

SKAIT: A Parameterized Key Assignment Scheme for Wireless Networks

Ramon Novales, Neeraj Mittal, and Kamil Sarac

Department of Computer Science

The University of Texas at Dallas

Richardson, TX 75080, USA

{rnovales, neerajm, ksarac}@utdallas.edu

Abstract—In this paper, we propose **SKAIT**, a parameterized symmetric key pre-distribution scheme that guarantees a secure and confidential channel between every pair of nodes in a wireless network. Parameterization enables control over the number of keys assigned to a node, and allows users to trade increased key space complexity for improved collusion resistance. We provide an analysis of the space complexity, time complexity, and collusion resistance, and we show that message exchange is secure against internal and external eavesdroppers. We also show via analysis and simulation that **SKAIT** possesses the ability to make efficient use of key storage capacities of at least $3\sqrt{n}$, and collusion resistance superior to that of two recently proposed schemes when the number of colluding nodes is small.

Keywords—collusion resistance; confidential communication; key pre-distribution; symmetric key assignment

I. INTRODUCTION

Due to their broadcast nature, communications in wireless networks are susceptible to eavesdropping and tampering by unauthorized parties or adversaries. Certain classes of networks, such as mobile [1], [2], ad-hoc [3], [4], and wireless sensor networks [5], [6], are composed of nodes that may be resource-constrained in terms of storage space, computational capability, or energy (battery power). Further, such networks may operate in an environment where access to a trusted central authority or other infrastructure is limited or nonexistent. In the face of such constraints, symmetric key pre-distribution has shown itself to be an appealing solution, and several schemes have been proposed [5], [7]–[15] to distribute these keys to nodes in the network. In such schemes, each node is assigned a set of keys drawn from a common key pool. If two nodes share one or more keys in common, then the nodes can use those key(s) to establish a secure channel between them. When we say that a channel is secure, we mean that adversaries outside the network do not possess the keys required to decrypt messages sent along that channel.

A related issue is that of *confidentiality* [10]. When we say that a channel is confidential, we mean that only the channel endpoints have the keys required to decrypt messages sent along that channel. In some applications, end-to-end confidentiality may be desired—that is, only the designated recipient of a message should be able to decrypt its contents. Consider, for example, the intermediate nodes

in a multi-hop path. Although they are part of the network, it may be required that such nodes are unable to decrypt the messages that they pass along. It is possible for a key pre-distribution scheme to enable secure channels, but not end-to-end confidentiality. In the basic scheme proposed by Eschenauer and Gligor [5], nodes are assigned keys that are selected probabilistically from a common key pool. Du, *et al.* build upon that scheme by using deployment knowledge to increase the probability that neighboring nodes will share a key in common [7]. To a similar end, Liu *et al.* propose a scheme which partitions nodes into groups and then makes use of that group knowledge when assigning keys. For all three of these key pre-distribution schemes, the same key may be used by multiple pairs of nodes to create a secure channel. In fact, nodes outside of these communicating pairs may also possess that same key. Thus, no guarantee of confidentiality can be made.

Several schemes which do support confidentiality have been proposed. Elmallah, *et al.* presented a logarithmic keying scheme which assigned $O(\log d)$ keys per node, where d is the maximum degree of a node in the network [12]. Gong and Wheeler developed a grid-based scheme which assigned $O(\sqrt{n})$ keys per node [9]. Kulkarni, *et al.* proposed a variant of this scheme, demonstrating that it is optimal provided no two nodes share more than two keys [10]. Aiyer, *et al.* then presented a family of k (where $1 \leq k \leq \log n$) key grid protocols, which enabled confidential communication between any pair of nodes in a network [11]. Mittal then extended the work of Elmallah, *et al.*, proposing a scheme that enables confidential communication by decomposing a network into a series of bipartite graphs [13]. Securing each of these bipartite graphs individually (using the scheme proposed by Elmallah, *et al.*) secures the network as a whole. Each node is assigned $O(\log^2 n)$ keys. With TASK [14], Novales and Mittal further extended this work. Template key assignment instances are used to secure each bipartite graph; these templates trade a slight decrease in collusion resistance for a larger reduction in space complexity. This allows bipartite graphs to be selectively reinforced, allocating key storage space to graphs which secure larger numbers of channels. The end result is a scheme with improved collusion resistance and the ability to use a wide range of key storage capacities, which other

schemes may not be able to fully utilize.

A. Our Contributions

We propose SKAIT (Symmetric Key Assignment by Identifier-Triplet), a novel parameterized symmetric key pre-distribution scheme that guarantees a secure and confidential channel between every pair of nodes in a network. SKAIT is suitable for use when network nodes face computational or storage constraints, such as in ad-hoc, mobile, or sensor networks. The parameterization of SKAIT allows a user to control the number of keys assigned to a node, making a trade-off between collusion resistance and key space complexity. This control is fine-grained, allowing effective use of available key storage capacity. Each node is assigned $w + 2\lceil \frac{n}{w} \rceil$ keys, where n is the number of nodes in the network and w is an adjustable parameter ranging from 3 to \sqrt{n} ; SKAIT is focused on making use of key storage capacities between $3\sqrt{n}$ and $\frac{2n}{3} + 3$ (for a network of 4096 nodes, this range is from 192 to 2734). Given the evolution of devices in ad-hoc, mobile, and sensor networks, it is not unreasonable to assume that key storage space is likely to grow with time. We demonstrate, via analysis, that message exchange is secure against internal and external eavesdroppers, that the key securing a link can be generated in $O(1)$ time, and that SKAIT assigns, at a minimum, $3\sqrt{n}$ keys to each node. We also show via analysis and simulation that SKAIT possesses collusion resistance superior to that of two recently proposed schemes, and the ability to efficiently make use of key storage capacities greater than $3\sqrt{n}$. When configured such that an equal number of keys is assigned to a node, our simulation results for networks of 4096 nodes show an 89.1% reduction in the number of lost channels from the scheme proposed by Novales and Mittal, and a 77.9% reduction from the family of protocols proposed by Aiyer, *et al.* SKAIT uses a novel algorithmic approach, and is not a refinement or expansion of prior work.

II. SYSTEM MODEL

We use the typical adversary and key distribution model [5]—keys are pre-loaded onto nodes prior to their deployment or activation, and once a node is captured, the adversary has access to any and all stored secrets. Further, we make no distinction between colluding nodes and captured nodes pooling their keys under the control of an adversary. From the perspective of the remainder of the network, both situations are the same—some number of nodes have pooled their keys and are able to collectively decrypt messages that they could not individually. Consequently, when describing nodes we use the terms “colluding” and “captured” interchangeably.

Establishing a channel between any pair of nodes implies that the network is a fully connected *logical* communications graph. A channel may be either a direct link between two nodes within radio range, or a multi-hop path between two

nodes out of radio range. We place no restriction on the distribution of nodes or whether the nodes are mobile or stationary; if the underlying routing protocol can accommodate mobile nodes, then SKAIT is unaffected. The confidentiality property also means that the multi-hop channel between two nodes is not restricted to a fixed path—any live path can be used. We consider denial of service attacks, routing attacks, and attacks on the symmetric cryptography algorithms themselves to be out of the scope of this paper, as these issues would affect all key pre-distribution schemes to some degree. We instead focus on the effects of collusion, as measured by r -collusion resistance [10]. Given r colluding nodes, the r -collusion resistance is the minimum fraction of channels in the network that remain secure against eavesdropping.

III. SKAIT

SKAIT generates a unique identifier-triplet for each node in the network; these identifiers are then used as indices into three smaller key assignment problems. The keys thus obtained are then combined to generate a distinct key to secure each channel in the network. The channels are secure and confidential because no nodes in the network (other than the channel endpoints) are capable of generating the key used to secure the channel. We first give an informal overview to show some of the intuition behind the scheme.

A. Overview

The main intuition behind SKAIT is to break the problem of assigning symmetric keys to nodes in a network into a set of smaller problems. For example, consider a network of n nodes, partitioned into w groups of equal size. Each group has a unique ID (ranging in this example from 0 to $w - 1$), and each node within a group is assigned a primary ID (0 to $\frac{n}{w} - 1$ in this example) which is unique only to its group; this means, for example, that there are w nodes with primary ID 0, but only one in each group. Consider one group, and assign each node within the group $\frac{n}{w} - 1$ pairwise keys such that each node can communicate securely and confidentially with every other node in the group. We denote the pairwise key shared by nodes with primary IDs i and j by $pid.(i, j)$. If we replicate this exact key assignment across all groups, extra steps must be taken to secure intra-group communications, as all nodes with the same primary ID will hold the exact same set of keys. Say we assign each group (and each node in the group) a unique symmetric key corresponding to the group ID u , denoted by $gid.(u)$. If nodes with primary IDs i and j within group u use the combination of $pid.(i, j)$ and $gid.(u)$ to communicate, then security and confidentiality for intra-group communications is restored; $pid.(i, j)$ may be held by nodes in every group, but only the ones in group u hold $gid.(u)$. We can then extend this idea to secure inter-group communication, and by taking advantage of the fact that primary ID-based key

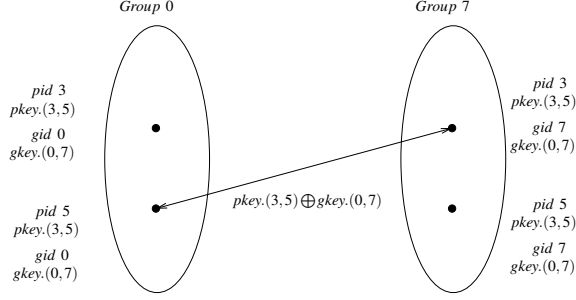


Figure 1. Communication not confidential with $pkey$ and $gkey$ alone.

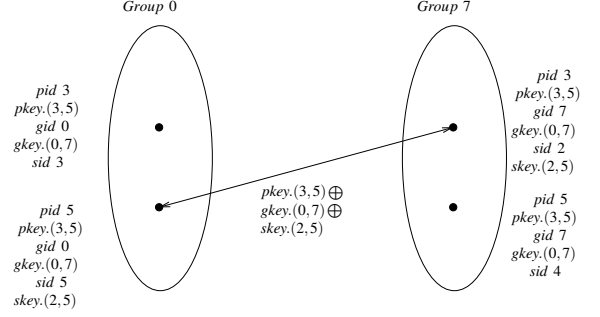


Figure 2. Communication confidential with $pkey$, $gkey$ and $skey$.

assignments are replicated across groups we can reduce the number of keys that a node must store, as explained below.

We first modify the primary ID key assignment to add a key used to communicate with a node with the same primary ID (e.g. $pkey.(i, i)$). We also modify the group ID based key assignment to make it similar to the primary ID assignment—each group ID has a corresponding symmetric key for use with each other group ID, as well as itself, denoted by $gkey.(u, v)$ (where u and v are valid group IDs). The node with primary ID i in group u and the node with primary ID j in group v would use the combination of $pid.(i, j)$ and $gid.(u, v)$ to establish a secure channel.

Using the combination of $pkey$ and $gkey$, it first appears that this is sufficient to provide secure, confidential communication. This, however, is not the case. Consider a network with 64 nodes, divided into 8 groups. If the node with primary ID 0 in group 0 wishes to communicate with the node with primary ID 0 in group 7, they can use the combination of $gkey.(0, 7)$ and $pkey(0, 0)$. Only nodes in groups 0 and 7 possess $gkey.(0, 7)$, and of those nodes, only nodes with primary ID 0 possess $pkey.(0, 0)$, making this channel secure. However, consider the scenario shown in Figure 1, where the node with primary ID 5 in group 0 uses the combination of $gkey.(0, 7)$ and $pkey.(3, 5)$ to communicate with the node with primary ID 3 in group 7. While no nodes outside these two groups hold these keys, the node with primary ID 3 in group 0 holds both $gkey.(0, 7)$ and $pkey.(3, 5)$ (as does the node with primary ID 5 in group 7).

SKAIT solves this problem by generating another identifier (which we refer to as the *secondary ID*) for each node, such that two nodes with the same primary ID or the same group ID are guaranteed to have different secondary IDs. The secondary ID is then used as the basis for pairwise symmetric key assignment (in a manner similar to the primary and group IDs), and the combination of group, primary, and secondary ID-based keys is used for communication. The uniqueness of the secondary ID within the group means that nodes within the group will no longer be able to eavesdrop, as shown in Figure 2. Neither the node with primary ID 3 in group 0 nor the node with primary ID 5 in group 7 have the

secondary ID necessary to possess $skey.(2, 5)$. An interesting question is how to generate such secondary IDs efficiently.

While the intuition behind SKAIT is simple, we give a formal presentation of the scheme for completeness. We discuss ways to generate secondary IDs so as to satisfy the uniqueness property. We also demonstrate using mathematical analysis as well as simulation experiments that *this simple scheme exhibits surprisingly good collusion resistance* for small numbers of colluding nodes. We begin with the generation of the group, primary, and secondary IDs.

B. Node Identifiers

Consider a wireless network composed of n nodes, each assigned a unique identifier taken from the range $[0, n)$. The network is partitioned into w groups, where $3 \leq w \leq \sqrt{n}$; each group contains m nodes, where $m = \lceil \frac{n}{w} \rceil$. Note that this implies that $w \leq m$. Each group is then assigned a unique identifier taken from the range $[0, w)$. Each node is further assigned a group, primary, and secondary identifier (ID), as follows:

1) *Group identifier*: We use gid_i , where $0 \leq i < n$, to denote the group ID for the node with unique identifier i . Node i is assigned to the j^{th} group, where $j = i \bmod w$. Thus, $gid_i = i \bmod w$.

2) *Primary identifier*: We use pid_i , where $0 \leq i < n$, to denote the primary identifier for the node with unique identifier i . Let $pid_i = \lfloor i/w \rfloor$; this value is unique to its group, and in the range $[0, m)$. Note that while a primary ID is unique within a particular group, it is not unique across all groups. That is, each group j , where $0 \leq j < w$, will contain nodes with primary ID k , where $0 \leq k < m$.

3) *Secondary identifier*: Each node i , where $0 \leq i < n$, is assigned a secondary ID, denoted by sid_i , where $sid_i = (gid_i + pid_i) \bmod m$. As with the primary ID, the secondary ID is in the range $[0, m)$, and is *unique* within a particular group, but not across all groups. While the uniqueness of a secondary ID with respect to a particular primary or group ID may seem intuitive, we present proofs of these properties for completeness.

Lemma 1. *If nodes i and j , where $0 \leq i, j < n$ and $i \neq j$, have the same primary ID, they must have different secondary IDs.*

Proof: If $sid_i = sid_j$, then $(gid_i + pid_i) \bmod m = (gid_j + pid_j) \bmod m$. Using modulo arithmetic, it implies that $gid_i + pid_i = gid_j + pid_j \pm km$, where k is some non-negative integer. If $pid_i = pid_j$, then $gid_i = gid_j \pm km$. Since the group IDs are from the range $[0, w)$, and $w \leq m$, we can conclude that $gid_i = gid_j$. Clearly, if two nodes have the same group IDs and the same primary IDs, then they are the same nodes, implying $i = j$, a contradiction. ■

Lemma 2. *If nodes i and j , where $0 \leq i, j < n$ and $i \neq j$, have the same group ID, they must have different secondary IDs.*

Proof: The proof is analogous to that of Lemma 1. ■

The secondary ID may be generated using other functions, so long as the *uniqueness* condition is satisfied. For example, it can be verified that any function of the form $sid_i = (gid_i + pid_i + c) \bmod m$, where c is some constant, also satisfies Lemmas 1 and 2.

C. Key Pool Construction

The key pool \mathcal{K} is the union of three distinct component key pools: \mathcal{K}^g , \mathcal{K}^p , and \mathcal{K}^s , representing the group, primary, and secondary identifiers respectively. Let t represent the number of IDs for a given type ($t = w$ for gid , and $t = m$ for pid and sid). The component key pools are then constructed such that there is a unique key for each (i, j) pair, where $0 \leq i, j < t$; note that this includes pairs where $i = j$. There are $\frac{(t+1)t}{2}$ such pairs, so each component key pool contains $\frac{1}{2}(t^2 + t)$ keys.

D. Key Assignment

Let K_i denote the key ring for node i , where $0 \leq i < n$. The key ring K_i is the union of three distinct component key rings: K_i^g , K_i^p , and K_i^s , drawn from the key pools \mathcal{K}^g , \mathcal{K}^p , and \mathcal{K}^s , respectively. Component key ring K_i^g is the set of keys from \mathcal{K}^g which correspond to the (gid_i, j) pairs, where $0 \leq j < w$. Similarly, K_i^p is the set of keys from \mathcal{K}^p corresponding to the (pid_i, j) pairs, and K_i^s is the set of keys from \mathcal{K}^s corresponding to the (sid_i, j) pairs, where $0 \leq j < m$. Each node is therefore assigned $|K_i^g| + |K_i^p| + |K_i^s|$ keys, so $|K_i| = w + 2m$.

E. Creating Secure Channels

If nodes i and j wish to establish a secure channel, they select a set of keys common to K_i and K_j , denoted by CK_{ij} , such that $CK_{ij} \cap K_x \neq CK_{ij}$, where $0 \leq x < n$ and $x \neq i, j$. The XOR of all keys in CK_{ij} is then used to encrypt and decrypt all messages sent along the channel. We use $sk.(i, j)$ to denote this key.

1) *Generating $sk.(i, j)$:* To generate $sk.(i, j)$, node i first calculates gid_i , gid_j , pid_i , pid_j , sid_i , and sid_j as described in Section III-B. Node i then selects the pairwise keys $gkey.(gid_i, gid_j)$, $pkey.(pid_i, pid_j)$, and $skey.(sid_i, sid_j)$ from K_i^g , K_i^p , and K_i^s , respectively. Node j generates $sk.(i, j)$ in a similar manner.

2) *Security from Eavesdropping:* The link secured with $sk.(i, j)$ is secure from eavesdropping. Nodes other than i and j will not be able to decrypt the message contents, as they will not be able to generate $sk.(i, j)$.

Theorem 1. *Aside from nodes i and j , no other nodes possess the three keys required to generate $sk.(i, j)$.*

Proof: Nodes i and j will use $gkey.(gid_i, gid_j)$, $pkey.(pid_i, pid_j)$, and $skey.(sid_i, sid_j)$ to generate $sk.(i, j)$. Let node x represent a would-be eavesdropper. In order to generate $sk.(i, j)$, node x must have the gid of either node i or node j , the pid of either node i or node j , and the sid of either node i or node j . Consider the possible locations for node x :

Case 1: Node x is in a different group from both i and j . This means that $gid_x \neq gid_i$ and $gid_x \neq gid_j$, and by construction, node x cannot possess $gkey.(gid_i, gid_j)$. Thus, node x cannot generate $sk.(i, j)$, and cannot eavesdrop.

Case 2: Node x is in the same group as both i and j ; that is, $gid_x = gid_i = gid_j$. By construction, $pid_x \neq pid_i \neq pid_j$, and node x cannot possess $pkey.(pid_i, pid_j)$. Thus, node x cannot generate $sk.(i, j)$, and cannot eavesdrop.

Case 3: Node x is in the same group as node i , and nodes i and j are in different groups; that is, $gid_x = gid_i$, and $gid_i \neq gid_j$. Two nodes in the same group cannot share the same pid ; therefore, in order to be able to eavesdrop, $pid_x = pid_j$. These ID assignments mean that node x possesses $gkey.(gid_i, gid_j)$ and $pkey.(pid_i, pid_j)$. Lemma 2 states that nodes with the same group ID must have different secondary IDs; therefore, $sid_x \neq sid_i$. Lemma 1 states that nodes with the same primary ID must have different secondary IDs; therefore, $sid_x \neq sid_j$. Thus, by construction, node x cannot possess $skey.(sid_i, sid_j)$, cannot generate $sk.(i, j)$, and cannot eavesdrop. The same is true if node x is in the same group as j .

Since node x is unable to construct $sk.(i, j)$, it cannot eavesdrop on nodes i and j . ■

3) *Time Complexity:* Calculation of the group, primary, and secondary IDs are $O(1)$ operations, as is retrieving the appropriate pairwise key and calculating the XOR. Thus, the shared key can be generated in $O(1)$ time.

F. Collusion Resistance

SKAIT protects against eavesdropping by a node acting alone; however, it is possible for two or more colluding nodes to pool their keys and thereby gain the ability to decrypt channels that they were unable to individually. We say a channel is *lost* if the keys necessary to decrypt

Table I
NOTATION FOR COLLUSION RESISTANCE

f_g	fraction of <i>gids</i> lost
f_p	fraction of <i>pids</i> lost
\bar{f}_g, \bar{f}_p	$(1-f_g), (1-f_p)$
$ B , C , B \cap C $	number of nodes in $B, C,$ and $B \cap C$
$ B_\ell , C_\ell , (B \cap C)_\ell $	number of channels between nodes in $B, C,$ and $B \cap C$
$f_{ B_\ell }, f_{ C_\ell }, f_{ (B \cap C)_\ell }$	fraction of channels between nodes in $B, C,$ and $B \cap C$

messages along the channel are held by an adversary or a group of colluding nodes (which may include x and/or y). We say a node is lost if it is either under the control of an adversary, or belongs to a group of colluding nodes. Similarly, we say a *gid*, *pid*, or *sid* is lost if it belongs to a node which is lost. If a node or identifier is lost, this implies that all keys associated with that node or identifier are held by an adversary or are pooled by colluding nodes. Given r colluding nodes, the r -collusion resistance of a scheme, denoted by $\rho.r$, is the minimum fraction of channels which remain secure.

Fact 1. $\rho.r = 0$ when $r \geq m$.

We next present an analysis of the r -collusion resistance when $2 \leq r < m$.

Theorem 2. When $2 \leq r < m$,

$$\rho.r \geq \begin{cases} \bar{f}_p^2 + \bar{f}_g^2 - \bar{f}_p^2 \bar{f}_g^2 - \frac{\bar{f}_g + \bar{f}_p}{n-1} & r \leq w \\ \bar{f}_p^2 - \frac{\bar{f}_p}{n-1} & r > w, \end{cases}$$

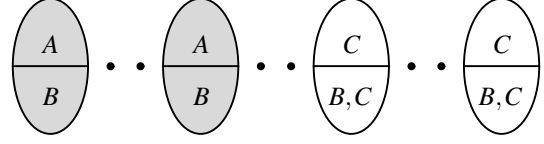
where $\bar{f}_p = 1 - \frac{rw}{n}$ and $\bar{f}_g = 1 - \frac{r}{w}$.

Proof: Consider a network of n nodes partitioned into w groups, as shown in Figure 3. The groups are arranged so that the groups that have lost their *gids* are to the left (these are the shaded groups in Figure 3). The nodes within the groups are arranged so that nodes which have lost their *pids* are in the top half of the oval, above the horizontal line.

Nodes in each group can be categorized into three sets as described:

- A: nodes that have lost both their *gid* and *pid*. This includes nodes that have been compromised, as well as nodes that are not compromised, but share a *gid* with one compromised node and a *pid* with another compromised node.
- B: nodes that have not lost their *pid*. Note that this can include nodes whose *gid* is compromised (in Figure 3 these are the nodes in region B in the shaded groups).
- C: nodes that have not lost their *gid*. Note that this can include nodes that have lost their *pid*.

We make use of the notation shown in Table I.



A: nodes that have lost both their *gid* and *pid*
B: nodes that have not lost their *pid*
C: nodes that have not lost their *gid*
□ = groups that have lost their *gid*

Figure 3. Node set membership.

The number of nodes in B is given by

$$|B| = (1-f_p) \binom{n}{w} = (1-f_p) \binom{n}{w} = \bar{f}_p n.$$

The number of nodes in C is given by

$$|C| = (1-f_g) \binom{n}{w} = (1-f_g) \binom{n}{w} = \bar{f}_g n.$$

The number of nodes in the intersection of B and C is

$$|B \cap C| = (1-f_p) \binom{n}{w} (1-f_g) \binom{n}{w} = \bar{f}_p \bar{f}_g n.$$

The number of channels between nodes in B is given by

$$|B_\ell| = \frac{(\bar{f}_p n)(\bar{f}_p n - 1)}{2},$$

making the fraction of channels between nodes in B

$$f_{|B_\ell|} = \frac{(\bar{f}_p n)(\bar{f}_p n - 1)}{2} \frac{2}{n(n-1)} = \frac{\bar{f}_p(\bar{f}_p n - 1)}{n-1}. \quad (1)$$

It can be shown that

$$\bar{f}_p - \frac{1}{n-1} \leq \frac{\bar{f}_p n - 1}{n-1} \leq \bar{f}_p \text{ when } 0 \leq \bar{f}_p \leq 1, \quad (2)$$

so substituting the results of (2) into (1) yields

$$\bar{f}_p \left(\bar{f}_p - \frac{1}{n-1} \right) \leq f_{|B_\ell|} \leq \bar{f}_p^2. \quad (3)$$

Similarly, the number of channels between nodes in C is

$$|C_\ell| = \frac{(\bar{f}_g n)(\bar{f}_g n - 1)}{2},$$

and like (3), it can be shown that the fraction of channels between nodes in C is

$$\bar{f}_g \left(\bar{f}_g - \frac{1}{n-1} \right) \leq f_{|C_\ell|} \leq \bar{f}_g^2.$$

The number of channels between nodes in the intersection of B and C is

$$|(B \cap C)_\ell| = \frac{(\bar{f}_p \bar{f}_g n)(\bar{f}_p \bar{f}_g n - 1)}{2}.$$

As with $f_{|B_\ell|}$ and $f_{|C_\ell|}$, it can be shown that the fraction of channels is

$$\bar{f}_p \bar{f}_g \left(\bar{f}_p \bar{f}_g - \frac{1}{n-1} \right) \leq f_{|(B \cap C)_\ell|} \leq \bar{f}_p^2 \bar{f}_g^2.$$

We can then determine the minimum fraction of uncompromised channels by summing the minimum values of $f_{|B_\ell|}$ and $f_{|C_\ell|}$, and subtracting away the maximum value of the overlap $f_{|(B \cap C)_\ell|}$:

$$f_{|B_\ell|} + f_{|C_\ell|} - f_{|(B \cap C)_\ell|} = \bar{f}_p^2 + \bar{f}_g^2 - \bar{f}_p^2 \bar{f}_g^2 - \frac{\bar{f}_g + \bar{f}_p}{n-1}.$$

From the discussion above, we know that an adversary can compromise one *pid* with every compromised node, so $f_p \leq \frac{r}{m} = \frac{rw}{n}$. Similarly, we can express f_g in terms of r and w . Since $w \leq m$ and $r \leq m$,

$$f_g = \begin{cases} \frac{r}{w} & r \leq w \\ 1 & r > w. \end{cases}$$

Thus,

$$\rho.r \geq \begin{cases} \bar{f}_p^2 + \bar{f}_g^2 - \bar{f}_p^2 \bar{f}_g^2 - \frac{\bar{f}_g + \bar{f}_p}{n-1} & r \leq w \\ \bar{f}_p^2 - \frac{\bar{f}_p}{n-1} & r > w. \end{cases} \quad (4)$$

A similar analysis may be performed for *gids* and *sids* (instead of *gids* and *pids*). However, to obtain a lower bound, we conservatively assume that *sids* are lost at the same rate as *pids*. ■

G. Space Complexity vs. Collision Resistance

By adjusting w from 3 to \sqrt{n} , we can control the number of keys assigned to a node. While $w = 2$ produces a valid key distribution, per Section III-D it will assign $n + 2$ keys per node. In this case, it would be preferable to use the pairwise scheme which assigns $n - 1$ keys to each node. Therefore, we can use SKAIT to take advantage of key storage capacity ranging from $3\sqrt{n}$ to $\frac{2n}{3} + 3$. Section III-F shows that as we decrease w , we are trading increased space complexity for increased r -collision resistance. Table II illustrates how the space complexity and r -collision resistance change for certain values of w (n is assumed to be sufficiently large such that $\frac{1}{n-1} \approx 0$). The plot in Figure 4 shows how the r -collision resistance changes with w , when $r \leq \frac{\sqrt{n}}{2}$. Note that w is expressed in this figure as a fraction of \sqrt{n} (that is, $.1\sqrt{n} \leq w \leq \sqrt{n}$).

Table II
SPACE COMPLEXITY, r -COLLISION RESISTANCE VS. w

w	keys	$\rho.r, r \leq \frac{\sqrt{n}}{2}$
\sqrt{n}	$3\sqrt{n}$	0.438
$\frac{\sqrt{n}}{2}$	$4\sqrt{n} + \frac{\sqrt{n}}{2}$	0.563
$\frac{\sqrt{n}}{4}$	$8\sqrt{n} + \frac{\sqrt{n}}{4}$	0.766
$\frac{\sqrt{n}}{8}$	$16\sqrt{n} + \frac{\sqrt{n}}{8}$	0.879

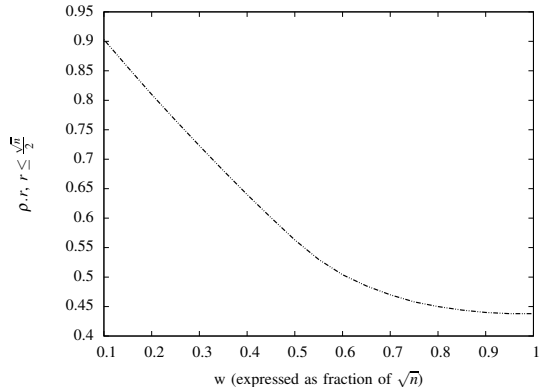


Figure 4. r -Collision Resistance vs. w

IV. COMPARISON WITH EXISTING SCHEMES

We select TASK [14] and the family of protocols proposed by Aiyer, *et al.* [11] for comparison with SKAIT. They are selected because they both support confidential communication in networks with fully connected logical graphs. Further, both schemes allow a trade-off to be made between space complexity and collision resistance. We are unaware of any other schemes which also satisfy these properties. We do not address wireless networking issues such as spatial or interference constraints, since they would affect all three schemes to the same extent.

A. Collision Resistance

1) *Analytical Results:* For the family of protocols proposed by Aiyer, *et al.*, we select $k = 2$. We do this because it has the highest collision resistance of any protocol in the family, save for $k = 1$, which corresponds to a complete pairwise key scheme where each node must store $n - 1$ keys. Further, the protocol where $k = 2$ assigns $4\sqrt{n}$ keys per node, which is comparable to SKAIT. As per the analysis in [11], if $r \leq \frac{\sqrt{n}}{2}$, then $\rho.r \geq \frac{1}{16}$.

In TASK, keys are assigned to nodes by an algorithm that seeks to meet a target number of keys. As a result, it is difficult to make a direct analytical comparison.

For SKAIT, we select $w = \sqrt{n}$; this corresponds to the *worst* r -collision resistance for the scheme, and assigns $3\sqrt{n}$ keys per node. As per (4), if $r \leq \frac{\sqrt{n}}{2}$, then $\rho.r \geq \frac{7}{16}$ (again, n is assumed to be sufficiently large such that $\frac{1}{n-1} \approx 0$). Clearly, the analytically established lower bound of r -collision resistance for SKAIT is much larger than that of the Aiyer, *et al.* scheme.

2) *Simulation Results:* We now present simulation results for networks containing 4096 nodes. For each simulation, the results of 30 trials were averaged together. Simulations were also performed for networks containing 1024 and 16384 nodes; the results are similar to those detailed below, but are omitted due to space constraints.

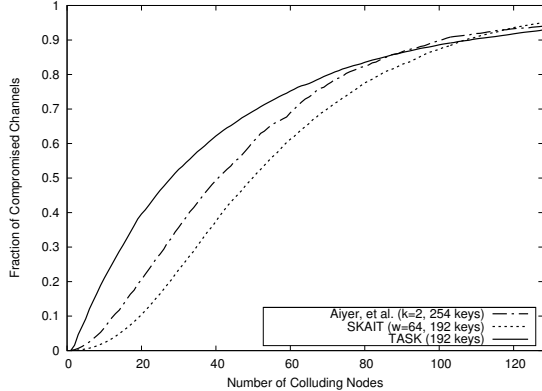


Figure 5. Collusion Resistance (4096 nodes)

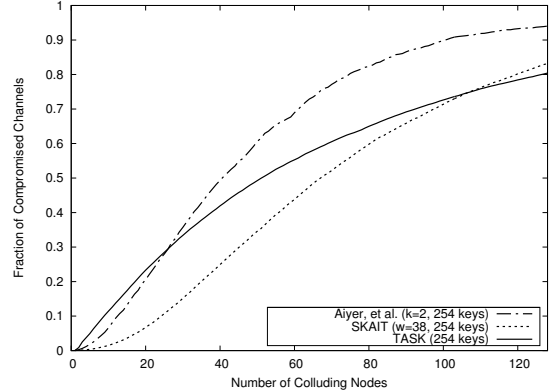


Figure 6. Collusion Resistance (4096 nodes)

In the first simulation, scheme parameters were set to match the conditions specified in Section IV-A1. For the Aiyer, *et al.* scheme, $k = 2$, which results in 254 keys assigned to each node. For SKAIT, $w = \sqrt{n} = 64$, which results in 192 keys assigned to each node. TASK was configured to assign 192 keys per node, the same as SKAIT. Simulation results are shown in Figure 5. When 16 nodes are colluding, SKAIT gives a 56.0% reduction in the fraction of lost channels over the Aiyer, *et al.* scheme, and a 79.8% reduction compared to TASK. When 32 nodes are colluding, SKAIT gives a 34.7% reduction in the fraction of lost channels over the Aiyer, *et al.* scheme, and a 54.5% reduction compared to TASK. Eventually there are fewer lost channels for TASK than for SKAIT; this crossover point occurs when 106 nodes are compromised. It is important to note that these results represent the *best* collusion resistance possible with the Aiyer, *et al.* scheme (barring $k = 1$, which corresponds to a pairwise scheme assigning 4095 keys to each node), and that SKAIT in this case is using 24.4% fewer keys than the Aiyer, *et al.* scheme.

In the next simulation, we adjust the parameters for TASK and SKAIT so that all three schemes assign 254 keys per node; simulation results are shown in Figure 6. When 8 nodes collude, SKAIT provides a 89.1% reduction in the number of lost channels compared to TASK, and a 77.9% reduction compared to the Aiyer, *et al.* scheme. When 32 nodes collude, SKAIT gives a 50.2% reduction in the number of lost channels compared to TASK, and a 55.9% reduction compared to the Aiyer, *et al.* scheme. When 25 nodes are colluding, TASK and the Aiyer, *et al.* scheme have approximately the same amount of lost channels. At this point, SKAIT shows a 61.6% reduction in the number of lost channels. Again, note that TASK has fewer lost channels than SKAIT once 104 nodes have been compromised. At this point, however, 74% of the channels in the network are lost, and the network may require rekeying or some other corrective action for continued use.

B. Utilization of Available Key Storage Capacity

Both TASK and the family of protocols proposed by Aiyer, *et al.* allow the user to adjust the number of keys assigned to a node. The Aiyer, *et al.* protocols, however, have a gap in the range of key storage capacities that they are able to utilize. In the Aiyer, *et al.* family of protocols, as k decreases, the collusion resistance increases. When $k = 2$, $4\sqrt{n}$ keys are assigned per node; $k = 1$ is the pairwise scheme where $n - 1$ nodes are assigned to each node. Since k must be an integral value, the Aiyer, *et al.* protocols cannot make full use of key storage space greater than $4\sqrt{n}$ but less than $n - 1$. While SKAIT cannot be used for key storage capacities less than $3\sqrt{n}$, it can, however, make use of key storage capacities ranging from $3\sqrt{n}$ to $\frac{2n}{3} + 3$.

We now present simulation results which demonstrate the ability of SKAIT to efficiently make use of available key storage capacity. As before, the simulated networks contain 4096 nodes, and each plot represents the average of 30 trials. Figure 7 compares the collusion resistance for the schemes when attempting to make use of key capacities of approximately 512 and 1024 keys. The Aiyer, *et al.* scheme is omitted because, as noted above, it can only make use of 254 keys. SKAIT assigns 499 keys ($w = 17$), and 1032 keys ($w = 8$). TASK is able to make finer adjustments in the number of keys assigned, and allocates 512 and 1024 keys. Note that until 33 nodes are compromised, when SKAIT assigns 499 keys per node, there are fewer lost channels than when TASK assigns 1024 keys. When 16 nodes are compromised, SKAIT with 499 keys per node gives a 71.9% reduction in lost channels versus TASK assigning 512 keys and a 39.2% reduction versus TASK assigning 1024 keys. When 16 nodes are colluding, SKAIT with 1032 keys assigned gives a 66.3% reduction in lost channels versus TASK assigning 1024 keys; when 32 nodes are colluding, SKAIT provides a 58.4% reduction. As the number of colluding nodes increases, TASK eventually has fewer lost channels than SKAIT. When approximately 512 keys are assigned

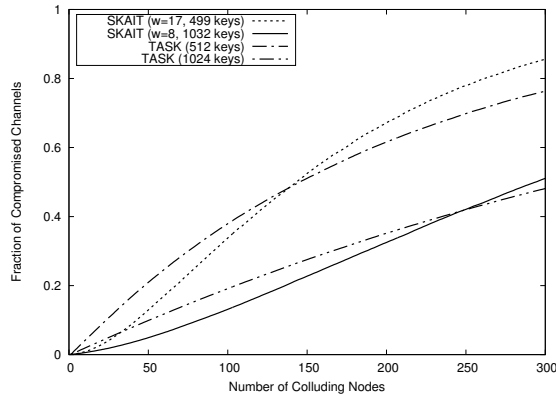


Figure 7. Collusion Resistance (4096 nodes)

per node, this crossover occurs at 138 colluding nodes, with 48.2% of channels lost; when approximately 1024 keys are assigned, the crossover occurs at 248 colluding nodes, with 41.8% of channels lost. Clearly, when smaller numbers of nodes are colluding, SKAIT uses larger key storage capacities more effectively than TASK.

V. CONCLUSION AND FUTURE WORK

We have presented SKAIT, a novel symmetric key pre-distribution scheme that guarantees secure and confidential communication channels between all nodes in a wireless network. Its parameterization allows users to adjust the number of keys assigned to a node, and thus trade-off space complexity and collusion resistance. SKAIT is focused on utilizing key storage capacity in the range from $3\sqrt{n}$ to $\frac{2n}{3} + 3$, and in that range it provides collusion resistance superior to that of the family of protocols proposed by Aiyer, *et al.* SKAIT can also make use of key storage capacity that the Aiyer, *et al.* protocols cannot. For lower numbers of colluding nodes, SKAIT also provides collusion resistance superior to that of the TASK scheme, especially for key capacities towards the upper end of the target range.

In the future, we would like to explore ways to extend SKAIT so that it can be used for key capacities smaller than $3\sqrt{n}$. Preliminary results for a recursive version of SKAIT are encouraging; under certain conditions it achieves collusion resistance similar to TASK and the Aiyer, *et al.* scheme, while using significantly fewer keys per node.

REFERENCES

- [1] J. Kong, P. Zefros, H. Luo, S. Lu, and L. Zhang, "Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks," in *Proceedings of the 9th International Conference on Network Protocols*, Nov. 2001, pp. 251–260.
- [2] V. Varadharajan and Y. Mu, "Design of Secure End-to-End Protocols for Mobile Systems," in *Proceedings of the IFIP World Conference on Mobile Communications*, Canberra, Australia, Sep. 1996, pp. 258–266.
- [3] J. Hubaux, L. Buttyán, and S. Capkun, "The Quest for Security in Mobile Ad-Hoc Networks," in *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Oct. 2001, pp. 4–5.
- [4] M. Tatebayashi, N. Matsuzaki, and D. B. Newman, "Key Distribution Protocol for Digital Mobile Communication Systems," in *Advances in Cryptology—Proceedings of the Annual International Cryptology Conference (CRYPTO)*, 1989, pp. 324–334.
- [5] L. Eschenauer and V. Gligor, "A Key Management Scheme for Distributed Sensor Networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, Nov. 2002, pp. 41–47.
- [6] A. Perrig, R. Szewczyk, V. Wen, D. Cullar, and J. D. Tygar, "SPINS: Security Protocols for Sensor Networks," in *Proceedings of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, Jul. 2001, pp. 189–199.
- [7] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A Pairwise Key Pre-distribution Scheme for Sensor Networks using Deployment Knowledge," *IEEE Transactions on Dependable and Secure Computing*, vol. 3, no. 2, pp. 62–77, 2006.
- [8] D. Liu, P. Ning, and W. Du, "Group-Based Key Predistribution for Wireless Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 4, no. 2, pp. 1–30, 2008.
- [9] L. Gong and D. J. Wheeler, "A Matrix Key-Distribution Scheme," *Journal of Cryptology: The Journal of the International Association for Cryptologic Research*, vol. 2, no. 1, pp. 51–59, 1990.
- [10] S. S. Kulkarni, M. G. Gouda, and A. Arora, "Secret Instantiation in Ad-Hoc Networks," *Journal of Computer Communications*, vol. 29, no. 2, pp. 200–215, Jan. 2006.
- [11] A. S. Aiyer, L. Alvisi, and M. G. Gouda, "Key Grids: A Protocol Family for Assigning Symmetric Keys," in *Proceedings of the 14th International Conference on Network Protocols*, Santa Barbara, California, USA, Nov. 2006, pp. 178–186.
- [12] E. S. Elmallah, M. G. Gouda, and S. S. Kulkarni, "Logarithmic Keying," *ACM Transactions on Autonomic and Adaptive Systems (TAAS)*, vol. 3, no. 4, pp. 1–18, 2008.
- [13] N. Mittal, "Space-Efficient Keying in Wireless Communication Networks," in *Proceedings of the 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, New York, USA, Oct. 2007.
- [14] R. Novales and N. Mittal, "TASK: Template-Based Key Assignment for Confidential Communication in Wireless Networks," in *Proceedings of the Symposium on Reliable Distributed Systems (SRDS)*, Sep. 2009.
- [15] S. A. Camtepe and B. Yener, "Key Distribution Mechanisms for Wireless Sensor Networks: A Survey," Rensselaer Polytechnic Institute, Computer Science Department, Tech. Rep. TR-05-07, 2005.