhead as a design parameter.

In [5], [8], [33], key management for wireless ad hoc networks is performed based on public key cryptography. In [8], [33], the authors employ a threshold cryptography scheme to verify the authenticity of public-key certificates. The private key of a member is divided into n shares and shares are assigned to n nodes that operate as servers. In order to verify the authenticity of a certificate signed by a member's private key, $(t+1)$ servers are required to provide their partial signature based on their share ($(n, t+1)$ threshold scheme). A combiner collects the partial signatures and verifies the authenticity of the certificate. In [5], the authors present a decentralized and self-organizing scheme, that allows public key management. In this paper, we assume that the wireless devices are energy and computationally limited and do not have the ability to support public key based communication and computations. Hence, we rely on symmetric key based scheme for secure communications. We now review Virtual Key Tree or Logical Key Hierarchy, one of the most widely used key distribution scheme that is suitable for symmetric keys in large networks.

C. Logical Key Hierarchy Based on Key Graphs

Key trees were independently proposed for group access control in wired networks, in [27] and [30]. The use of key trees addresses the scalability of the key distribution in terms of bandwidth and storage requirements.



(a)

Deletion of member M_1

(1) $GC \rightarrow M_2$: $\{K'_{1.1}\}_{K_{3.2}}$

(2) $GC \rightarrow \{M_3, M_4\}$: $\{K'_{1.1}\}_{K_{2.2}}$

(3) $GC \rightarrow \{M_2 - M_4\}$: $\{K'_0\}_{K'_{1.1}}$

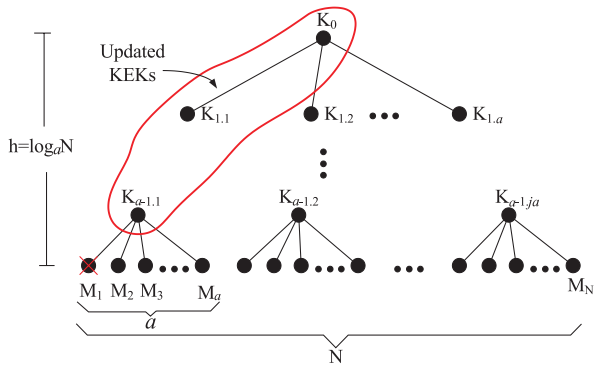
(4) $GC \rightarrow \{M_5 - M_8\}$: $\{K'_0\}_{K_{1.2}}$

(b)

Fig. 2. (a) A binary logical hierarchical key tree. Members are placed at the leaf nodes. Each member holds the keys traced along the path from the leaf to the root of the tree. If M_1 leaves the multicast group all keys known to him (K_0, $K_{1.1}$) are updated. (b) Update messages in the order in which they are sent by the GC after M_1 leaves the multicast group.

In figure 2(a), we present a binary key distribution tree used in wired networks for a multicast group of $N = 8$ members plus the GC. Such a tree is a visual representation of the key assignment to each member of the multicast group. Referring to figure 2(a), each member is associated with a single leaf node of the tree. Each member is assigned keys that are traced along the path from the leaf node to the root [27]. For example, M_1 is assigned keys $\{K_0, K_{1.1}, K_{2.1}, K_{3.1}\}$. K_0 is used as the SEK, while the rest of the keys are KEKs.

For dynamic multicast groups, member deletions trigger a significantly higher number of key updates than member joins [30]. Hence, scalable key tree solutions focus on reducing the number of key updates in case of a member deletion [4], [27], [30]. If M_1 leaves the multicast group, keys $\{K_0, K_{1.1}\}$ need to be updated and the messages shown in figure 2(b) need to be sent to appropriate group members. Note that key $K_{2.1}$ need not be updated, since it is not shared by multiple members and M_2



(a)

Messages sent from GC	Number
$\{K'_{\alpha-1.1}\}_{K_{\alpha.i}},\ i=2\ldots\alpha$	$\alpha-1$
$\{K'_{\alpha-2.1}\}_{K'_{\alpha-1.1}}$	1
$\{K'_{\alpha-2.1}\}_{K_{\alpha-1.i}}, i=2\ldots\alpha$	$\alpha-1$
$\ldots$	$\ldots$
$\{K'_0\}_{K'_{1.1}}$	1
$\{K'_0\}_{K_{1.i}},\ i=2\ldots\alpha$	$\alpha-1$
Total number of messages sent	$\alpha\log_\alpha N-1$

(b)

Fig. 3. (a) An α-ary hierarchical tree of height $h=\log_\alpha N$. After the deletion of member M_1, the $\log_\alpha N$ keys traced from M_1 to the root (except for the pariwise key shared between M_1 and the GC) of tree need to be updated, (b) the update messages sent from the GC to sub-groups to update the KEKs and SEK due to the deletion of M_1.

already shares pairwise key $K_{3.2}$ with the GC.

In figure 3(a), we show the general case of an α-ary hierarchical tree of height $h=\log_\alpha N$. We can verify that the communication for deleting a member from the multicast group is $\alpha\log_\alpha N-1$ messages [4]. If M_1 is deleted, the GC needs to update all the keys traced at the path from M_1 to the root of the tree, except for the pairwise key shared between M_1 and the GC (a total of $\log_\alpha N$ keys). In figure 3(b), we show the encrypted messages sent by the GC to update all keys (except for the pairwise key) known to member M_1. The GC needs to send $(\alpha-1)$ messages to update $K_{a-1.1}$ ($(\alpha-1)$ members hold $K_{a-1.1}$ after the deletion of M_1), and α messages to update each of the rest $\log_\alpha N-1$ keys. Hence, the number of keys sent by the GC for deleting M_1 or any other member is equal to $\alpha\log_\alpha N-1$. In [4], the authors propose the use of key trees in conjunction with pseudo-random functions to reduce the communication cost to $(\alpha-1)\log_\alpha N$ messages per member deletion.

As noted earlier, the existing key management solutions for wired networks [20], [27], [30], [32] are independent of the network topology. Network topology was considered in the key management for wireless cellular networks in [25], [26] where the authors propose a hierarchical key tree structure with each base station acting as a GC. The key tree used for secure multicast in [25], [26] was matched to the network infrastructure, by grouping together members that communicate with the same base station, in the same sub-tree of the key distribution.

None of the solutions that were developed for the wired, wireless cellular, or ad hoc networks, have attempted to build key trees that would consume the minimum key update energy. Since the trees are known to have $\mathcal{O}(\log N)$ key update messages, they are ideal candidates to be used in bandwidth-constrained environments including wireless networks [6]. Hence, building low energy consuming trees is a natural and important generalization for wireless environments. *To the best of our knowledge, our work is the first one that introduces the cross-layer design approach into the key distribution. In doing so, we incorporate the parameters such as location, medium pathloss and physical and power proximity, which influence the physical and network layer into the application layer design.* We now state the network model assumptions.

III. Network Model Assumptions

Network generation: We assume that the network consists of N multicast members plus the GC, randomly distributed in a specific area. We consider a single-sender multiple-receiver communication model. All users are capable of being relay nodes

and can corroboratively relay information between an origin and destination. Hence, the network is multi-hop. We also assume that the network nodes have the ability to generate and manage cryptographic keys.

Node location acquisition: The nodes of the network are assumed to be in a fixed location, after their initial placement. We assume that nodes have a mechanism to acquire their location information. Such information is often obtainable through the Global Positioning System (GPS) [7]. However, in many cases, GPS may not be available due to the expensive hardware required (e.g. sensor networks), or the lack of obstacle-free communication (indoor setting). Several approaches have been proposed for acquiring location information without a GPS receiver [1], [17], [21]. After a node correctly acquires its own location information, it can report the coordinates to other nodes through a location service algorithm [31].

Network initialization: We assume that the network has been successfully initialized and initial cryptographic quantities (at least pairwise trust) have been distributed. Several novel approaches that address the critical problem of secure initialization in resource-constrained ad hoc networks have recently been presented in [6], [24].

We further assume that the underlying routing is optimized in order to minimize the total power required for broadcast. Although it is known that finding the optimal solution for total minimum power broadcast is NP-complete [3], [16], several heuristics resulting in routing trees with satisfactory performance have been proposed in the recent literature [3], [15], [28].

Since our goal in this paper is to design key management algorithms and not protocols, we do not address the MAC layer implementation requirements of our algorithms.

Wireless medium and signal transmission: We consider the cases of a homogeneous and heterogeneous medium separately, since the complexity and inputs of the algorithms that we propose differ depending on the type of the medium. In the case of the homogeneous medium, we assume that the transmission power $P(d_{i,j})$ required for establishing a communication link between nodes i and j, is proportional to a constant exponent (attenuation factor γ) of the distance $d_{i,j}$, i.e. $P(d_{i,j}) \propto d_{i,j}^{\gamma}$. For simplicity, we set the proportionality constant to be equal to one. An example of a homogeneous path loss medium is an obstacle-free, open space terrain with Line of Sight (LOS) transmission. Note that for fixed length messages, transmission power is proportional to energy expenditure and vice versa.

For a heterogeneous medium, no single path loss model may characterize the signal transmission in the network deployment region. Even when node locations are relatively static, path loss attenuation can vary significantly when the network is deployed in mountains, dense foliage, urban region, or inside different floors of a building. In [19], different path loss models have been presented based on empirical data. Two most common models with varying path loss for calculating the power attenuation at a distance d from the transmitter are: (a) Suburban area - A slowly varying environment where the attenuation loss factor changes slowly across space. (b) Office building - A highly heterogeneous environment where the attenuation loss factor changes rapidly over space. We will use these models in simulations where we illustrate our algorithms.

Antenna model: We assume that omnidirectional antennas are used for transmission and reception of the signal [28]. The omnidirectionality of the antennas results in a property unique in the wireless environment known as the *wireless broadcast advantage* (WBA) [28]. When the sender transmits a message to a node, all nodes that lie within the transmission range can receive the broadcasted message for free. Hence, when an identical message needs to be sent to multiple receivers, the sender can significantly reduce the energy expenditure by directly transmitting the data to the farthest member. However, in secure multicast

the broadcast advantage can be exploited *only if* more than one receiver within the range holds the decryption key. Hence, the use of omnidirectional antenna does not guarantee WBA when the security is an added feature. We further assume that signal transmission is the major component of energy expenditure and ignore any energy cost due to computation and information processing [6], [18].

A. Design features of the key management scheme

To enable large group multicast in a wireless environment that is bandwidth and energy-constrained, the key management scheme shall realize the following design features:

A_1 : Secure key distribution of the cryptographic keys to valid members of the multicast group.

A_2 : Scalability in bandwidth and key storage requirements with respect to the group size N.

A_3 : Energy efficiency in the distribution of key updates, in the event of membership change.

A_4 : Providing the ability to reach a member even if all other members of the group were to be deleted.

Tree based key schemes in wired networks satisfy items A_1, A_2, and A_4. Moreover, it was shown that when bandwidth is a constraint, the optimal average number of keys to be assigned to member is $\mathcal{O}(\log N)$, where N is the group size. Keys trees achieve this bandwidth bound [27], [30] with a required user key storage of $\mathcal{O}(\log N)$.

However, at present, there is no design approach that incorporates the energy expenditure due to key updates in the design of the tree-based key schemes. We now show how to develop a suitable metric for design of energy-efficient key trees.

IV. An Energy Oriented Metric for Key Management in Wireless Ad Hoc Networks

In this section we introduce the average energy required to rekey a member as the energy efficiency metric of key trees. We then show that the newly defined metric is a function of the path loss model, the network topology and the key tree degree. We derive upper bounds on the average key update energy in terms of the path loss parameter, degree of the key distribution tree and the network area size. We then derive the degree of the tree that will yield the lowest upper bound. We also discuss how to derive the tree degrees reported in other literature [4], [30].

A. Average key update energy

Let $\tilde{E}_{M_i}$ denote the energy expenditure for updating the compromised keys after the deletion of the i^{th} member. Also, let $p(M_i)$ denote the probability for member M_i to leave the multicast group. We define the average key update energy, E_{Ave}, required for key update after a member deletion as:

$$E_{Ave} = \sum_{i=1}^{N} p(M_i)\tilde{E}_{M_i}. \tag{1}$$

We define the design metric update energy in the average sense, since each member requires different energy to receive a key update, while we are interested in constructing a key tree for all the members. As an example, in figure 2(a), if member M_1 were to be deleted, the messages shown in figure 2(b) need to be transmitted. On the other hand, if M_8 were to be deleted, the following message transmissions have to take place in the order listed:

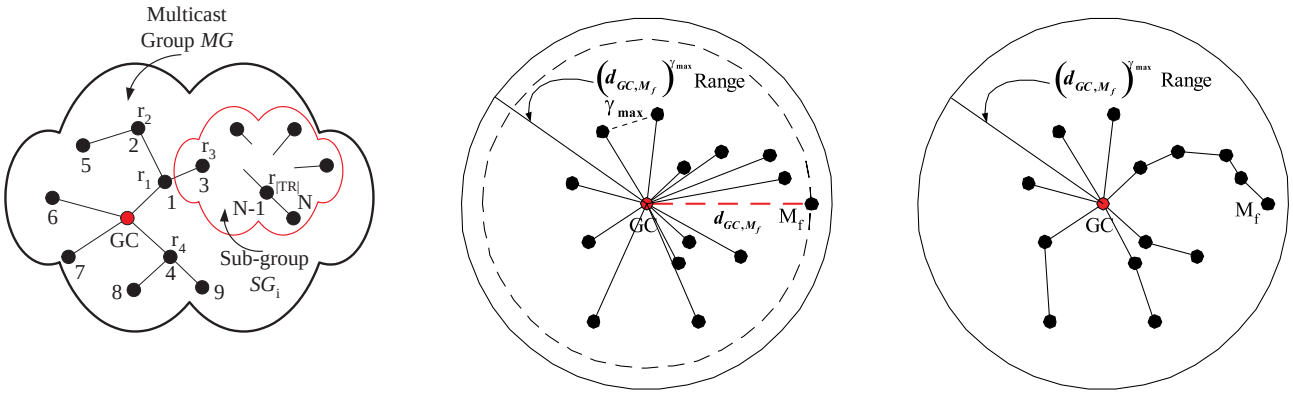


Fig. 4. (a) Theorem 1: When a message is sent to all members of MG, all relay nodes $TR = \{r_1, r_2, \ldots, r_{|TR|}\}$ have to transmit. When a message is sent to any sub-group $SG_i \subseteq MG$, a subset $TR_i \subseteq TR$ of relay nodes need to transmit. (b) Theorem 2: A single transmission of power $(d_{GC,M_f})^{\gamma_{max}}$ reaches all members of MG with one hop, resulting in routing tree R, (c) The optimal multicast routing tree R^* always requires less power than R, by definition.

$$(1)\ \ GC \rightarrow M_7: \quad \{K'_{1.2}\}_{K_{3.7}} \qquad (2)\ \ GC \rightarrow \{M_5, M_6\}: \quad \{K'_{1.2}\}_{K_{2.3}}$$
$$(3)\ \ GC \rightarrow \{M_5 - M_7\}: \quad \{K'_0\}_{K'_{1.2}} \qquad (4)\ \ GC \rightarrow \{M_1 - M_4\}: \quad \{K'_0\}_{K_{1.1}}$$

For any arbitrary network topology, $\tilde{E}_{M_1} \neq \tilde{E}_{M_8}$ i.e. the deletion of M_1 and M_8 result in different energy expenditures. Hence, using E_{Ave} instead of $\tilde{E}_{M_i}$ removes the unwanted dependencies on the specific member being deleted. Note that when all members have equal probability of being deleted, $E_{Ave} = 1/N \sum_{i=1}^{N} \tilde{E}_{M_i}$ is proportional to the total energy expenditure for updating the compromised keys. None of the metrics proposed for the wired networks, with the exception of time delay, is calculated with network dependency. We now examine the properties of E_{Ave} and derive the upper bound for it.

B. Dependency of E_{Ave} on the group size N, the tree degree α, and the network topology

In this section, we examine the dependency of the E_{Ave} on the group size N, the degree of the key distribution tree α and the network topology. E_{Ave} depends on the energy $\tilde{E}_{M_i}$ required to deliver key updates if each member M_i were to be deleted from the multicast group MG. Regardless of which member is deleted, the GC needs to transmit $(\alpha \log_\alpha N - 1)$ key update messages [30]. These messages are routed to different sub-groups $SG_i \subseteq MG$, where $i = 1 \ldots (\alpha \log_\alpha N - 1)$. Letting E_X^R denote the energy expenditure for sending an identical message to every member of the sub-group $X \subseteq MG$ via the routing tree R, the energy expenditure $\tilde{E}_{M_i}$ for updating keys after the deletion of member $M_i \in MG$ is:

$$\tilde{E}_{M_i} = \sum_{i=1}^{\alpha \log_\alpha N - 1} E_{SG_i}^R. \tag{2}$$

The $\tilde{E}_{M_i}$ depends on the routing tree R and cannot be analytically expressed unless a specific realization of R is provided. Hence, we derive an upper bound on $\tilde{E}_{M_i}$, making use of a sequence of properties of the WBA. To do so, we first prove the following theorem:

Theorem 1: The energy required to deliver a message to a sub-group of members $SG_i \subseteq MG$ via a routing tree R, cannot be greater than the energy required to deliver a message to all members of MG, via the same routing tree R.

$$E_{SG_i}^R \leq E_{MG}^R, \quad \forall\ SG_i \subseteq MG. \tag{3}$$

Proof: Let $TR = \{r_1, r_2, \ldots, r_{|TR|}\}$ denote the set of all relay nodes in the multicast routing tree R utilized by the multicast group MG. In order to deliver a single message to every member of the multicast group MG, every node in TR has to transmit. Hence,

$$E^R_{MG} = \sum_{r_i \in TR} E_{r_i}, \tag{4}$$

where E_{r_i} denotes the energy required for transmission of one message by relay node r_i. In order to deliver a message to a sub-group $SG_i \subseteq MG$, a subset $TR_i \subseteq TR$ of the relay nodes has to transmit (no more than the total number of relay nodes can transmit). Hence, $|TR_i| \leq |TR|,\ \forall SG_i \subseteq MG$. The energy to deliver a message to a sub-group SG_i is:

$$E^R_{SG_i} = \sum_{TR_i} E_{r_i} \leq \sum_{TR} E_{r_i} = E^R_{MG}, \quad \forall\ SG_i \subseteq MG \tag{5}$$

■

In figure 4(a) we illustrate Theorem 1. To deliver a message to all members of MG, all relay nodes need to transmit. However, in order to deliver a message to a sub-group SG_i no more than the whole set TR of relay nodes may transmit. Hence, the energy required to deliver a message to any sub-group of MG, is no greater than the energy required to deliver a message to all members of MG.

Using Theorem 1 we can bound the energy expenditure for updating keys after the deletion of M_i expressed in (2) as:

$$\tilde{E}_{M_i} = \sum_{i=1}^{\alpha \log_\alpha N - 1} E^R_{SG_i} \leq E^R_{MG} \left(\alpha \log_\alpha N - 1\right). \tag{6}$$

The bound in (6) holds for an arbitrary routing tree R. Hence, when we use the minimum total power routing tree R^*, by combining (1) and (6), we can bound the average key update energy E_{Ave} as:

$$\begin{aligned} E_{Ave} = \sum_{i=1}^{N} p(M_i)\tilde{E}_{M_i} &\leq \sum_{i=1}^{N} p\left(M_i\right) E^{R^*}_{MG} \left(\alpha \log_\alpha N - 1\right) \\ &\leq E^{R^*}_{MG} \left(\alpha \log_\alpha N - 1\right) \sum_{i=1}^{N} p\left(M_i\right) \\ &\leq E^{R^*}_{MG} \left(\alpha \log_\alpha N - 1\right). \end{aligned} \tag{7}$$

The bound in (7) has two different components. The first component is the minimum energy $E^{R^*}_{MG}$, required for sending a message to the whole multicast group MG. While $E^{R^*}_{MG}$ depends on the wireless medium characteristics and the network topology, we can relax the network topology dependency by bounding $E^{R^*}_{MG}$ using only the wireless medium characteristics and the size of the deployment region.

Let γ_{max} denote the maximum value of the attenuation factor for the heterogeneous medium where the network is deployed, and T_{trans} denote the duration of the transmission of one message. Let M_f denote the farthest member from the GC, and let d_{GC,M_f}, the distance between GC and M_f, denote the radius of the deployment region[5]. Then, the following theorem holds:

Theorem 2: The energy required to deliver a single message to every member of the multicast group MG via the minimum

[5]The diameter or the size of the deployment region may also be defined as the maximum physical distance between any two nodes of the network. However, such a definition leads to a looser upper bound and is not considered.

total power routing tree R^* is no greater than the energy required to deliver a single message from the GC to M_f via a one hop transmission and assuming an attenuation factor of γ_{max} between the GC and M_f.

$$E_{MG}^{R^*} \leq E_{M_f}^{R} = \left(d_{GC,M_f}\right)^{\gamma_{max}} T_{trans}. \tag{8}$$

Proof: Let γ_i denote the attenuation factor in the link between the GC and member M_i, with $\gamma_i \leq \gamma_{max}\ \forall M_i \in MG$. The transmission power required for communication via a one hop link between M_i and GC is, $P(d_{GC,M_i}) = (d_{GC,M_i})^{\gamma_i}$, Since $d_{GC,M_i} \leq d_{GC,M_f},\ \gamma_i \leq \gamma_{max},\ \forall M_i \in MG$, it follows that:

$$P(d_{GC,M_i}) = (d_{GC,M_i})^{\gamma_i} \leq (d_{GC,M_f})^{\gamma_{max}}, \quad \forall M_i \in MG. \tag{9}$$

Hence, by letting the GC transmit with power $(d_{GC,M_f})^{\gamma_{max}}$, we reach every member $M_i \in MG$. This is equivalent to constructing a multicast routing tree R where every member is connected to the GC with one hop. The power required to deliver a message to all members of MG according to R, cannot be less than the minimum total power obtained by R^*. Hence, the energy expenditure to deliver a message to all $M_i \in MG$, via R^* cannot be greater than the energy expenditure to deliver the same message to all $M_i \in MG$ via R,

$$E_{MG}^{R^*} \leq E_{MG}^{R} \leq \left(d_{GC,M_f}\right)^{\gamma_{max}} T_{trans}. \tag{10}$$

■

In figure 4(b), the GC transmits with power $(d_{GC,M_f})^{\gamma_{max}}$, thus being able to reach every member with one hop. In figure 4(c) we show a generic optimal multicast routing tree R^* for the same network as in figure 4(b). Since R^* is optimal it holds that $E_{MG}^{R^*} \leq \left(d_{GC,M_f}\right)^{\gamma_{max}} T_{trans}$.

The second component of the bound in (7), is the number of update messages sent by the GC for deleting a member from the multicast group. While the number of messages grows logarithmically with the group size N, and N is not a design parameter, we can calculate the tree degree α^* that minimizes the number of update messages.

$$\begin{aligned} \frac{d}{d\alpha}\left(\alpha \log_\alpha N - 1\right) &= 0 \\ \Rightarrow \alpha^* &= e. \end{aligned} \tag{11}$$

The degree of the tree has to be an integer number and hence, the lowest upper bound for E_{Ave} is achieved when $\alpha = 3$. The lowest upper bound for the average key update energy, independent of the network topology and distribution of member leaves is:

$$E_{Ave} \leq (\alpha^* \log_{\alpha^*} N)\left(d_{GC,M_f}\right)^{\gamma_{max}} T_{trans} = (3 \log_3 N)\left(d_{GC,M_f}\right)^{\gamma_{max}} T_{trans}. \tag{12}$$

Note that if we optimize the tree degree α, to minimize the number of rekey messages when both joins and leaves are taken into account and assuming that leaves are as likely to occur as joins, it can be shown that $\alpha^* = 4$ [30]. Also, if we consider key trees using pseudo-random functions as in [4], the optimal tree degree α that minimizes the number of rekey messages is $\alpha^* = 2$. The analysis presented in this section holds for both pseudo-random function trees as in [4], and joint consideration of

joins and leaves as in [30], with the upper bound in (12) adjusted according to the optimal tree degree in each case. The optimal tree degree in (12) can be adjusted to any assumed model of joins and leaves, or any other key tree structure.

We now examine how we can reduce E_{Ave} by exploiting the "power proximity" of the multicast receiver nodes.

V. Impact of "Power Proximity" on the Energy Efficiency of Key Management

E_{Ave} is directly related to the individual energies $\tilde{E}_{M_i}$, for updating keys after each member leaves the multicast group. In (2), we expressed $\tilde{E}_{M_i}$, as the sum of the energies $E^R_{SG_i}$, required to deliver key updates to sub-groups SG_i, via the routing tree R. The routing tree R is optimized for distributing the multicast application data to group members and hence, is not a design parameter. The sub-groups SG_i are determined by how we place the members to the leafs of the key distribution tree, i.e. the way that we choose to assign common KEK's to members. To reduce $E^R_{SG_i}$, we need to group members in the key tree in such a way that less energy is required to deliver to them key updates via R. To do so, we introduce the property of "power proximity." "Power proximity" is similar to the physical proximity, with transmission power used as a metric instead of Euclidean distance. A formal definition is given below.

Definition: "Power proximity"– Given nodes (i, j, k) we say that the node i is in "power proximity" to node j compared to node k, if $P(d_{i,j}) < P(d_{j,k})$, where $P(d_{a,b})$ denotes the transmission power required for establishing communication[6] between nodes a, b.

Given the definition of "power proximity," we show how we can incorporate it in the key tree design in both the cases of a homogeneous and heterogeneous medium.

A. Network deployed in a homogeneous medium

In a homogeneous medium, the transmission power for communication between nodes i, j is a monotonically increasing function of the distance $d_{i,j}$. Under the assumption that routing is optimally selected to minimize the total transmission power, nodes in physical proximity have overlapping routing paths [28]. Intuitively, nodes that are physically close will have common links in the path from the GC towards them and hence, should also share common keys in order to receive the same key updates and reduce the energy expenditure of the key distribution.

To illustrate the need for designing a physical proximity based key distribution, we consider the ad hoc network in figure 5(a), which is deployed in a homogeneous medium. Note that $E_{GC,M_2} > E_{GC,M_1}$ since $d_{GC,M_2} > d_{GC,M_1}$. The routing tree shown in figure 5(a) is optimal in total transmit power. In the key tree of figure 5(c), denoted as Tree A, we randomly place the four members of the multicast group in the leaves of the key tree, independent of the network topology as in wired networks. The second row of Table I shows the average key update energy for Tree A, denoted as E^A_{Ave}, and computed based on (1) by assuming that it is equally likely ($p(M_i) = \frac{1}{N}$) for each member to leave the multicast group.

Assume now that the members are grouped according to their physical proximity. Then, M_1 is grouped with M_4, and M_2 with M_3, resulting in the physical proximity based key tree of figure 5(d), denoted as Tree B. The third row of Table I, shows

[6]Note that although we do not directly consider the routing tree as a design parameter, the idea of "power proximity" is inherent in energy-aware routing[28]. By letting the weights of each link to indicate the amount of power required to maintain the link connectivity, we can construct a minimum spanning routing (MST) tree based on "power proximity." In fact, the MST of this type uses the criterion of the definition of "power proximity."

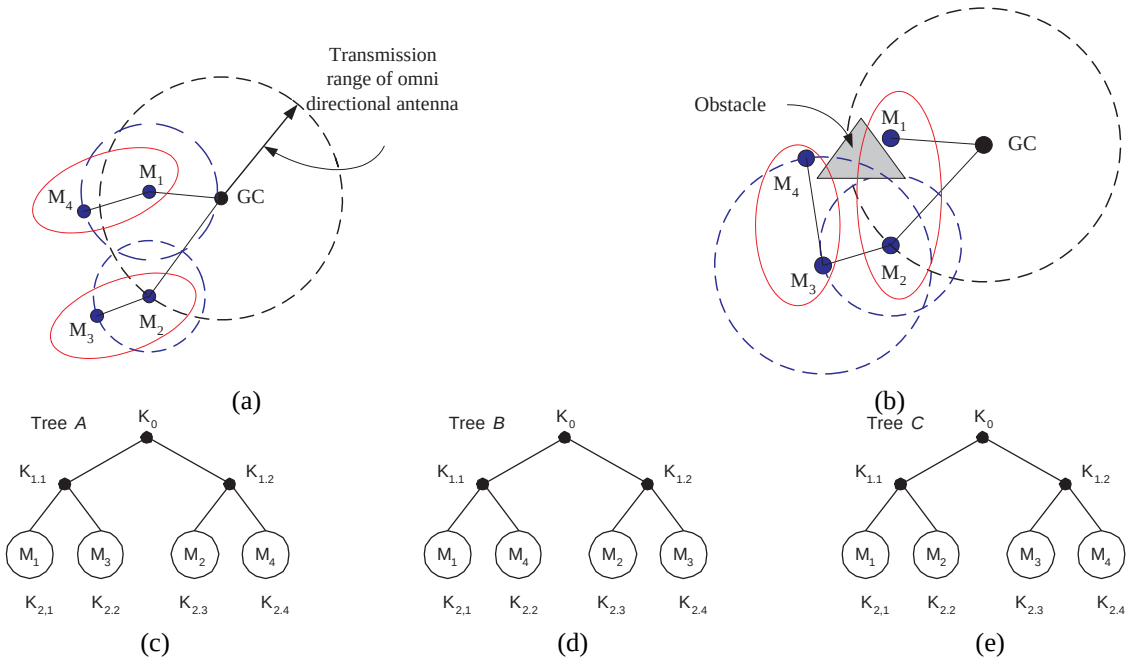


Fig. 5. An ad hoc network and the corresponding routing tree with the minimum total transmission power, deployed in (a) a homogeneous medium, (b) a heterogeneous medium, (c) a random key distribution tree, Tree A, (d) a key distribution tree based on physical proximity, Tree B, (e) a key distribution tree based on "power proximity," Tree C.

the average key update energy for Tree B, denoted as E^B_{Ave}, and computed based on (1) by assuming that it is equally likely ($p(M_i) = \frac{1}{N}$) for each member to leave the multicast group. The energy saved by performing a rekey operation with the physical proximity based key Tree B over the random key Tree A for the network of figure 5(a) is computed as:

$$\begin{aligned} E^A_{Ave} - E^B_{Ave} &= \frac{2}{4}\left(E^R_{\{M_2,M_4\}} + E^R_{\{M_1,M_3\}} - E^R_{\{M_1,M_4\}} - E^R_{\{M_2,M_3\}}\right) \\ &= \frac{2}{4}\left(E_{GC\to M_2} - E_{GC\to M_1}\right) > 0, \end{aligned} \tag{13}$$

where $E_{A\to B}$ denotes the energy required for transmission of a key from node A to node B. The saved energy in (13) is positive since for a homogeneous medium (constant γ) and $d_{GC,M_2} > d_{GC,M_1}$, it is implied $E_{GC\to M_2} > E_{GC\to M_1}$.

B. Network deployed in a heterogeneous medium

We now consider the case of an ad hoc network deployed in a heterogeneous medium, where the attenuation factor γ varies significantly over space. Under heterogeneous path loss, physical proximity in not a monotonic property of "power proximity." Closely located nodes do not necessarily receive messages via overlapping routing paths. Hence, node location information alone is not sufficient for constructing an energy-efficient key tree.

Method	**Average key update energy**
Random tree	$E^A_{Ave} = \frac{1}{4}\left(\sum_i E^R_{\{M_i\}} + 2E^R_{\{M_2,M_4\}} + 2E^R_{\{M_1,M_3\}}\right)$
Physical proximity	$E^B_{Ave} = \frac{1}{4}\left(\sum_i E^R_{\{M_i\}} + 2E^R_{\{M_1,M_4\}} + 2E^R_{\{M_2,M_3\}}\right)$
"Power proximity"	$E^C_{Ave} = \frac{1}{4}\left(\sum_i E^R_{\{M_i\}} + 2E^R_{\{M_1,M_2\}} + 2E^R_{\{M_3,M_4\}}\right)$

TABLE I

COMPARISON OF E_{Ave} FOR THE KEY TREES OF FIGURE 5(C), (D), (E). E_{Ave} IS COMPUTED BASED ON EQ. (1) FOR $p(M_i) = \frac{1}{4}, i = 1 \ldots 4$

To illustrate the above observation, we consider the ad hoc network shown in figure 5(b), in which nodes have the same locations as in figure 5(a). However, there exists a physical obstacle between nodes M_1 and M_4. Thus, the attenuation factor for signal transmission between M_1 and M_4 is significantly higher than the obstacle-free network regions, and in the optimal routing tree in total transmission power, M_4 is connected to the network through M_3.

We now show that in an environment with variable path loss, we are able to construct an energy-efficient key tree by correlating nodes according to their "power proximity," rather than physical proximity. We may acquire such information either by using path loss information in addition to the node location [1], [7], [17], [21], or by measuring the required transmission power for communication between pairs of nodes. Members that are closely located in terms of power are grouped together (placed adjacently to the key tree).

For the network in figure 5(b), we construct the key distribution tree in figure 5(e) denoted as Tree C. We place members adjacent to the key tree according to their "power proximity." M_1 is grouped with M_2, and M_3 with M_4 in order to minimize the total communication power variance of clusters of two members. The last row of Table I, shows the average key update energy for Tree C, denoted as E^C_{Ave}, and computed based on (1) by assuming that it is equally likely ($p(M_i) = \frac{1}{N}$) for each member to leave the multicast group. The energy saved for performing a rekey operation by incorporating location as well as the path loss information instead of location alone is computed as the energy gain due to use of Tree C over Tree B :

$$\begin{aligned} E^B_{Ave} - E^C_{Ave} &= \frac{2}{4}\left(E^R_{\{M_1,M_4\}} + E^R_{\{M_2,M_3\}} - E^R_{\{M_1,M_2\}} - E^R_{\{M_3,M_4\}}\right) \\ &= \frac{2}{4}\left(E_{M_2 \rightarrow M_3}\right) > 0. \end{aligned} \tag{14}$$

Based on our analysis in Sections V-A and V-B we make the following conclusions:

Remark 1: When the medium is homogeneous, we can reduce the energy expenditure for key distribution by assigning common keys to members within physical proximity.

Remark 2: When the medium is heterogeneous medium, we need to employ "power proximity" to generate an energy-efficient key tree hierarchy.

Based on remarks 1 and 2, we develop our key distribution algorithms for the homogeneous and heterogeneous cases.

VI. Incorporating Physical Proximity into Key Trees for a Homogeneous Medium

In Section IV.2, we showed that if nodes in physical proximity are placed adjacently in the key distribution tree, we save significant amount of energy resources when the medium is homogeneous. In this Section, we present an algorithm for the homogeneous medium that exploit physical proximity to generate a energy-efficient key tree. In order to systematically construct a key tree hierarchy, we first cluster nodes based on their location. The clustering is used to form a hierarchy. We translate the physical clustering of the nodes into a key tree hierarchy, thus obtaining an energy-efficient key distribution tree. The task of developing an energy-efficient key distribution scheme is reduced to the task of identifying (a) a physical proximity based clustering mechanism, and (b) building a cluster hierarchy that utilizes the physical proximity based clustering. We discuss both tasks in the following sections.

A. Physical proximity based clustering for energy-efficient key distribution

For the homogeneous medium, we assume that only the node location information is available. Hence, any clustering technique needs to be model-free while taking the location into account. We also note that for the homogeneous case, the Euclidean distance between the nodes is a natural metric for identifying and grouping neighbor nodes. Certainly some other distance metric such as the Minkowsky metric [9] can be used as well, but the monotonicity of the power to the distance in the case of constant γ, makes the Euclidean distance a very attractive metric, since it leads to low complexity algorithms.

A candidate clustering technique should: (a) require only location information as an input, (b) identify the physical network clusters with high success and, (c) generate clusters of equal size so that the cluster hierarchy is mapped to a balanced key tree hierarchy (to maintain constant bandwidth overhead under key updates).

Problem formulation

Let the coordinates of node i be $x_i = (x_{i_1}, x_{i_2})$. The squared Euclidean distance between two nodes i and i' is equal to:

$$d^2_{i,i'} = \sum_{j=1}^{2} \left(x_{ij} - x_{i'j} \right)^2 = \left\| x_i - x_{i'} \right\|^2 . \tag{15}$$

If C denotes an assignment of the nodes of the network into α clusters, the dissimilarity function expressing the total inter-cluster dissimilarity $W(C)$ is:

$$W(C) = \sum_{k=1}^{\alpha} \sum_{C(i)=k} \left\| x_i - m_k \right\|^2 , \tag{16}$$

where $C(i) = k$ denotes the assignment of the i^{th} point to the k^{th} cluster, and m_k denotes the mean (centroid) of cluster k. Inter-cluster dissimilarity refers to the dissimilarity between the nodes of the same cluster. We want to compute the optimal cluster configuration C^* that minimizes (16), subject to the constraint that the sizes of the resulting clusters are equal. This can be expressed as:

$$C^* = \arg\min_{C} \sum_{k=1}^{\alpha} \sum_{C(i)=k} \left\| x_i - m_k \right\|^2 , \quad \ni \quad |C(i)| = |C(j)|, \quad \forall i, j. \tag{17}$$

Note that this formulation provides an optimal way to create α sub-clusters from one cluster. This location based clustering has to be iteratively applied to generate the desired cluster hierarchy.

Solution approach

If we relax the constraint $|C(i)| = |C(j)|, \quad \forall i, j$, in (17), and allow clusters of different sizes, the solution to the optimization problem in (17), can be efficiently approximated by any mean square based clustering algorithm that uses Euclidean metric. The K-means [9] algorithm uses squared Euclidean distance as a dissimilarity measure to cluster different objects. It also generates clusters by minimizing the total cluster variance (minimum square error approach). Note that K-means may result in a sub-optimal local minimum solution depending on the initial selection of clusters, and hence, the best solution out of several random initial cluster assignments should be adopted [9]. However, K-means is easily implemented and hence, is an ideal solution for computationally limited devices. Algorithmic details on solving (17) without any constraint on the cluster size are given in [9].

In order to satisfy the equal cluster size constraint, posed in (17), we need a refinement algorithm (RA) that balances the cluster sizes. According to (17), the RA should result in balanced clusters with the lowest total inter-cluster dissimilarity. In

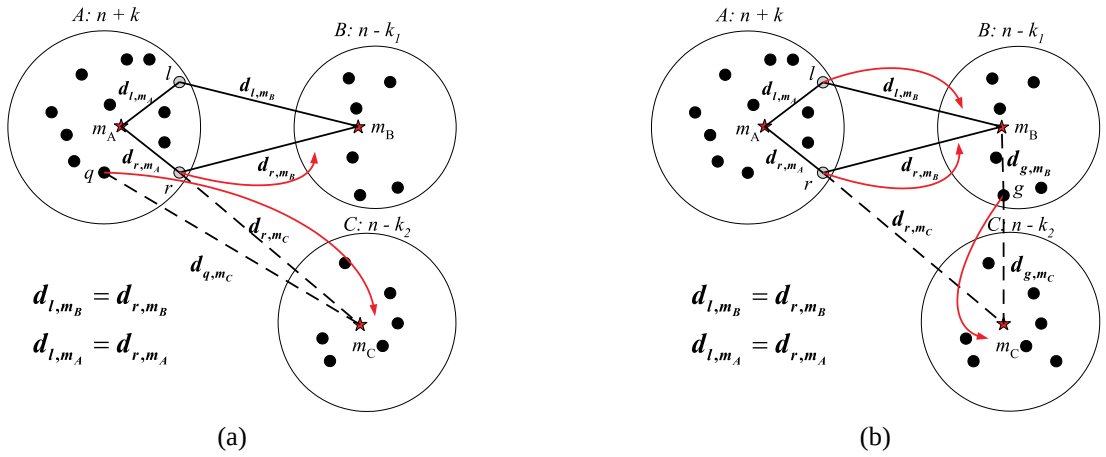


Fig. 6. Sub-optimality of the refinement algorithm. Three un-balanced clusters A, B, C with $|A| = n + k$, $|B| = n - k_1$, $|C| = n - k_2$ and $k = k_1 + k_2$, (a) moving r to B and q to C, results in a sub-optimal solution, (b) moving l, r to B and g to C, results in a better solution than moving l to B and r to C.

the binary tree case, given two clusters A, B with $|A| > |B|$, the refinement algorithm moves objects $i_1, i_2, \ldots, i_k \in A$, with $k = \left\lfloor \frac{|A|-|B|}{2} \right\rfloor$, from cluster A to cluster B, such that the inter-cluster dissimilarity after the refinement is minimally increased. We choose the objects $i_1, i_2, \ldots, i_k \in A$ such that:

$$i_j = \arg\min_{i \in A} [d^2_{i,m_B} - d^2_{i,m_A}], \quad j = 1 : \left\lfloor \frac{|A| - |B|}{2} \right\rfloor, \tag{18}$$

where m_A and m_B are the centroids of clusters A, B.

B. On the optimality of the refinement algorithm

The refinement algorithm [7] balances the clusters sizes obtained from the application of the clustering algorithm. When we balance two clusters, we move an object from cluster A to cluster B so that we minimally increase the total inter-cluster dissimilarity. For the binary case, this greedy approach leads to an optimal solution for the constrained optimization problem in (17). However, if the number of clusters is greater than two, the refinement algorithm in (18) need not lead to balanced clusters of minimum total dissimilarity. We illustrate these points with the example given below.

Consider the clusters A, B, C shown in figure 6, with $|A| = n+k$, $|B| = n-k_1$, $|C| = n-k_2$ and $k = k_1+k_2$. We specialize to the case where $k = 2$, $k_1 = 1$ and $k_2 = 1$. According to the refinement algorithm, we must move two objects from A to B, C (one to B, one to C). We initially find the object $i^* \in A$ such that:

$$i^* = \arg\min_{i \in A} [d^2_{i,m_X} - d^2_{i,m_A}], \quad X = \{B, C\}. \tag{19}$$

For figure 6(a) there are two optimal objects, $i^* = \{l, r\}$ that can be moved from set A since $d_{l,m_A} = d_{r,m_A}$ and $d_{l,m_B} = d_{r,m_B}$. Assume that object r is moved from A to B. Note that object r minimally increases the cluster dissimilarity, out of all objects of A that could be possibly moved to cluster C. Hence, if any other object (q for example) is moved from A to C to balance C, the total cluster dissimilarity will be higher compared to the case where l is moved to B and r is moved to C.

Finding the two objects from cluster A to be moved to clusters B and C respectively so that the total dissimilarity is minimized

[7] The pseudo-code of the algorithm is presented in figure 7(b).

requires exhaustive search through all possible combinations of object movements. In our example, two points out of the $(n+2)$ points in set A need to be moved. There are $(n+2)(n+1)$ possible combinations to be inspected. In the general case where $k_1, k_2, \ldots, k_s$ objects need to be moved from cluster A that has $n+k$ initial points, to clusters $B_1, B_2, \ldots, B_s$ which contain $n-k_1, n-k_2, \cdots n-k_s$ points respectively, where $k = \sum_{i=1}^{s} k_i$, the number of possible combination is:

$$\binom{n+k}{k_1}\binom{n+k-k_1}{k_2}\cdots\binom{n+k_s}{k_s}. \tag{20}$$

A careful consideration shows that when the number of clusters involved is more than two, identification and moving of a set of objects to different clusters need to consider *all* clusters simultaneously, not just the cluster with extra objects. In other words, simply moving the extra objects with the highest dissimilarity from the bigger clusters to the smaller ones need not lead to optimality. We illustrate it now.

Consider the figure 6(b). Set $k = 2$, $k_1 = 1$, and $k_2 = 1$. Hence, to construct balanced clusters, two objects need to be removed from the cluster A and the size of the clusters B and C should increase by one. We need to move objects with minimal increase in dissimilarity. Assume that moving node r or l from A to B increases minimally the total cluster dissimilarity. Also assume that the node g in cluster B has the lowest dissimilarity with respect to cluster C. Then, moving both objects l, r to cluster B and then moving object g from cluster B to cluster C will result in lower total cluster dissimilarity than simply moving l to B and r to C.

From this example, we note that examining and maintaining a list of points and their dissimilarities for the every cluster and every point is important. Hence, when there are more than two clusters, the complexity of finding the globally optimal solution for the optimization problem in (17) requires inspection in each cluster and is even higher than the one expressed in (20). *Therefore, due to the complexity of finding the optimal solution, it is preferable to adopt the sub-optimal solution for balancing the clusters, provided by the refinement algorithm.*

C. A sub-optimal energy-efficient key distribution scheme based on physical proximity

We now develop an algorithm that maps the physical proximity based clustering into a hierarchical key tree structure. We need to construct a key tree of fixed degree α. Initially, the global cluster is divided into α sub-clusters using K-means. Then, we employ the RA algorithm that balances the cluster sizes by moving the most dissimilar objects to appropriate clusters. The RA leads to the construction of a balanced key tree when $N = \alpha^n, n \in \mathbb{Z}$ and allows us to construct a structure as close to the balanced as possible when $N \neq \alpha^n$. Each cluster is subsequently divided into α new ones, until clusters of at most α members are created (after $\log_\alpha N$ splits). Since our algorithm uses only location information as input, we call it *Location-Aware Key Distribution Algorithm* (LocKeD). The figure 7 presents the pseudo code for LocKeD. We now describe the notational and algorithmic details of figure (7).

Let $\mathcal{P}$ denote the set containing all the two-dimensional points (objects) corresponding to the location of the nodes. Let $C = \{C(1), C(2), \ldots, C(\alpha)\}$ denote a partition of $\mathcal{P}$ into α subsets (clusters), i.e. $\bigcup_i C(i) = \mathcal{P}$. Initially, all objects belong to the global cluster $\mathcal{P}$. The function *AssignKey()* assigns a common key to every subset (cluster) of its argument set. For example, *AssignKey*($\mathcal{P}$) will assign the SEK to every member of the global cluster $\mathcal{P}$.

Location-Aware Key Distribution - LocKeD

$C = \{\mathcal{P}\}$
AssignKey(C)
index=1
while $index < \lceil \log_\alpha(N) \rceil$
  $C\_temp = \{\emptyset\}$
  $thres = \lceil \frac{N}{\alpha^{index}} \rceil$
  for $i = 1 : |C|$
    R=*Kmeans*$(C(i), \alpha)$
    R=*Refine*$(R, thres)$
    AssignKey(R)
    $C\_temp = C\_temp \bigcup R$
    index++
  end for
$C = C\_temp$
end while

(a)

Refinement Algorithm - RA

$C_{Low} = \{C(i) \in C : |C(i)| < thres\}$
$C_{High} = \{C(i) \in C : |C(i)| > thres\}$
repeat until $C_{High} = \emptyset$
  find $x^* \in A,\ A \in C_{High}$

$$x^* = \arg\min_{x \in A}[diss(x, m_B) - diss(x, m_A)],$$
$$\forall\ x \in A,\ \forall\ A \in C_{High},\ \forall\ B \in C_{Low}$$

  move x^* to cluster B

  $C_{Low} = \{C(i) \in C : |C(i)| < thres\}$
  $C_{High} = \{C(i) \in C : |C(i)| > thres\}$

end repeat

(b)

Fig. 7. Pseudo code for (a) the location-aware key distribution algorithm (LocKeD) and (b) the Refinement Algorithm (RA). Repeated application of *Kmeans()* function followed by the Refinement Algorithm *Refine()* for balancing the clustering sizes, generates the cluster hierarchy. Function *AssignKey()* assigns a common key to every member of its argument set.

The *index* variable counts the number of steps required until the termination of the algorithm. The *thres* variable holds the number of members each cluster ought to contain at level $l = index$ of the key tree construction. The root of the tree is at level $l = 0$. The $Kmeans(C(i), \alpha)$ function divides the set $C(i)$ into α clusters and returns the cluster configuration to variable R. The *Refine(R,thres)* function balances the clusters sizes of clusters in R according to the *thres* variable. Then, *AssignKey()* is applied to assign different keys to every cluster in R. The process is repeated until $\lceil \log_\alpha N \rceil$ steps have been completed.

Computational Complexity of LocKeD: In terms of algorithmic complexity, the LocKeD algorithm iteratively applies K-means up to N times in the worst case (generation of a binary tree). K-means has algorithmic complexity of $O(N)$ [9]. Hence, the complexity of the LocKeD is $O(N^2)$. We now present an example of how to construct a key tree based on LocKeD algorithm.

Application of LocKeD on a sample network deployed in a homogeneous medium

Consider the network in figure 8(a), deployed in a homogeneous medium with an attenuation factor $\gamma = 2$. We will construct a location-aware key distribution tree of degree $\alpha = 2$ with nodes $\{2, 3, \ldots, 9\}$ being the members $\{M_2, M_3, \ldots, M_9\}$ of the multicast group, respectively. Initially, all members belong to the global cluster $\mathcal{P}$.

Note that the GC does not participate in the clustering. The key tree is constructed by executing the following steps:

Step 1: Assign the SEK K_0 to every member of the global cluster $\mathcal{P}$.

Step 2: Create two clusters by splitting the global cluster. The two clusters that yield minimal total cluster dissimilarity are:

$$C_1 = \{M_2, M_3, M_4, M_6, M_8, M_9\}, C_2 = \{M_5, M_7\}.$$

Since we seek to construct a balanced key tree, apply the refinement algorithm to balance the clusters sizes. Move M_2 and M_6 to cluster C_2. Assign two different KEKs to members of clusters C_1 and C_2. Members of C_1 are assigned KEK $K_{1.1}$ and members of C_2 are assigned KEK $K_{1.2}$.

Step 3: Create clusters of two members, by splitting the clusters of four members. The four created clusters are:

$$C_3 = \{M_2, M_6\},\ C_4 = \{M_3, M_4\}, C_5 = \{M_8, M_9\},\ C_6 = \{M_5, M_7\}.$$

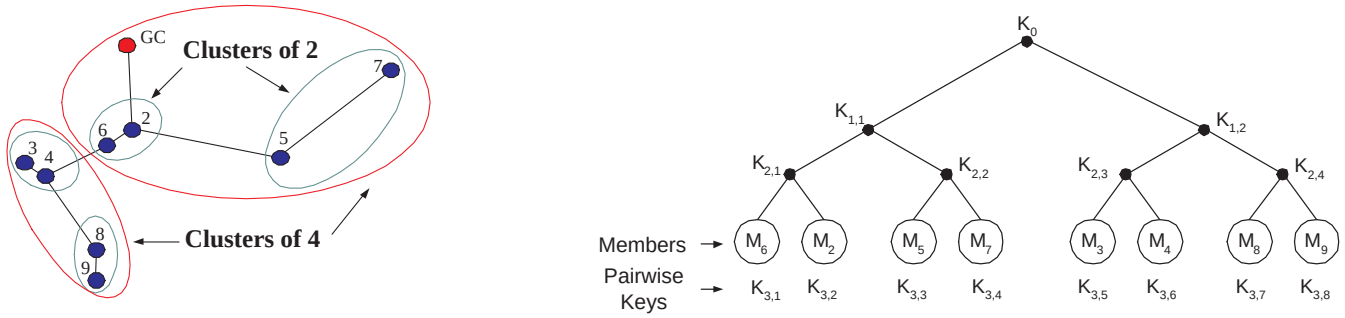


Fig. 8. (a) An ad hoc network deployed in a homogeneous medium and the corresponding routing paths. Iterative application of the location based clustering and the resulting cluster hierarchy. (b) The key distribution tree resulting from the application of LocKeD.

Again, different KEKs are assigned to members of clusters C_3-C_6. Members of C_3 are assigned KEK $K_{2.1}$, members of C_4 are assigned KEK $K_{2.2}$, members of C_5 are assigned KEK $K_{2.3}$ and members of C_6 are assigned KEK $K_{2.4}$. At this point we have completed the $\lceil \log_\alpha N \rceil$ steps required by LocKeD and the algorithm terminates.

The resulting hierarchical key tree constructed using LocKeD is shown in figure 8(b). We now study the heterogeneous case.

VII. "Power Proximity" Based Key Distribution for a Heterogeneous Medium

A. Characteristics of the heterogeneous medium

When the wireless medium is heterogeneous, the signal attenuation factor is not unique for the network deployment area. An office building is a typical example of an environment where the attenuation factor varies even across very short distances. The signal attenuation for nodes located in different floors is significantly higher than for nodes located in the same floor [19]. The heterogeneity of the medium creates additional challenges in performing energy-efficient key distribution.

Constraint 1: As shown by the example in Section V-B, in a heterogeneous medium, physical proximity between two nodes does not equate to less transmission power needed for communication between those nodes. Hence, in a heterogeneous medium, Euclidean distance is not a suitable metric to express the dissimilarity between the objects (nodes) that need to be clustered.

We showed in Section V-B that direct use of "power proximity" leads to energy-efficient key distribution, when the medium is heterogeneous. Hence, we propose the use of transmission power as the dissimilarity measure for performing the clustering. We define the dissimilarity between two nodes i,j for the heterogeneous medium as:

$$diss(i,j) = P(d_{i,j}). \tag{21}$$

We note that the K-means algorithm used as critical component of the balanced clustering algorithm in the homogeneous medium case, cannot use any arbitrary dissimilarity measure but Euclidean distance, since it utilizes the notion of mean vectors. Hence, for heterogeneous case, we cannot use K-means as a component of our "power proximity" based algorithm.

Constraint 2: In a heterogeneous environment, different network regions need to be described using different path loss models [19]. Depending upon node location and the medium, a different function shall be used to calculate the dissimilarity between two nodes. Hence, any solution approach should allow the simultaneous use of arbitrary and multiple dissimilarity measures representing different network regions.

Our task of developing an energy-efficient key distribution algorithm for the heterogeneous medium is reduced to, (a) identifying a "power proximity" based algorithm to identify clusters with high success, (b) generating a cluster hierarchy that will be mapped into a key tree hierarchy. We now present techniques suitable for "power proximity" based clustering.

B. "Power proximity" based clustering for energy-efficient key distribution

As noted above, the K-means clustering cannot be part of the balanced clustering algorithm to be developed in heterogeneous case. A candidate solution needs to be able to handle arbitrary dissimilarity metrics. We use two different approaches for clustering in the heterogeneous case. The first approach employs a clustering technique known as K-medoids [10], that minimizes the total inter-cluster dissimilarity. Hence, K-medoids exploits "power proximity" in the optimal way. In order to create a key tree of fixed degree, K-medoids clusters have to be balanced and the algorithm has to be iteratively applied for every level of the tree. Though optimal in cluster quality, the complexity of K-medoids is prohibitive for large networks and therefore, we adopt a sub-optimal solution based on randomized sampling.

Our second approach is based on Divisible Hierarchical Clustering (DHC) [10]. DHC minimizes the average inter-cluster dissimilarity within each cluster, while directly generating a cluster hierarchy. The hierarchical feature of DHC, along with the ability to use any arbitrary dissimilarity measure, makes this solution attractive for creating a key tree hierarchy. In order to produce a balanced key tree, we need to ensure that at each stage the clusters are balanced. We describe both approaches in detail in the following sections.

1) Minimizing the total inter-cluster dissimilarity: We now describe the first formulation that satisfies the constraint 1 and constraint 2 and exploits the "power proximity".

Problem formulation

We need a clustering technique that, (a) uses power $P(d_{i,j})$ as a dissimilarity measure to generate clusters and group together the most "similar" nodes, (b) generates clusters of equal size. While using an arbitrary dissimilarity metric other than the Euclidean distance, it is not feasible to define the centroid of a cluster. Hence, the total cluster dissimilarity cannot be computed with respect to the centroid, as in (17).

To overcome this limitation, we identify the most centrally located object within a cluster as a cluster representative, called medoid. We then compute the inter-cluster dissimilarity by adding the dissimilarities of each object of a cluster with its medoid. In order to construct α clusters $C = \{C_1, C_2, \ldots, C_\alpha\}$, we select α medoids, $M = \{m_1, m_2, \ldots m_\alpha\}$, one for each cluster. For each choice of medoids, an object $i,\ i \notin M$, is assigned to the cluster $C_j,\ j = 1 \ldots \alpha$, if:

$$diss(i, m_j) \leq diss(i, m_r), \forall r = 1, \ldots, \alpha, \tag{22}$$

where m_x denotes the medoid of the cluster C_x. Using the medoids as reference points, the total inter-cluster dissimilarity is computed as:

$$W(C) = \sum_{i=1}^{N} \min_{m_j=1,..,\alpha} diss\,(i, m_j)\,. \tag{23}$$

We want to find the optimal medoids $M^* = \{m_1^*, m_2^*, \ldots, m_\alpha^*\}$ that minimize (23), subject to the constraint that the sizes of

the resulting clusters are equal. Therefore:

$$C^* = \arg \min_{\{m_r\},C} \sum_{i=1}^{N} \min_{m_j=1,..,\alpha} diss\left(i, m_j\right), \quad \ni \quad |C(i)| = |C(j)|, \quad \forall i, j. \tag{24}$$

Solution approach

Let us first consider solving the optimization problem in (24), without the constraint $|C(i)| = |C(j)|, \quad \forall i, j$, imposed on the cluster sizes. Kaufman and Rousseeuw [10] proposed a solution that minimizes the total inter-cluster dissimilarity in (24). Their K-medoids method called *Partitioning Around Medoids* (PAM) [10], repeats successive exchanges between medoids and ordinary objects until the medoid set resulting in the smallest cluster dissimilarity is found. While PAM is optimal, it scales poorly with group size and hence, Kaufman and Rousseeuw proposed a scalable sub-optimal K-medoids method called *Clustering LARge Applications* (CLARA) [10], based on randomized sampling.

K-medoids algorithm however, leads to clusters of unequal sizes [10]. Hence, in order to satisfy the constraint posed in (24), we need the refinement algorithm (RA) of figure 7(b) to balance the cluster sizes. The criterion by which successive objects $i_1, i_2, \ldots, i_k \in A$ with $k = \left\lfloor \frac{|A|-|B|}{2} \right\rfloor$, are moved from cluster A to cluster B with $|A| > |B|$, is modified to reflect the dissimilarity metric used in the heterogeneous medium. We choose the objects $i_1, i_2, \ldots, i_k \in A$ such that:

$$i_j = \arg \min_{i \in A} [P(d_{i,m_B}) - P(d_{i,m_A})], \quad j = 1 : \left\lfloor \frac{|A| - |B|}{2} \right\rfloor, \tag{25}$$

where m_A and m_B refer to the medoids of clusters A and B, respectively. By minimally increasing $W(C)$ at each object re-assignment, we achieve the optimal solution for the constrained optimization problem in (24) in the case of binary trees. Following similar analysis as in the case of the homogeneous medium, when more than two clusters need to be balanced (degree of the tree > 2), we can show that while the refinement algorithm presented is only sub-optimal, the complexity of the optimal solution is prohibitively high as the number of nodes grows. Hence, we adopt the sub-optimal solution.

We now present the second approach that is based on minimizing the average dissimilarity within each cluster.

2) Minimizing the average inter-cluster dissimilarity: We now describe a clustering technique that minimizes the average dissimilarity within a cluster, instead of the total cluster dissimilarity. The advantage of using the average over the total dissimilarity is that we do not comparably compute the dissimilarity with respect to a single cluster object as in K-medoids method. Furthermore, we can provide a solution inspired by divisible hierarchical clustering (DHC) that inherently provides a cluster hierarchy. We first introduce the following quantities.

Cluster Diameter: Diameter $diam$ of a cluster A is defined as the highest dissimilarity between two objects within the cluster given by:

$$diam(A) := \max_{i,j \in A} P(d_{i,j}) \tag{26}$$

Average inter-cluster dissimilarity of an object: For an object i in cluster A, the average inter-cluster dissimilarity, denoted by $a(i)$ is defined as the average of the dissimilarities of i with all other objects in A as:

$$a\left(i\right) = \frac{1}{|A| - 1} \sum_{j \in A, j \neq i} P(d_{i,j}). \tag{27}$$

Average intra-cluster dissimilarity of an object: For an object $i,\ i \in A$, and given a cluster $B, i \in B$, the average intra-cluster dissimilarity, denoted $w(i, B)$ is given by:

$$w\left(i, B\right) = \frac{1}{|B|} \sum_{j \in B} P(d_{i,j}). \tag{28}$$

Description of the algorithm: Initially, all objects are moved to a global cluster A. The object $i^* \in A$

$$i^* = \arg\max_{i \in A} a(i), \tag{29}$$

with the highest dissimilarity is moved to a new cluster B. Quantities $a(i)$ and $w(i, B)$ are then recomputed for all $i \in A$. An object $m \in A$ is moved from cluster A to cluster B, only if m is more similar to cluster B,

$$m = \arg\max_{i \in A} \left[a(m) - w(m, B)\right], \qquad a(m) - w(m, B) \geq 0. \tag{30}$$

The moving of objects is repeated until no object in A is more similar to B, i.e. $a(i) \leq w(i, B),\ \forall i \in A$. At this stage clusters A and B have been finalized as parent clusters. In the next step, the cluster with the biggest diameter is further split into two new clusters using the previous steps. Though this procedure generates a binary hierarchical tree, we can modify it to generate a tree of arbitrary degree α. To construct one level of a hierarchical tree of degree α, the following steps are followed:

1) Perform (α-1) successive splits of the global cluster, leading to α total clusters.
2) Set α clusters as parents on the first level of the hierarchy.
3) Repeat steps 1-2 until every child cluster contains α objects.

In DHC, the two clusters created by a split of one cluster have minimum average inter-cluster dissimilarity. However, this minimization need not necessarily lead to clusters of equal sizes. Hence, the RA algorithm needs to be applied to balance the cluster sizes.

After Step 1, we utilize the RA algorithm developed in figure 7(b). According to figure 7(b), an object $x^* \in A$ is moved from a cluster $A \in C_{High}$ with more objects than the threshold $thres$, to a cluster $B \in C_{Low}$ with less objects than $thres$, if:

$$x^* = \arg\min_{x \in A} [diss(x, m_A) - diss(x, m_B)], \quad \forall\ x \in A,\ \forall\ A \in C_{High},\ \forall\ B \in C_{Low}. \tag{31}$$

However, no notion of a mean point or representative cluster object exists if average inter-cluster dissimilarity is used. We therefore move the object $x^* \in A$, from a cluster $A \in C_{High}$ with more objects than the threshold $thres$, to a cluster $B \in C_{Low}$ with less objects than $thres$, if:

$$x^* = \arg\max_{x \in A} \left[a(x) - w(x, B)\right], \quad \forall\ x \in A,\ \forall\ A \in C_{High},\ \forall\ B \in C_{Low}. \tag{32}$$

The algorithmic details of DHC are given in figure 9(b) and will be explained in detail in the following section.

C. Two power-aware key distribution schemes based on "power proximity"

We now show how the two "power proximity" based clustering algorithms presented earlier can be used to develop hierarchical key trees for the heterogeneous medium. We construct a key tree of fixed degree α. Since our algorithms use transmission power

Power-Aware Key Distribution (PAKeD)

K-medoids Clustering (PAKeD-KM)

$C = \{\mathcal{P}\}$
AssignKey(C)
$diss = Power(C, \gamma)$

if $|C| > MaxSize$
  Clust=CLARA
end if

index=1

while $index < \lceil \log_\alpha(N) \rceil$
  $C\_temp = \{\emptyset\}$,
  $thres = \lceil \frac{N}{\alpha^{index}} \rceil$

  for $i = 1 : |C|$
    R=*Divide*$(C(i), \alpha, Clust)$
    R=*Refine*$(R, thres)$
    AssignKey(R),
    $C\_temp = C\_temp \bigcup R$
    index++
  end for

$C = C\_temp$
end while

(a)

Divisible Hierarchical Clustering (PAKeD-DH)

$C = \{\mathcal{P}\}, \quad index = 1$
AssignKey(C)
$diss = Power(C, \gamma)$
while $index < \lceil \log_\alpha(N) \rceil$
$thres = \lceil \frac{N}{\alpha^{index}} \rceil$
for j=1:$|C|$
  $C\_temp = \emptyset$
  while $|C\_temp| \leq \alpha$
    $A := \max_{J \in C\_temp} diam(J), \quad B = \emptyset$
    $Diss\_A = Ave\_Diss(A, diss)$
    $i^* = \arg\max_{i \in A} Diss\_A$
    $A = A - \{i^*\}, \quad B = \{i^*\}$
    If $|A| = 1$ *stop*
    else repeat
      $Diss\_A = Ave\_Diss(A, diss)$
      $Diss\_B = Ave\_Diss(B, diss)$
      $max\_diss = \max_{i \in A}(Diss\_A - Diss\_B)$
      $m = \arg\max_{i \in A}(Diss\_A - Diss\_B)$
      if $max\_diss > 0$
        $B = B \bigcup \{m\}, \;\; A = A - \{m\}$
      else
    end repeat
    $C\_temp = C\_temp \bigcup \{A, B\}$
  end while
end for
C =*Refine*$(C\_temp, thres)$
index=index++
end while

(b)

Fig. 9. Pseudo code for the Power-Aware Key Distribution algorithm (PAKeD), (a) when clustering is performed using K-medoids (PAKeD-KM), and (b) when we directly generate a hierarchical key tree using divisible hierarchical clustering (PAKeD-DH).

as the dissimilarity metric, we refer to them as Power-Aware Key Distribution (PAKeD). We name the algorithm that makes use of K-Medoids as Power-Aware Key Distribution–K-Medoids (PAKeD-KM). We name the algorithm that makes use of Divisible Hierarchy as Power-Aware Key Distribution–Divisible Hierarchy (PAKeD-DH).

1) Power Aware Key Distribution based on K-Medoids (PAKeD-KM): In figure 9(a), we present the pseudo code for the Power-Aware Key Distribution - K-Medoids (PAKeD-KM) algorithm, that utilizes a "power proximity" clustering algorithm based on K-medoids [9], [10]. We now describe the notational and algorithmic details of PAKeD-KM given in figure 9(a).

Initially, all members (objects) belong to the global cluster $\mathcal{P}$. The *AssignKey*($\mathcal{P}$) function assigns the SEK to all members of the group. The *Power(C,*γ*)* function computes the dissimilarities between members according to the path loss information, and stores them in matrix *diss*.

Choice between CLARA and PAM in PAKeD-KM: Depending on the network size, we employ either PAM or CLARA as a method for dividing the global cluster into sub clusters. CLARA algorithm is chosen over PAM if the network size N is bigger than a threshold $MaxSize$. Studies in [10] showed that PAM becomes inefficient for data sets bigger than 100 objects. The choice is stored in variable $Clust$ with the default being PAM. If $MaxSize > 100$, $Clust$ is set to CLARA.

The $Divide(C(i), \alpha, Clust)$ function partitions $C(i)$ to α clusters according to $Clust$ method, and returns the created clusters to variable R. The *Refine(R,thres)* function *balances* the cluster sizes and the *AssignKey(R)* assigns keys to the clusters in R. After $\lceil \log_{\alpha}(N) \rceil$ steps the algorithm terminates.

Computational Complexity of the PAKeD-KM: The algorithmic complexity of PAM is $\mathcal{O}((1+\beta)N^2)$, where β is the number of medoid swaps required in K-medoids algorithm [10]. CLARA reduces the complexity to $\mathcal{O}(N)$ through randomized sampling [10]. Either PAM or CLARA are iteratively applied at most N times (binary key tree) and hence the algorithmic complexity of PAKeD-KM is $\mathcal{O}((1+\beta)N^3)$, or $\mathcal{O}(N^2)$, depending on whether PAM or CLARA is applied.

2) Power Aware Key Distribution based on Divisible Hierarchy (PAKeD-DH): In figure 9(b), we present the pseudo code for the Power-Aware Key Distribution - Divisible Hierarchical clustering (PAKeD-DH) algorithm, that utilizes a "power proximity" clustering algorithm, based on divisible hierarchical clustering [9], [10].

For the PAKeD-DH algorithm, the basic steps are as described in Section VII-B. Initially, the SEK is assigned to all members of the multicast group with *AssignKey*($\mathcal{P}$) and the dissimilarity matrix $diss$ is computed as in PAKeD-KM algorithm. The cluster with the highest diameter is split into sub clusters A, B. In order to create the cluster B, the average dissimilarities $a(i)$ and $w(i, B)$ are stored in $Diss\_A$, $Diss\_B$, respectively. Cluster splitting is repeated until α clusters have been created. Then, the α clusters are balanced with the refinement algorithm *Refine()*, according to the threshold $thres$, and a key is assigned to each cluster. This process is repeated for every level of the tree hierarchy. The algorithm terminates after $\lceil \log_{\alpha}(N) \rceil$ steps.

Computational Complexity of PAKeD-DH: The complexity of divisible hierarchical clustering is $\mathcal{O}(N^3)$ [10]. Divisible hierarchical clustering outputs a cluster hierarchy and need not be iteratively applied as in the case of K-means or K-medoids clustering. Hence, the complexity of PAKeD-DH is $\mathcal{O}(N^3)$.

We now present the performance evaluation of the algorithms we developed.

VIII. Performance Evaluation

A. Simulation setup

Simulation studies were performed on randomly generated network topologies confined in a specific region. After the node placement, the location of the nodes is assumed to be fixed. Following the network generation, the routing paths were established. Though any suitable algorithm can be applied to provide the routing paths, we used the Broadcast Incremental Power (BIP) algorithm presented in [28]. The resulting routing tree is used to calculate the consumed energy for transmitting key updates to each of the group members.

Since there is no algorithm to provide the minimum energy solution for the key distribution tree construction, we performed an exhaustive search for small group sizes $N = 8, N = 16$. For larger group sizes, $N = 32, 64, 128, 256$, we generated for each network instance, 10,000 different random key tree structures. Out of the randomly generated structures, we selected the key trees that resulted in the minimum and maximum E_{Ave}, and compared them with the performance of our algorithms. We also computed the mean performance of the 10,000 random key trees and compared that with LocKeD, PAKeD-KM, and PAKeD-DH algorithms. We repeated the same comparison for 100 different network topologies and averaged the results.

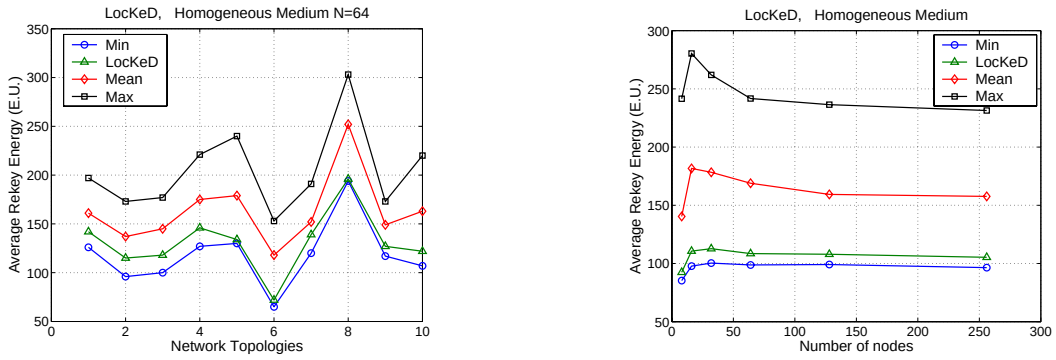


Fig. 10. Experiment 1- Homogeneous Medium: (a) Application of LocKeD in a free space area for 10 different network topologies of 64 nodes plus the GC, compared with the energy expenditure of the minimum, maximum and mean performing tree out of the 10,000 examined tree structures. (b) Application of LocKeD in a free space area for different network sizes averaged over 100 network topologies.

B. Experiment 1: Network deployed in a homogeneous medium - Free space case

In our first experiment we assumed that the network was deployed in an open space area. The medium was assumed to be homogeneous and the attenuation factor was set to $\gamma = 2$. We confined the network in a 10x10 region and evaluated the performance of LocKeD.

Initially, we measured the performance of the LocKeD algorithm, for 10 different network topologies of 64 nodes plus the GC. In figure 10(a), we compare the LocKeD with the minimum and maximum performing tree as well as the average, out of the 10,000 randomly generated tree structures. We can observe the key tree with the best performance spends 1.3%-16.7% less energy than the LocKeD. However, if location is neglected, the LocKeD spends 24.4%-54.2% less, compared to the key tree with the maximum average key update energy and 14.2%-36.4% less, compared to the mean of all randomly generated trees, for the 10 different networks in figure 10(a).

In figure 10(b), we show the results of LocKeD as a function of the multicast group size N, averaged over 100 network topologies for each N. The LocKeD spends on average 9% more energy for re-keying, compared to the key tree with the minimum E_{Ave}. LocKeD spends on average 57% less energy for re-keying, compared to the key tree with the maximum E_{Ave}, and 32% less, compared to the mean of all randomly generated trees.

We note that although we increased the multicast group size, we kept the deployment region constant. Hence, the node density increased. When the network density increases over the same network region, the total transmission energy to reach all members decreases, since more efficient paths are chosen by the routing algorithm (less high-power transmissions are required, since nodes are in closer proximity). One would expect that for larger network sizes we need to transmit more messages for rekeying and, hence, we would spend more energy. However, due to the node density increase, we observe that we need less energy to perform rekeying, as messages are sent through more power-efficient paths. Therefore, in figure 10(b), the average key update energy required for rekeying decreases as the network size increases, with the gains leveling off after 64 nodes. Also, we note that E_{Ave} increases as the network size increases from 8 to 16, since for such network sizes the node density is low and the increase in transmitted messages is not compensated by more efficient route discovery.

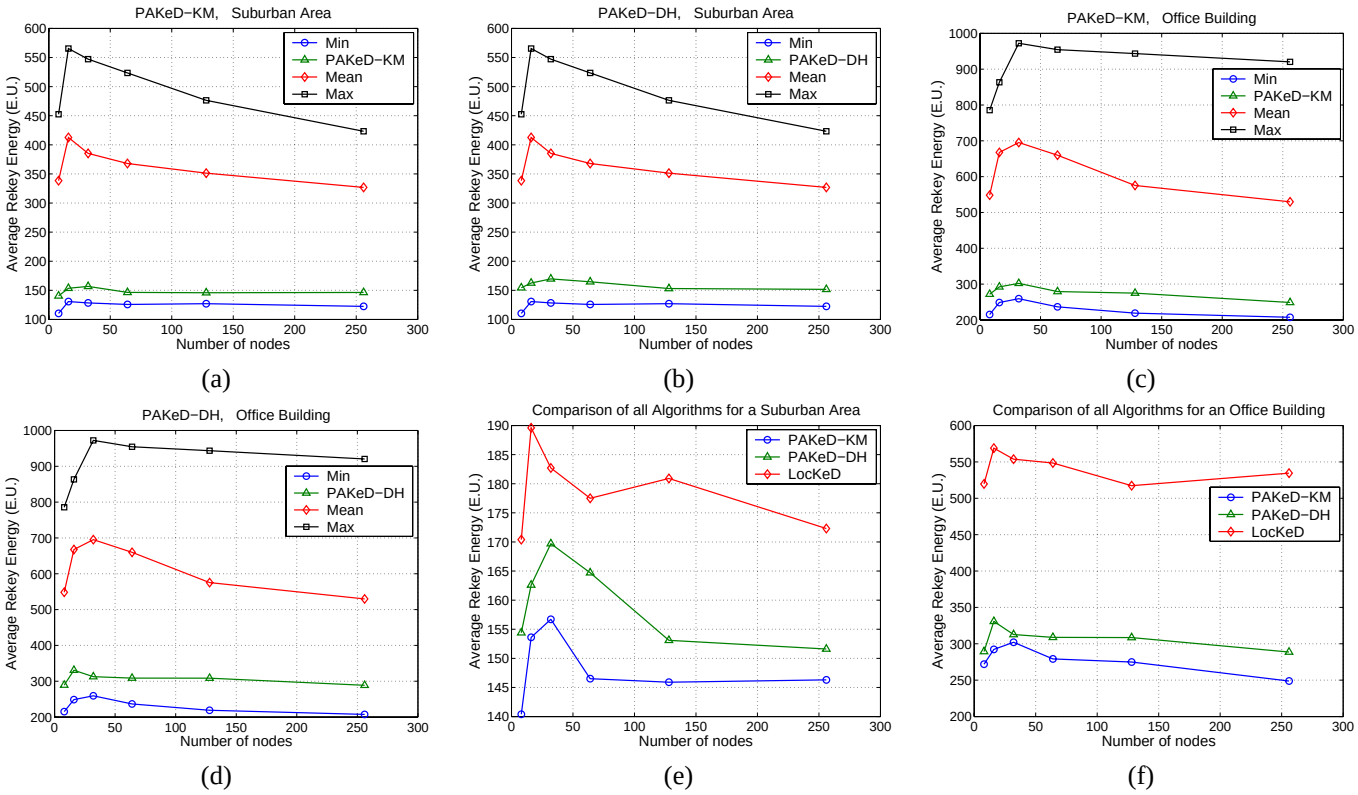


Fig. 11. Experiments 2 and 3- Heterogeneous Medium: Application of PAKeD-X, X$\in \{KM, DH\}$, for different network sizes, compared with the energy expenditure of the minimum, maximum and mean performing tree out of the 10,000 randomly generated tree structures when, (a) the network is deployed in a suburban area and PAKeD-KM is applied (b) the network is deployed in a suburban area and PAKeD-DH is applied, (c) the network is deployed in an office building and PAKeD-KM is applied, (d) the network is deployed in an office building and PAKeD-DH is applied. (e) Comparison of PAKeD-KM, PAKeD-DH, LocKeD for a network deployed in a, (e) suburban area, (f) office building.

C. Experiment 2: Network deployed in a heterogeneous Medium - Suburban area

In our second experiment, we evaluated the performance of the PAKeD for a slowly varying heterogeneous medium. We considered a suburban area where the attenuation factor γ is not constant throughout the network deployment region. However, we assumed that it changes slowly across space. We confined our network in a 12x12 square region and assumed the path loss model in (33) of table II, for computing the required power for establishing a communication link between nodes i, j. According to the model in (33), nodes that are located near-by are assumed to be in line-of-sight (LOS, $\gamma = 2$), while nodes located farther away experience higher attenuation ($\gamma = 3$ or $\gamma = 4$), due to interference caused by physical obstacles [19]. In figure 11(a), we compare the PAKeD-KM with the minimum, maximum, and average performing tree out of the 10,000 randomly generated trees, as a function of the multicast group size N, for networks deployed in a suburban area. We observe that the PAKeD-KM spends on average 19% more energy for rekeying, compared to the key tree with the minimum average key update energy. The PAKeD-KM spends on average 70% less energy for rekeying, compared to the key tree with the maximum average update, and 59%, compared to the mean of all randomly generated trees. The PAKeD-DH is 26% worse than the minimum E_{Ave} key tree, while being 67% better than the worst performing tree and 56% better than the mean of all the randomly generated trees.

We note that the key tree with the maximum E_{Ave} spends *almost three times* as much energy as the tree constructed with PAKeD-KM. This is due to the fact that sending messages in a heterogeneous environment requires more energy than in a homogeneous one, and using an inefficient key distribution scheme can lead to great waste of energy resources.

In figure 11(e), we compare the PAKeD-KM with the PAKeD-DH and the LocKeD, for networks of different group sizes

$$P(d_{i,j}) = \begin{cases} d_{i,j}^2, & d_{i,j} < 4 \\ d_{i,j}^3, & 4 \leq d_{i,j} < 8 \\ d_{i,j}^4, & 8 \leq d_{i,j} \leq 12 \end{cases} \quad (33)$$

(A)

$PL(d_{i,j}) = PL(d_0) + 10\gamma \log\left(\frac{d_{i,j}}{d_0}\right)$		
Location	γ	σ **(dB)**
Same floor	2.76	12.9
Through one floor	4.19	5.1
Through two floors	5.04	6.5
Through three floors	5.22	6.7

(B)

TABLE II

(A) PATH LOSS MODEL FOR AN OFFICE BUILDING. PL DENOTES THE SIGNAL ATTENUATION IN DB AND d_0 IS THE CLOSE-IN REFERENCE DISTANCE SET TO 1M. THE TABLE SHOWS TYPICAL VALUES OF γ AND ITS STANDARD DEVIATION σ. THE STANDARD DEVIATION INDICATES THE AMOUNT OF VARIABILITY OF γ IN EACH REGION, MEASURED IN DIFFERENT LOCATIONS INSIDE THE BUILDING [19]. (B) PATH LOSS MODEL FOR A SUBURBAN AREA.

deployed in a suburban area. We observe that the PAKeD-KM and the PAKeD-DH have very similar performance. PAKeD-KM performs 6.8% on average better than the PAKeD-DH, since the PAKeD-KM minimizes the total cluster dissimilarity, while PAKeD-DH considers the average dissimilarity.

We can also observe that the LocKeD performs the worst among the three algorithms as expected. The LocKeD performs 20% worse than the PAKeD-KM and 12% worse than the PAKeD-DH. By comparing figures 11(d) and 11(f), we note that the performance of the LocKeD is still significantly better than the average and worst case random key trees. The LocKeD performance is explained by the path loss model assumed. According to (33), physical proximity that is used as a clustering criterion in the LocKeD, is strongly correlated with "power proximity" that is used as a clustering criterion in PAKeD.

D. Experiment 3: Network deployed in a heterogeneous medium - indoor environment

In our third simulation experiment, we evaluated the performance of the PAKeD algorithm for a rapidly changing heterogeneous medium. We assumed that the network is deployed in a four-story office building emulated by a 12x12x12 cubic space and the path loss between nodes i, j is described in table II(A). In figure 11(d), we compare the PAKeD-KM with the minimum and maximum performing tree as well as the average out of the 10,000 randomly generated trees, for different multicast group sizes in an indoor environment. In figure 11(e), we compare the PAKeD-DH with the minimum, maximum, and average performing tree, in the same indoor environment and for the same networks. We observe that the PAKeD-KM spends on average 17% more energy for rekeying, compared to the key tree with the minimum average key update energy. The PAKeD-KM spends on average 72% less energy for rekeying, compared to the key tree with the maximum average update, and 56% less when compared to the mean of all randomly generated trees. The PAKeD-DH is 27% worse than the minimum average key update energy key tree, 66% better than the worst performing tree and 50% better than the mean of all the randomly generated trees.

In figure 11(f), we compare the PAKeD-KM with the PAKeD-DH and the LocKeD for different group sizes deployed in an office building. The PAKeD-KM performs 12.4% on average, better than the PAKeD-DH. As expected, the LocKeD performs poorly in the indoor environment by spending 96% and 74% more energy for rekeying than the PAKeD-KM and the PAKeD-DH respectively. In the indoor environment physical proximity is not increasing monotonically with "power proximity" and clustering based on location fails to give an energy-efficient key distribution tree.

IX. CONCLUSIONS

We studied the problem of constructing energy-efficient key distribution schemes for securing multicast communications in wireless ad hoc networks. We noted that while the balanced key trees are bandwidth-efficient solutions in average sense, the key

trees did not consider energy as a design parameter. In order to incorporate the energy-efficiency into the key trees, we introduced a new performance evaluation metric called average key update energy. We characterized this metric in terms of the network topology, the properties of the propagation medium and the degree of the key tree. Thus, the design of energy-efficient key distribution problem is a cross-layer design problem that incorporates physical and network layer parameters. We then noted that depending on whether the propagation medium is homogeneous or heterogeneous, we could formulate problems with different cost functions and computational complexities for the cross-layer design problem.

When the medium is homogeneous, we showed that the node location can be used to design energy-efficient balanced key trees. For heterogeneous medium, we developed algorithms that consider "power proximity" in the design of balanced key trees. We also presented the complexities of our algorithms and showed the pitfalls of trying to find a globally optimal solution. We presented simulation results and applied our algorithms to obstacle-free open space, suburban and indoor environments and showed significant energy savings, using our algorithms that demonstrate the advantage in using a cross-layer design approach.

ACKNOWLEDGEMENTS

We thank anonymous reviewers for their valuable comments.

REFERENCES

[1] P. Bahl and V. Padmanabhan, RADAR: An In-Building RF-Based User Location and Tracking System, in: *Proceedings of the IEEE Conference on Computer Communications* (INFOCOM 2000) (March 2000) pp. 565-574.

[2] M. Baugher, R. Canetti, L. Dondeti, F. Lindholm, MSEC Group Key Management Architecture, IETF draft available at *http://www.ietf.org/internet-drafts/draft-ietf-msec-gkmarch-07.txt.*

[3] M. Cagalj, J.P. Hubaux and C. Enz, Minimum-Energy Broadcast In All Wireless Networks: NP-Completeness and Distribution Issues, in: *Proceedings* 8^{th} *ACM Annual International Conference on Mobile Computing and Networking* (MOBICOM 2002) (September 2002) pp. 172-182.

[4] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor and B. Pinkas, Multicast Security: A Taxonomy and Some Efficient Constructions, in: *Proceedings of the IEEE Conference on Computer Communications* (INFOCOM 1999) (March 1999) pp. 708-716.

[5] S. Capkun, L. Buttyan, and J. Hubaux Self-Organized Public-Key Management for Mobile Ad-Hoc Networks, IEEE Transactions on Mobile Computing (TMC) (Jan.-March 2002) vol. 2, no. 1 52-64.

[6] D.W. Carman G.H. Cirincione and B.J. Matt, Energy-Efficient and Low-Latency Key Management for Sensor Networks, in: Proceedings. 23^{rd} Army Science Conference (December 2002).

[7] G. Dommety and R. Jain, Potential Networking Applications of Global Positioning Systems (GPS), Technical report TR-24, The Ohio State University (1996).

[8] Z. Haas and L. Zhou, Securing Ad hoc Networks, IEEE Network Magazine, (Nov.-Dec. 1999), vol. 13, no. 6, 24-30.

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

[10] L. Kaufman and P.J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, (John Wiley & Sons, 1990).

[11] T. Kaya, G. Lin, G. Noubir, A. Yilmaz, Secure Multicast Groups on Ad Hoc Networks, in: *Proceedings First ACM Workshop on Security of Ad Hoc and Sensor Networks* (SASN 2003) (October 2003) pp. 94-102.

[12] L. Lazos and R. Poovendran, Secure Broadcast in Energy-Aware Wireless Sensor Networks, IEEE International Symposium on Advances in Wireless Communications (ISWC'02), Victoria BC, Canada, September 2002.

[13] L. Lazos and R. Poovendran, Energy-Aware Secure Multicast Communication in Ad-hoc Networks Using Geographic Location Information, in: *Proceedings IEEE International Conference on Acoustics Speech and Signal Processing* (April 2003) vol. 4, pp. 201-204.

[14] L. Lazos and R. Poovendran, Cross-Layer Design for Energy-Efficient Secure Multicast Communications in Wireless Ad Hoc Networks, in: *Proceedings IEEE International Conference in Communications* (ICC 2004) (June 2004).

[15] S. Lee, W. Su, J. Hsu, M. Gerla and R. Bagrodia, A Performance Comparison Study of Ad hoc Wireless Multicast Protocols, in: *Proceedings IEEE Conference on Computer Communications* (INFOCOM 2000) (March 2000) pp. 565-574.

[16] W. Liang, Constructing Minimum-Energy Broadcast Trees in Wireless Ad Hoc Networks, in: *Proceedings* 3^{th} *ACM International Symposium on Mobile Ad Hoc Networking and Computing* (MobiHoc 2002) (June 2002) pp. 112-122.

[17] D. Niculescu and B. Nath, DV Based Positioning in Ad hoc Networks, Journal of Telecommunication Systems, (January 2003) vol. 22, iss. 1-4, 267-280.

[18] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, Energy-Aware Wireless Microsensor Networks, IEEE Signal Processing Magazine, (March 2002) vol. 19, iss. 2, 40-50.

[19] T. Rappaport, *Wireless Communications: Principles & Practice*, (Prentice Hall, New Jersey 1996).

[20] A. Perrig, D. Song and D. Tygar, ELK, a new Protocol for Efficient Large-Group Key Distribution, in: *Proceedings IEEE Security and Privacy Symposium 2001* (May 2001) pp. 247-262.

[21] N.B. Priyantha, A. Chakraborty and H. Balakrishnan, The Cricket Location-Support System, in: *Proceedings Sixth Annual ACM International Conference on Mobile Computing and Networking* (MOBICOM 2000) (August 2000) pp. 32-43.

[22] N. Sastry, U. Shankar and D. Wagner, Secure Verification of Location Claims, in: *Proceedings ACM Workshop on Wireless Security* (WISE 2003) (September 2003) pp. 1-10.

[23] J. Snocyink, S. Suri and G. Varghese, A Lower Bound for Multicast Key Distribution, in: *Proceedings IEEE Conference on Computer Communications* (INFOCOM 2001) (March 2001) vol. 1, pp. 442-432.

[24] F. Stajano and R. Anderson, The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks, in: *Proceedings Seventh International Workshop on Security Protocols Security Protocols* (April 1999) pp. 172-194.

[25] Y. Sun, W. Trappe and R. Liu, An Efficient Key Management Scheme for Secure Wireless Multicast, in: *Proceedings IEEE International Conference on Communications* (ICC 2002) (May 2002) vol. 2, pp. 1236 -1240.

[26] Y. Sun, W. Trappe, and R. Liu, Topology-aware Key Management Schemes for Wireless Multicast, in: *Proceedings Global Communications* (GLOBECOM 2003) (Dec. 2003) vol. 3, pp. 1471-1475.

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

[28] J.E. Wieselthier, G.D. Nguyen and A. Ephremides, On the Construction of Energy Efficient Broadcast and Multicast Trees in Wireless Networks, in: *Proceedings IEEE Conference on Computer Communications* (INFOCOM 2000) (March 2000) pp. 586-594.

[29] Wireless Integrated Network Sensors, University of California, Los Angeles. Available: http://www.janet.ucla.edu/WINS.

[30] C.K. Wong, M. Gouda and S. Lam, Secure Group Communications Using Key Graphs, IEEE/ACM Transactions on Networking, (Feb. 2000) vol. 8, no.1, pp. 16-31.

[31] Y. Xue, B. Li, and K. Nahrstedt, A Scalable Location Management Scheme in Mobile Ad-Hoc Networks, in: *Proceedings IEEE Conference on Local Computer Networks* (LCN 2001) (November 2001) pp. 102-111.

[32] Y. Yang, X. Li and S. Lam, Reliable Group Rekeying: Design and Performance Analysis, in: *Proceedings Special Interest Group on Data Communication* (SIGCOMM 2001) (August 2001) pp. 27-38.

[33] 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).