Mitigating Eavesdropping Attack Using Secure Key Management Scheme in Wireless Mesh Networks

Farah Kandah, Yashaswi Singh
Department of Computer Science, North Dakota State University, Fargo, ND 58105
Email: {Farah.Kandah, Yashaswi.Singh}@ndsu.edu

Weiyi Zhang
AT&T Labs - Research, Middletown, NJ 07748
Email: wzhang@ieee.org

***Abstract*— Wireless mesh network (WMN) is a rapid deployed, self organized and multi-hop wireless networks. However, the wireless and distributed natures of WMN make it subject to various kinds of malicious attacks, which raise a great challenge in securing these networks. Most existing security mechanisms are based on cryptographic keys where a high degree key management services are in demand. In this paper, we present an effective secure key management scheme (SKeMS) which seeks an encryption key assignment such that the induced network is securely key connected and well protected against potential malicious eavesdropping attacks. Compared to previous work, our scheme assigns the available encryption keys among all the nodes in the network. Our simulation results show that our scheme outperforms previous schemes through providing a network that is resistant to malicious eavesdropping attack.**

***Index Terms*— Wireless mesh network, secure key management, adversary, malicious eavesdropping attack**

I. Introduction

A wireless mesh network (WMN) is a multihop wireless network consisting of a large number of wireless nodes, such as mesh gateways (*which are connected with a wired network to the internet*), mesh routers (*which can relay packets through wireless channels*), and mesh end users [1] [2] [3]. The architecture of WMN is shown in Fig. 1. With more attentions on WMNs lately, the security issues become more important and urgent for managing and deploying in such networks [4]. The flexible deployment nature and the lack of fixed infrastructure make WMNs suffer from varieties of security attacks [4] [5], where the existence of such attacks might hold back the potential advantages and wide scale deployment of this promising wireless network technology. Most current security mechanisms (*e.g., encryption and digital signature*) which can be used for WMNs are based on cryptographic keys and thus providing a well designed key management services are in demand [5] [6]. Key management service responsibilities include establishing a trusted secure communication between nodes as well as keep track of bindings between keys [5] [6].

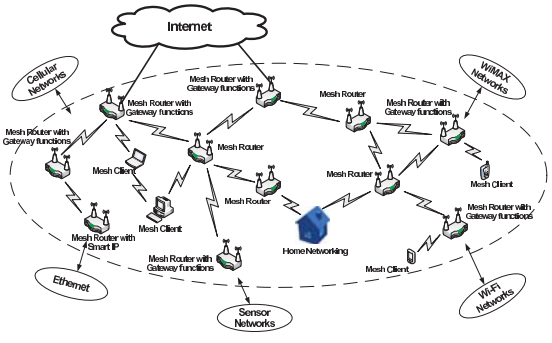


Figure 1. Wireless mesh network architecture

Recent researches have shown that security attacks is holding back the potential advantages and wide-scale deployment of wireless networking technology [4] [5]. Several key management schemes [4] [5] [7] [8] have been proposed for wireless networks and claimed to have high security. However, their weakness such as high computational overhead, storage overhead and vulnerability to some kinds of attacks are undeniable.

Du *et al.* in [7] proposed a key management scheme for heterogenous sensor networks. Each high-end sensor is preloaded with M keys, and each low-end sensor is preloaded with L keys ($M >> L$) in a *pre-distribution phase*, where the keys are randomly picked from a pool of keys (P) without replacement. Followed by the *discovery phase*, which is used to check if neighboring sensors have a shared key, and the *key setup phase*, which is used to find a shared key between any two neighboring sensors when the discovery phase returns that there is no common key between them. *The pool size can affect this proposed scheme*, where with a large pool size and a small K keys randomly selected from P to be stored in each node, a better security can be provided [7]. On the other hand with small pool size, there will be a chance of having more nodes shared common keys in the neighborhood which might harm the network due to various adversary attacks. Moreover, neither all the generated keys in the pool are being used nor the keys in high-end or low-end

doi:10.4304/jcm.7.8.596-605

sensors. Our proposed scheme use the available number of keys (K) to be assigned *among* all the nodes, without generating too many unnecessary keys, and keep the network as secure as possible.

Zhao *et al.* in [4] propose an elliptic curve cryptosystem (ECC)-based self-certified public key cryptosystem. Their scheme supports an efficient authentication and key agreement between a mesh client and a mesh router. Another key management scheme has been proposed in [9], in which the authors designed a key management scheme that includes selective distribution and revocation of keys to sensor nodes as well as node rekeying. In their proposed scheme a key pre-distribution phase required a large pool (P) of keys (*e.g.,* 2^{17} - 2^{20} *keys*), where each sensor's memory is preloaded with a random drawing of K keys out of (P) without replacement. Shared key discovery phase follows the previous phase to establish the topology, where every node discovers its neighbor in its wireless communication range with which it shares keys. *Our proposed scheme differ from this scheme where the network topology is known in advance, and we assign K keys among the nodes, to provide a secured pre-established key distribution*.

A low-computational and scalable key management model for WMNs was proposed in [5], where the authors aim to guarantee a well performed key management service and protection against potential attacks. Another study in [10] considers the problem of designing a key management scheme in a clustered distributed sensor networks, where the probability of node compromise in different deployment regions is known in advance. In their scheme, the network with N nodes is divided into S subgroups, in which, each node within a subgroup is preloaded with a set of keys using the scheme in [9]. Different probability of node compromise values are assigned to different subgroup. Our proposed scheme differs in that, all nodes in the network have the same probability to be compromised.

The rest of this paper is organized as follows. Our system model and motivations are discussed in Section II. We formally define our problem we are going to study in Section III. Our secure key management scheme is presented in Section IV, which is followed by numerical results in Section V. We conclude the paper in Section VI.

II. Models and Motivations

First in this section, we will describe our network model and the adversary model. Then, formally we will discuss our motivations towards this work. Note that, in this work the terms edge and link are interchangeable, the terms mesh router (MR) and mesh node or simply node are interchangeable, also the terms encryption key and key are interchangeable.

A. Network Model

We assume a large WMN consists of a number of mesh routers (MR)s which are stationary and without energy constraints. These MRs provide access to mesh clients and also relay information from one MR to another through wireless multi-hop. All MRs use the same fixed transmission power ($R > 0$). We use a undirected bi-connected graph $G(V; E)$ to model the wireless mesh network where V is the set of n nodes and E is the set of m links in the network. For each pair of nodes $(u; v)$, there exist a undirected edge $e \in E$ if and only if $d(u; v) \leq R_u$, where $d(u; v)$ is the Euclidean distance between u and v, and R_u is the transmission range of node u. Each edge between any pair of nodes $(u; v)$ in G corresponds to a potential wireless link between nodes u and v in the network. Note that, in this work for security purposes, we assume that there is no communication between any two neighboring nodes (*nodes in the transmission range of each other*), unless they shared a common encryption key.

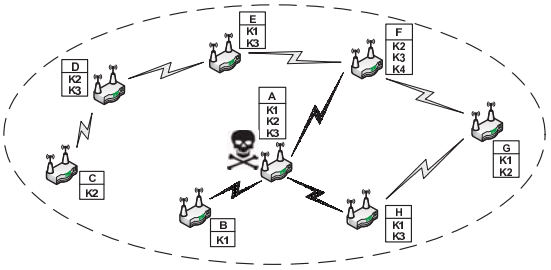


Figure 2. Malicious eavesdropping attack

B. Threat Model

To disturb WMN operations, the adversary may launch arbitrary attacks such as passive eavesdropping, bogus message injection and physical-layer jamming [11]. In this paper we focus on passive eavesdropping attacks in WMNs. Due to the broadcast nature of wireless channels, all nodes that fall in the transmission range of a specific node, say u, can receive its transmitted messages [12]. In this paper we assume that the adversary can compromise an arbitrary number of mesh nodes, through physical capture or software bugs, thus gaining full control of them. Once compromised, the adversary will extract all the security information stored in the compromised nodes as well as the encryption keys preloaded into their memories. The adversary can capture any message that is being sent by any of the compromised node's neighbors. If a message is encrypted using any encryption key that is preloaded to the compromised node, in this case the adversary will be able to decrypt the message and extract its content.

To illustrate the attack, we will use an example in Fig. 2. In this example, 8 MRs are forming a WMN, in which, each MR is preloaded with a number of encryption keys. Note that, each node can communicate with its neighbor if they share at least one encryption key, the links in the figure correspond to the existence of shared keys between different nodes. Let us assume that node A has been compromised, in this case all the one hop neighbors of node A will be monitored by the compromised node (A). Monitored links are noted with black links in the

figure. *When node F sends an encrypted message to node E using key ($K3$), due to the broadcast nature of wireless networks, and since node A lies within node F's transmission range and has the key ($K3$), the adversary at node A will listen to all the encrypted messages that comes out of node F using ($K3$).* The same thing would happen if node H is broadcasting a message to G encrypted with key ($K1$), as long as the compromised node (A) has this key, it will keep spying on all the messages in its neighborhood (*transmission range*) that is encrypted using ($K1$).

C. Motivations

In this section we discuss our motivations towards this work.

- In practice, when a node is compromised by an adversary, all the information stored in that node will be extracted by that adversary, including the set of encryption keys preloaded to that node. We realized that, *the way in which keys are assigned to/among all the nodes in the network could make the network resistant or vulnerable to malicious attacks* such as the one discussed in II-B.
  We observed that previously key management schemes did not consider the effect of sharing the same keys between nodes within a 2-hops neighboring range of a any node, say u. Due to the broadcast nature of wireless channels, when any neighboring node of u, say v, send a message M_k to any other node outside the communication range of node u, if this message is encrypted using key k which is shared between nodes u and v, then the adversary will be able to decrypt the message content using the keys extracted from the compromised node u. This attack has been discussed in II-B.
- Previous works [7] [9] indicate that, to assign keys to nodes in a network, a large pool of keys must exist, from where a set of keys (K) is chosen randomly to be assigned to each node. We realized that, *not all the generated keys in the pool will be used, nor all the keys stored in some nodes*. Moreover, *the ratio between the pool size and the number of keys $|K|$ will affect the network*, where with large ratio, more variety keys can be chosen from the pool for each node. If the process of generating encryption keys for the pool of keys is expensive, this will affect the key assignment scheme, in that the variety in choosing keys from the pool will be small, and that in turn will lead to having the same keys being shared between multiple nodes in the same neighborhood.

In this work, we aim to provide a key assignment scheme between the MRs, by assigning K available encryption keys *among* all nodes in a common neighborhood to be as different as possible such that the malicious eavesdropping attack can be reduced.

III. Problem Statement

In this section, we will formally state the definition of our optimization problem that we are going to study.

*Definition 1 (**Shared encryption key** ($Sk_{u,v}$)):* Given any two neighboring nodes $u, v \in G$, if there is an encryption key $k \in keys(u) \cap keys(v)$, then we can say that there exists a shared encryption key $Sk_{u,v}$ between node u and node v, where $keys(u)$ and $keys(v)$ are the sets of keys which are preloaded to node u and v respectively. □

*Definition 2 (**2-hop compromised nodes** ($2CN_u$)):* Given nodes $u, v, w \in G$, where v is a 1-hop neighbor of u, and w is a 2-hop neighbor of u via v. If node u has been compromised, the *2-hop compromised nodes* of node u ($2CN_u$) is defined as the set of nodes (w), for which node v sends messages encrypted by any key $k \in Sk_{u,v} \cap Sk_{v,w}$. □

*Definition 3 (**Node compromise ability** ($NCA(u)$)):* Given a network G, we define the *node compromise ability* (*NCA*) for a compromised node $u \in G$, as the number of nodes in the set $2CN_u$. □

This is given in Eq. 1.

$$NCA(u) = |2CN_u| \tag{1}$$

*Definition 4 (**Malicious eavesdropping ability** (*MEA*)):* Given a network G with n nodes, where each node has been preloaded with a set of encryption keys. The *malicious eavesdropping ability* in the network is defined as the maximum *NCA* among all nodes in G. □

This is shown in Eq. 2.

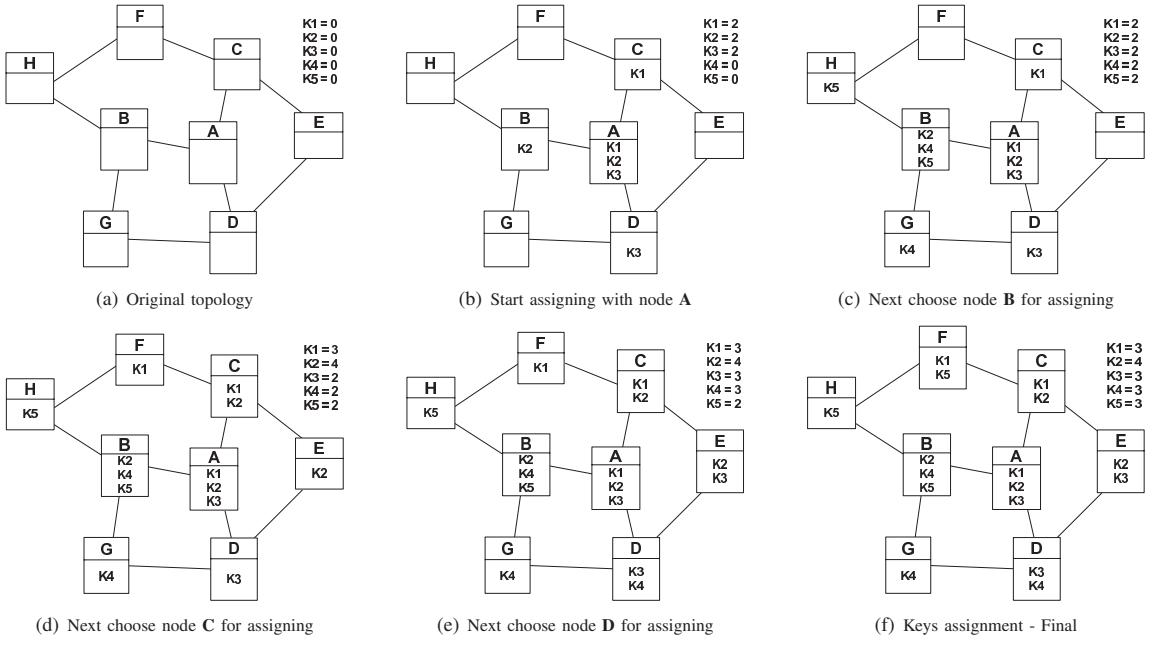
$$MEA = max\{NCA(n)|n \in G\} \tag{2}$$

It is worth nothing that before encryption keys are given to each node, it is impossible to measure the malicious eavesdropping ability of the network. However, because the broadcast nature of wireless mesh networks, if we can distribute the encryption keys to be as different as possible, we could minimize the malicious eavesdropping ability in the network. Note that, different encryption key assignment can induce different corresponding communications, as well as increasing or decreasing the malicious eavesdropping ability. We formalize our secure key management scheme problem in the following:

*Definition 5 (**SKeMS** problem):* Given a network G and a set of encryption keys (K), the **S**ecure **Ke**y **M**anagement **S**cheme (**SKeMS**) seeks a key assignment design such that the *MEA* in the network is minimized using $|K|$ encryption keys. □

IV. A Secure Key Management Scheme

In order to have a secure WMN that is resistant to malicious eavesdropping attacks, we in this work provide a secure key management scheme (SKeMS) that seeks to minimize the malicious eavesdropping ability (*MEA*) in the network. Our proposed solution is listed in Algorithm



(a) Original topology (b) Start assigning with node **A** (c) Next choose node **B** for assigning

(d) Next choose node **C** for assigning (e) Next choose node **D** for assigning (f) Keys assignment - Final

Figure 3. Secure key management scheme (SKeMS) example

1, the notation's description to be used in our scheme is given in Table I.

TABLE I.
NOTATION USED IN OUR SCHEME DESCRIPTION

Notation	Description
u, v, w	Nodes
NIR(u)	u's neighbors that have no common keys with u
K	A set of available encryption keys
k	An encryption key
$keys(u)$	A set of keys in node u

Given a network G and a set of encryption keys K, first in line 1, we initialize $keys$ in each node in G to the empty set. For all nodes in G, find node, say u, that has the highest number of neighbors that do not have common keys with u (Lines 3-4). After choosing the node with the highest *NIR* we start assigning the keys between that node and all its neighbors (Lines 7-16). *The idea in this key assignment design is that we try to assign the keys among the nodes to be as different as possible, while keeping the network securely key connected*. After choosing the node to start with, say node u, we start by taking node u and one of its neighbors, say v, as a pair of nodes and assign a key on both nodes to be used as a shared key for secure communication (Line 5-6).

In lines 7-9, if the chosen pair of nodes u and v has not been assigned any key yet, we will choose the least used key from the set of available keys K and assign it on both nodes, so as to be used as an encryption key for their communications. If node u has previously been assigned some keys, in this case we will choose the least used key from K not been used on any neighboring nodes of node u or node v, so as to make the assignment as diffrent as possible. This idea can be seen clear in the example in Fig. 3. If node u has already assigned some keys, but it is not sharing any of them with node v. In this case we will choose the least used key from the available keys not in w, where node w is a neighbor of node u, which already share a key with node u. Then we will add the key to both u and v nodes. (Lines 10-12) If both chosen nodes have been assigned some keys but there is no shared key between them, in this case we will choose the least used

Algorithm 1 Secure Key Management Scheme (G, K)

1: **for** each node $u \in G$ **do**
2: $keys(u) = \emptyset$;
3: **end for**
4: **for** all nodes in G **do**
5: **for** each node $u \in G$ **do**
6: Find *NIR*(u);
7: Calculate $|$*NIR*(u)$|$;
8: **end for**
9: Choose node $u \in G$ with the highest $|$*NIR*(u)$|$;
10: **for** each node $v \in$ *NIR*(u) **do**
11: //Assign keys between node u and node $v \in$ *NIR*(u) based on the following rules:
12: **if** $keys(u) = \emptyset$ and $keys(v) = \emptyset$ **then**
13: Choose k as the least used key from K;
14: Add k to $keys(u)$ and $keys(v)$;
15: **else if** $keys(u) \neq \emptyset$ and $keys(v) = \emptyset$ **then**
16: Choose k as the least used key from K not in $keys(w)$, where w is a neighbor of u, if applicable, else choose k as the least used key from K;
17: Add k to $keys(u)$ and $keys(v)$;
18: **else if** $keys(u) \neq \emptyset$ and $keys(v) \neq \emptyset$ **then**
19: Choose k as the least used key from K not on w where $w \in NIR(u) \cup NIR(v)$, if applicable, else choose the least used key from K;
20: Add k to $keys(u)$ and $keys(v)$;
21: **end if**
22: **end for**
23: **end for**

key from K not been used on either u *or* v's neighbors, if applicable, else we will choose the least used key from K (Lines 13-15).

We use an example in Fig. 3 to illustrate our algorithm. For simplicity, we assume five keys available to be assigned among 8 nodes. Fig. 3(a) shows the original network topology. According to Algorithm 1, We start with node A, since it has the highest number of neighboring nodes that do not share keys with it. In here we have nodes B, C and node D as node A's neighbors. Let us start with the nodes pair (A, C), we will refer to lines 7-9 in our algorithm. We choose the least used key from K, say key K1, and assigned it to both nodes. Next we will follow the same steps before to assign keys between nodes A, B and A, D. This assignment can be seen in Fig. 3(b).

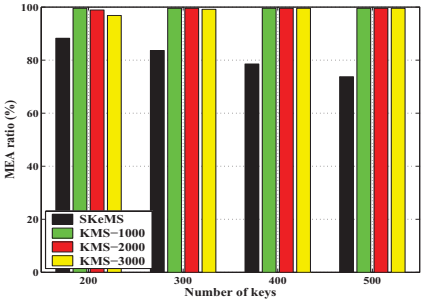
Our next step is shown in Fig. 3(c). We choose node B to continue the key assignment, since node B has the highest *NIR* among all the nodes in the network. It can be seen that node G and H has no shared key among them, but since node B has some keys, we will try to assign a new key between these two nodes to have a different key assignment that can stand among the malicious eavesdropping attacks. In this case we follow our algorithm in lines 10-12 to assign the keys. Since node A which is node B's neighbor is using keys (K1, K2, K3) for encryption, and node D which is node G's neighbor is using key K3, we will choose the least used key from K which is not been used on nodes A and D, in here we choose K4 as the encryption key between node B and node G. We followed the same steps to assign a key between node B and node H, where we choose K5 as the encryption key between them.

Next we start assigning keys on the next node with the highest *NIR* among all the nodes in the network. Here node C and D have the same *NIR*, thus we choose the node which has the most number of neighbors with the smaller number of keys. We choose node C according to the previous rule and start assigning the keys. This step is shown in Fig. 3(d). Following node C, we choose node D according to Algorithm .1. The key assignment for node D is shown in Fig. 3(e). Our final key assignment provided by our proposed SKeMS scheme is shown in Fig. 3(f).

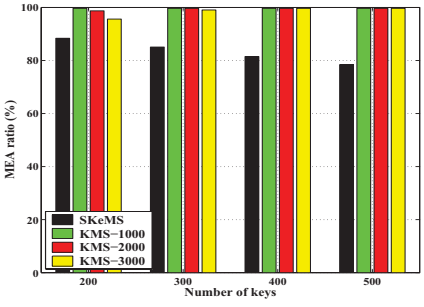
V. Numerical Results

To illustrate the performance of our scheme, we implemented our solution (denoted by **SKeMS** in the figures), and compared it with previous scheme in [7] (denoted by **KMS** in the figures). We considered static WMN with n nodes uniformly distributed in a square playing field. Each node has a fixed transmission range of $250m$. The results shown are the average of 5 test runs for various scenarios.

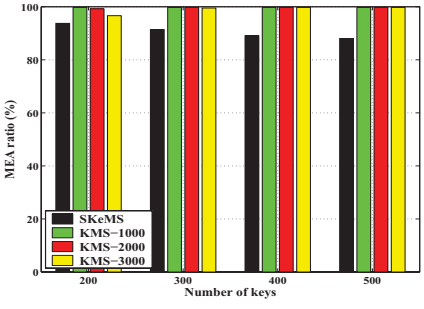
The first metric used for performance evaluation is ***malicious eavesdropping ability ratio*** (denoted as *MEA ratio* in the figures), which is calculated as the neighbor compromise ability (*NCA*) divided by the total number of



(a) 300 nodes in $1000m$x$1000m$



(b) 400 nodes in $1000m$x$1000m$

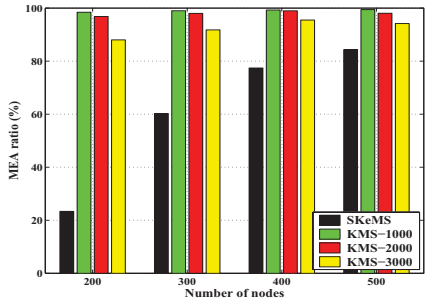


(c) 500 nodes in $1000m$x$1000m$

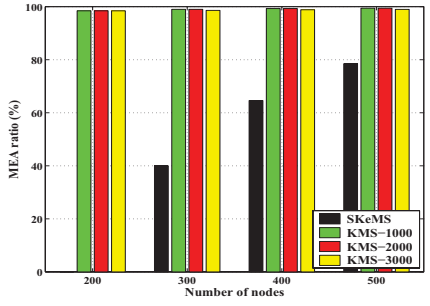
Figure 4. *MEA ratio* with different number of keys

neighboring nodes that are vulnerable to eavesdropping attack (discussed in subsection II-B). *Having smaller MEA ratio indicates that the network is more secured and more resistant against malicious eavesdropping attacks*. In our first tested scenario, we randomly distributed 300, 400 and 500 nodes in a 10×10^5 square meters. To achieve better security for KMS scheme, we provide different pool sizes ranges from (1000−3000) keys. The available number of keys K ranges from (200−500) keys. *Note that, having different pool sizes doesn't affect our SKeMS scheme, since we distribute the set of keys (K) among all the nodes in the network rather than upload each node with the set of keys (K)*. Our first scenario's results are shown in Fig. 4.

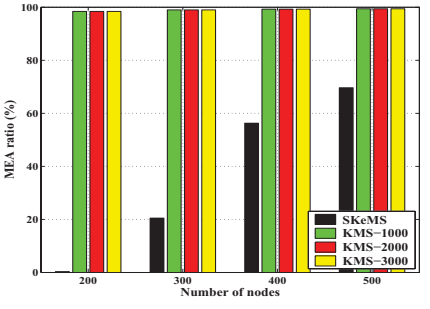
Fig. 4(a) shows the *MEA ratio* versus different number of keys ranges from (200−500) keys. In our proposed scheme, we used available keys to assign them among all nodes in the network. On the other hand, in the KMS, each node will be uploaded with the available number of keys which are chosen randomly from the provided key pool. For the KMS scheme increasing the



(a) 200 Keys in a 1500mx1500m



(b) 300 Keys in a 1500mx1500m



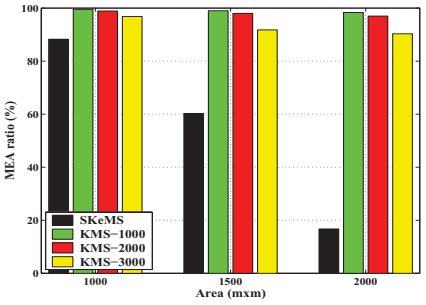
(c) 400 Keys in a 1500mx1500m

Figure 5. *MEA ratio* with different number of nodes

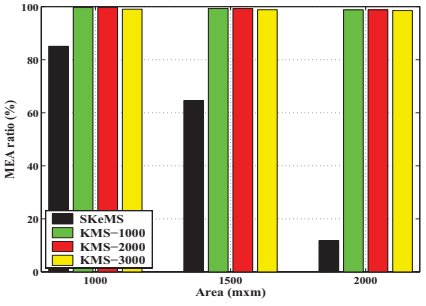
pool size will decrease the *MEA ratio*. For example, with 200 keys chosen from a pool size of 1000 keys, we have an *MEA ratio* of 98%, while with 3000 keys pool size with the same number of keys we have a ratio of 94.6%. Compared to our scheme, the results show that our scheme outperforms the KMS scheme in all different tested pool sizes. For example, with 300 keys, our scheme has an *MEA ratio* of 82%, where by using the KMS scheme with 3000 pool size we got a ratio of 98.6%. These results also show that, by increasing the number of available keys, we can provide a better *MEA ratio*, since we have more variety of keys to be assigned among the nodes in the network. For example, with 200 keys, the ratio is 86%, while with 500 keys it drops to 73%. The same results' trend can be seen in Fig. 4(b), where we distribute 400 nodes in a 1000m square field. Increasing network density (*the number of nodes in a square area size*) by increasing the number of nodes in the same area size, would increase the number of nodes that are vulnerable to malicious eavesdropping attack, since the number of neighboring nodes of the malicious nodes will increase. Our results in Fig. 4(b) show that by applying our SKeMS scheme, we can provide a better MEA ratio compared to that when applying KMS scheme with different pool sizes. For example, with 400 nodes and 200 keys, we can provide an MEA ratio of 86% compared to an MEA ratio of 96% when applying KMS scheme with 3000 pool size. Also increasing the number of keys available to be assigned among the nodes can provide a better security against malicious eavesdropping ability when applying our SKeMS scheme. Our results in 4(b) show that, by applying our SkeMS scheme with 200 keys we can provide an MEA ratio of 86% compared to that of 77% with 500 keys. On the other hand, KMS scheme required a large pool size to provide a better varieties of keys to be assigned to each node. The same results' trend can be seen in Fig. 4(c), where we distribute 500 nodes in a 1000×1000 area size.

Fig. 5 show the results of our second scenario, in which we studied the schemes' performance with different number of nodes $(200-500)$ in 225×10^4 area size. In Fig. 5(a) we tested the performance with 200 available keys to be assigned in the network. The results in Fig. 5(a) show that, by applying our SKeMS scheme, the *MEA ratio* is increasing with the increase in the number of nodes, due to having more common shared keys between the nodes in the neighborhood. However, our scheme's MEA ratio is still better compared with the *MEA ratio* when applying the KMS scheme. For example, with 200 nodes and 200 keys we can provide an MEA ratio of 24% compared to that of 87% compared to that when applying KMS scheme with 3000 pool size. The same results' trend can be seen in Fig. 5(b) and Fig 5(c), with 400 and 500 keys respectively to be assigned.

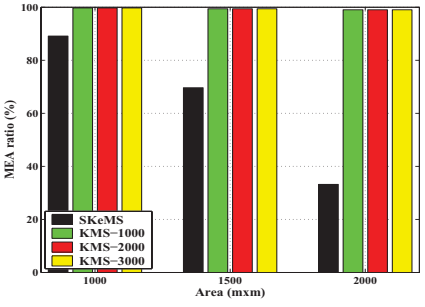
To show the relation between the number of available keys and the number of nodes in the network, we applied our SKeMS scheme to three different network sizes and compared it with that when applying KMS scheme. The results are shown in Fig. 6. In Fig. 6(a) we show the results of the case, where 300 nodes distributed in different area sizes (100×10^4, 225×10^4 and 400×10^4 square meters) with 200 available keys. The results show that in sparse network, the number of neighboring nodes became smaller with increasing the area size, this indeed decreases the number of nodes that can be affected with the malicious eavesdropping attack. For example, with our SKeMS scheme in 100×10^4 square area size, the *MEA ratio* is 86%, while by doubling the area size to 400×10^4 square meters, we can decrease the *MEA ratio* to 16.7%. Increasing the area size also has an effect on the KMS scheme performance, for example, with 3000 pool size and 200 keys in 100×10^4 meter square area size the *MEA ratio* is 98%, while this ratio decreased in a 2000 square meters area size to 90%. The same trend can be seen in Fig. 6(b) with 400 nodes and 300 keys. It is obvious that our SKeMS scheme performs much better than the previous KMS scheme in providing smaller *MEA ratio* which indicates a more secured network. Also we show the results with 500 nodes and 400 keys in different area sizes in Fig. 6(c), which show the same trend as in the



(a) 300 nodes with 200 keys

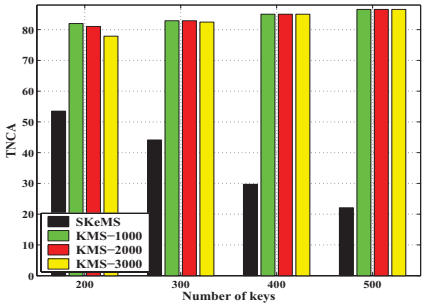


(b) 400 nodes with 300 keys

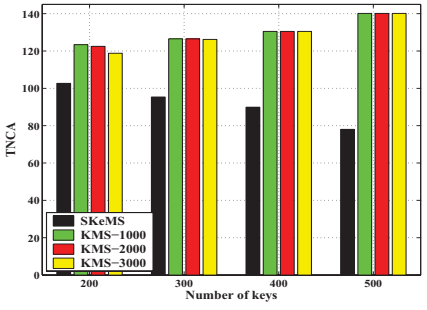


(c) 500 nodes with 400 keys

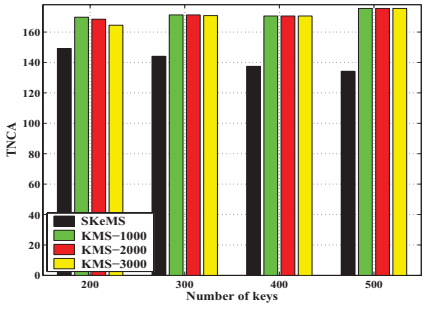
Figure 6. *MEA ratio* in different area sizes



(a) 200 Nodes in a $1000m$x$1000m$



(b) 300 Nodes in a $1000m$x$1000m$



(c) 400 Nodes in a $1000m$x$1000m$

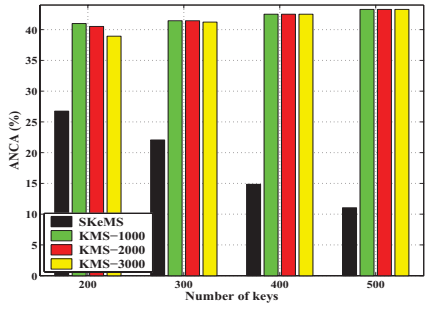
Figure 7. Total *NCA* with different number of keys

previous figures.

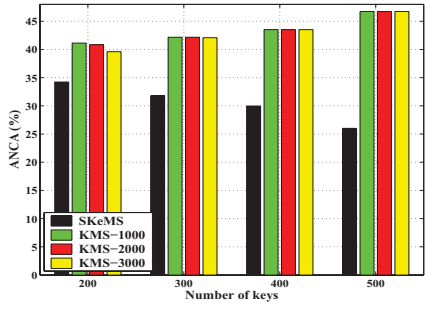
The second metric used for performance evaluation is ***total neighbor compromise ability*** (denoted as *TNCA* in the figures), which is calculated as the total number of 2-hop nodes away from the compromised node(s) that are vulnerable to eavesdropping malicious attack (discussed in subsection II-B) launched by the malicious node(s). To show the performance of our scheme and compared it to the KMS scheme, we set number of malicious nodes in the network to be 10% of the total number of nodes in the networks. Our results for the *total neighbor compromise ability* performance metric are shown in Fig .7. In Fig. 7(a) we show the results of distributing 200 nodes in 100×10^4 square area size with $(200-500)$ available keys to be assigned to/among nodes in the network. Our results show that by applying our SKeMS scheme, we can decrease the total number of nodes that are vulnerable to malicious eavesdropping attack, by increasing the number of available keys to be assigned among all the nodes in the network, which provides a large variety of keys to be assigned among the nodes, while keeping the network securely connected. For example, with 200 keys, 53 nodes are vulnerable to malicious eavesdropping attack compared to that of 22 nodes when applying our SKeMS scheme with 500 keys. It is obvious that, our SKeMS scheme provide a better security among the network, by providing smaller number of nodes that are vulnerable to malicious eavesdropping attack as shown in the results in Fig. 7.

To support our observation, we tested our SKeMS scheme and compared it to the KMS scheme with different number of nodes under the same circumstances. Fig. 7(b) show the results of 300 nodes in 100×10^4 square area size with $(200-500)$ available keys to be assigned to/among nodes in the network. It can be seen that our SKeMS scheme provides a more secured network against the malicious eavesdropping attack (discussed in subsection II-B). The same results' trend can be seen in Fig. 7(c) where we distributed 400 nodes in 100×10^4 square area size.

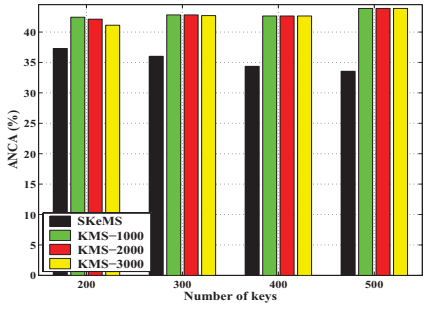
Our third metric used for performance evaluation is ***neighbor compromise ability ratio*** (denoted as *ANCA ratio* in the figures), which is calculated as the total number of 2-hop nodes away from the compromised



(a) 200 Nodes in a $1000m$x$1000m$



(b) 300 Nodes in a $1000m$x$1000m$



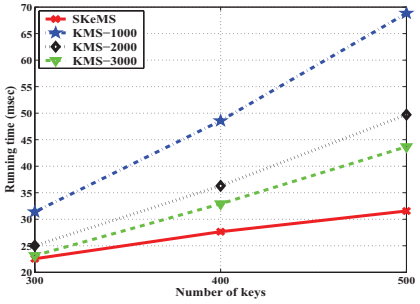
(c) 400 Nodes in a $1000m$x$1000m$

Figure 8. Average *NCA* with different number of keys

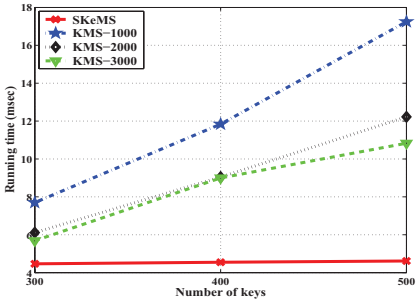
node(s) that are vulnerable to eavesdropping malicious attack (discussed in subsection II-B) launched by the malicious node(s) to the total number of nodes in the networks. The smaller the ANCA ratio provided by a key management scheme, the more secure is the network. Our results for the third performance metric are shown in Fig. 8. We set the number of malicious nodes in the network to be 10% out of the total number of nodes in the network. Fig. 8(a) show the results of applying our SKeMS scheme compared to the KMS scheme, with 200 nodes distributed in 100×10^4 square area size. We tested both key management schemes with different number of available keys ranges between (200−500 keys) to be assigned to/among the nodes in the network. It can be seen that, with our SKeMS scheme, we can provide the network with better security against malicious eavesdropping attack, indicated by smaller ANCA ratio compared to that when applying KMS scheme. For example, with 200 nodes and 400 keys we can provide an ANCA ratio of 15% with our scheme compared to that of 42% when applying the KMS scheme. We also tested the schemes performance with more dense network, where we distributed 300 and 400 nodes in 100×10^4 square area size. The results are shown in Fig. 8(b) and Fig. 8(c) respectively. The same trend can be seen in the figures, where our scheme provides more secure network with a smaller ANCA ratio compared to that when applying the KMS scheme.

Our last performance metric used for performance evaluation is the ***running time***, which is defined as the running time that the scheme takes to assign the available keys to/among the nodes in the network. We tested our scheme and compared it to the KMS scheme in two different scenarios shown in Fig. 9 and Fig. 10. Our first scenario is shown in Fig. 9(a), where 200 nodes are randomly distributed in a 100×10^4 square meters field size. We assign 300, 400 and 500 keys for both schemes. For the KMS scheme we measured the running time with different pool sizes (1000−3000). Since the KMS scheme depends on the pool size, where with different pool sizes, it can be seen that the KMS scheme takes more time in assigning the keys from 1000 keys pool size and find the *MCA* of the network compared to larger pool sizes, due to having more number of neighboring nodes that share more common keys. Compared to our scheme, it can be seen that our scheme outperforms the KMS scheme in all tested cases. Two more cases were studied and the corresponding results are shown in Fig. 9(b) and Fig. 9(c), where we changed the area size to be 225×10^4 and 200×10^4 respectively. It can be seen than our SKeMS scheme most of the time provides more secured network in a shorter period of time compared to that when applying the KMS scheme. For example, in Fig. 9(b), the results show that, with 400 keys, our scheme assigns the available keys among the network in $5msec$ compared to that $8.7msec$ when applying the KMS scheme with 3000 pool key size.

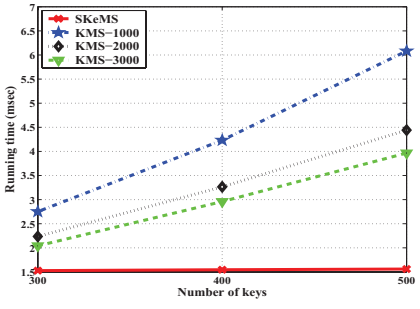
The second scenario is shown in Fig. 10, in which, we changed the network density by increasing the area size from 100×10^4 to 200×10^4 square meters. It can be seen that our SKeMS scheme provides a more secured network in less time compared to the KMS scheme. which needs to randomly pick k number of different keys from the key pool and store them at each node. We tested the KMS scheme with 1000, 2000 and 3000 key pool sizes. The results show that, in sparse networks the number of neighboring nodes decreases, which leads to having less number of nodes that shared common keys in the neighborhood thus consumes less time in assigning keys to generate a secured network that is resistant to the eavesdropping attack. In Fig. 10(c) we showed the performance results of our SKeMS scheme compared to the KMS scheme, when distributing 300 nodes in different area sizes and 500 available keys to be assigned to/among all the nodes in the network. It is obvious that our scheme consumes less time to provide a better secure network compared to that with the KMS scheme. For example, in 225×10^4, our SKeMS scheme consume $27msec$ to assign the available keys among the nodes in the network compared to that of $62msec$ when applying the KMS scheme under the same circumstances.



(a) 200 nodes in 1000×1000

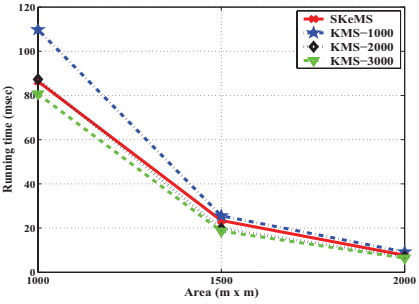


(b) 200 nodes in 1500×1500

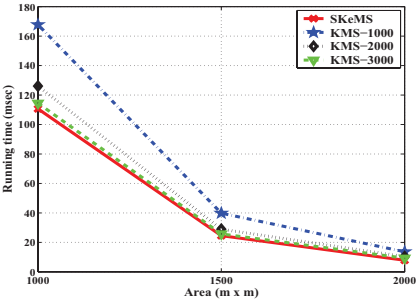


(c) 200 nodes in 2000×2000

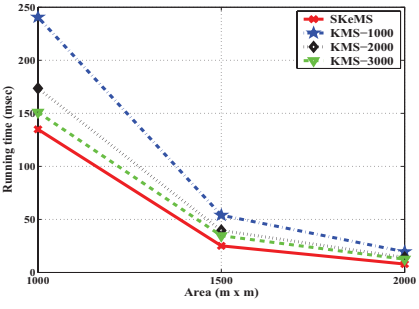
Figure 9. Running time with different number of keys



(a) 300 nodes with 300 keys



(b) 300 nodes with 400 keys



(c) 300 nodes with 500 keys

Figure 10. Running time for different area sizes

VI. Conclusion

In this paper, we defined the **S**ecure **K**ey **M**anagement **S**cheme (SKeMS) problem, and presented an effective solution that seeks a key assignment to provide a network that is resistant to malicious eavesdropping attacks. Previous work has shown that the number of assigned keys at each node depends on the key pool size. In our scheme we used the provided keys without depending on the pool size, and assigned these keys among all the nodes in the network. Simulation results showed that our solution performs well in terms of *malicious eavesdropping ability ratio*, *total neighbor compromise ability*, *neighbor compromise ability ratio* and *less running time*. To sum up, in this paper, we showed that a good key management scheme can ensure a more secure network.

References

[1] I. F. Akyildiz, X. Wang, W. Wang, Wireless mesh networks: a survey; *Elsevier Journal of Computer Networks*, vol.47, Issue.4, pp.445-487, 2005.

[2] A. Raniwala, T. Chiueh, Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network; *IEEE INFOCOM'05*, vol.3, pp.2223- 2234, Miami, Fl., USA.

[3] A. Raniwala, K. Gopalan, T. Chiueh, Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks; *ACM MobiCom'04*, Vol.8, Issue.2, pp.50-65, Philadelphia, PA, USA.

[4] X. Zhao, Y. Lv, T. H. Yeap, B. Hou, A Novel Authentication and Key Agreement Scheme for Wireless Mesh Networks; *In Proceedings of NCM'09*, pp.471-474, Washington, DC, USA.

[5] L. Gao, E. Chang, S. Parvin, S. Han, T. Dillon, A Secure Key Management Model for Wireless Mesh Networks; *IEEE AINA'10*, pp.655-660, Washington, DC, USA.

[6] N. Asokan, P. Ginzboorg, Key Agreement in Ad Hoc Networks; *Computer Communications*, vol.23, pp.1627-1637, 2000.

[7] X. Du, Y. Xiao, M. Guizani, H. H. Chen, An effective key management scheme for heterogeneous sensor networks; *Ad Hoc Networks, Special Issues in Sensor and Ad Hoc Networks*, vol.5, Issue.1, pp.24-34, 2007.

[8] P. Loree, K. Nygard, X. Du, An efficient post-deployment key establishment scheme for heterogeneous sensor net-

works; *IEEE GLOBECOM'09*, Honolulu, Hawaii, USA.
[9] L. Eschenauer, V. D. Gligor, A key-management scheme for distributed sensor networks; *ACM CCS'02*, Washington, DC, USA.
[10] S. P. Chan, R. Poovendran, M. T. Sun, A key management scheme in distributed sensor networks using attack probabilities; *IEEE GLOBECOM'05*, vol.2, pp.5, St. Louis, MO, USA.
[11] J. Shi, R. Zhang, and Y. Zhang, Secure range queries in tiered sensor networks; *IEEE INFOCOM'09*, pp.945-953, Rio de Janeiro, Brazil.
[12] M. Cagalj, J. Hubaux, C. Enz, Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues; *ACM MobiCom'02*, Atlanta, Georgia, USA.
[13] F. Kandah, W. Zhang, X. Du, Y. Singh, A Secure Key Management Scheme in Wireless Mesh Networks; *IEEE ICC'11*, pp.1-5.
[14] J. Tang, G. Xue, W. Zhang, Interference-Aware Topology Control and QoS Routing in Multi-Channel Wireless Mesh Networks; *ACM MobiHoc'05*, pp. 68-77, Urbana-champaign, IL, USA.
[15] W. Zhang, F. Kandah, J. Tang, K. Nygard, Interference-Aware Robust Topology Design in Multi-Channel Wireless Mesh Networks; *IEEE CCNC'10*, pp.6-10, LAS Vegas, NV, USA.