A Metric for Quantifying Vulnerability in Wireless Sensor Networks

Andrew Clark, Radha Poovendran
Network Security Lab (NSL)
Electrical Engineering Department, University of Washington
Seattle, Washington
Email: {awclark, rp3}@u.washington.edu

***Abstract*—Wireless sensor networks are often used in applications where message confidentiality, integrity, and authentication are required. Cryptography is a common mechanism for meeting these security requirements. The use of cryptography requires that nodes share secret keys, which are typically assigned to each node according to *key distribution schemes*. Given a key distribution, there is currently no design metric for evaluating vulnerability to key exposure. In this work, we introduce a metric for analyzing and comparing the resilience of key distributions by introducing the concept of a *Link Key Security Metric* (LKSM). We define the properties needed in an LKSM and provide a metric that satisfies them.**

I. Introduction

Wireless sensor networks (WSN) are used for remote sensing and aggregation of data, and are characterized by tight resource constraints (energy, computation, and communication), long lifetime, and unattended deployment with little supporting infrastructure. In applications where sensitive information is collected, such as troop movements, this makes WSN likely targets for adversarial attack. These properties have motivated the creation of new security protocols designed specifically for sensor networks [1].

Certain aspects of WSN security, such as information confidentiality and integrity, are typically addressed through cryptography. In order to use cryptographic primitives, however, sensor nodes must first be given cryptographic keys. This is often done by loading keys onto each node before deployment. Mechanisms for carrying this out are called *key distribution schemes*.

As an example of a key distribution scheme, suppose that each sensor node is given the same secret key, which is used to encrypt all communications. This scheme allows for data encryption with minimal storage overhead and key management. It quickly breaks down, however, as the adversary must only acquire one key (e.g., through cryptanalysis or physical capture of a node) to expose all network traffic. One possible solution to this problem is to give every pair of sensor nodes a unique key. Under this scheme, the effect of compromising one link or node is greatly reduced, but at the cost of high storage overhead ($O(n)$ for each node, and $O(n^2)$ for the entire network). Several efficient key distribution schemes, which lie between these two extremes and trade off security, connectivity, and storage, have been proposed [2]–[4].

This work was supported in part by the following grants: ARL CTA, DAAD19-01-2-0011; ARO MURI, W911NF-07-1-0287; and ARO PECASE W911NF-05-1-0491.

For a network with given resource constraints, there may be more than one possible key distribution scheme. In this case, resilience to key exposure adds another dimension to the network design; when choosing schemes that meet performance and connectivity requirements, it is desirable to select the key distribution scheme(s) with the highest security level. The design decision is currently not feasible due to the lack of a metric that quantifies vulnerability to key exposure. In addition, after deployment, some nodes or links may be compromised by the adversary, exposing their keys and making it necessary to reevaluate the security of the rest of the network. A metric that assigns a security level to each link based on the distribution of the remaining (non-exposed) keys is important for this evaluation process. However, no such quantitative measure currently exists.

In this work, we propose a metric for quantifying resilience to key exposure, both during the design stage and during post-deployment analysis.

Our Contributions: In this work, we introduce the concept of a *Link Key Security Metric* (LKSM), which assigns a security level to each communication link in the network based on the key distribution. We define properties that characterize an LKSM, and then present a metric satisfying these properties. We provide an exact algorithm for computing our metric, as well as a fast approximation algorithm suitable for larger networks, and show through analysis and simulation that the approximation can be performed efficiently and with low approximation error.

Our proposed metric makes no assumption about the adversary's strategy, and is instead based on the frequency of key reuse in the key distribution itself.

A. Related Work

The seminal work in key distribution in WSN is due to Eschenauer and Gligor [2]. The authors proposed a *probabilistic* key distribution scheme, in which a pool of keys is generated and each node is given a randomly chosen subset of the key pool. The authors were also the first to recognize the potential problem of key compromise in pooled key distribution schemes.

Following this work, several heuristic studies, based on random and intelligent node capture attacks, were performed [3]–[5] with the aim of mitigating the key compromise problem. Of particular interest is the q-composite key scheme of [3], in which each pair of sensor nodes determines all of their shared keys and then uses a hash of q of these keys as the encryption key. On average, this is expected to increase the cost of eavesdropping by increasing the number of keys the adversary must compromise.

The rest of the paper is organized as follows. In Section II, we describe our network model and problem statement, including a rigorous definition of what a link security metric means. In Section III, we present our metric, prove that it satisfies the definition in the previous section, and demonstrate how to compute it. In Section IV, we provide an approximate algorithm better suited for large networks. In Section V, we show via simulation that it is feasible to compute this metric for every link in the network. In Section VI, we present our conclusions and suggest directions for future work in this area.

TABLE I
NOTATION USED IN THIS PAPER

Notation	Definition
$\mathcal{N}$	Set of nodes
$\mathcal{K}$	Set of keys
$\mathcal{K}_i$	Set of keys held by node i
$\mathcal{N}_k$	Set of nodes holding key k
$\mathcal{K}_{ij}$	Set of keys securing the link (i,j); equal to $\mathcal{K}_i \cap \mathcal{K}_j$
n	Number of nodes
m	Number of keys assigned to each node
P	Number of keys in key pool
E	Set of communication links
$\mathcal{P}(\mathcal{S})$	The set of all subsets of $\mathcal{S}$, where $\mathcal{S}$ is a set
ϕ	Link security metric (defined below)

II. PROBLEM STATEMENT

A. Network Model

We assume a network of n nodes. Each node has a unique ID chosen from the set $\{1,\ldots,n\}$. The nodes are deployed over a certain region, with each node only capable of communicating with other nodes within its radio transmit range. We assume no additional infrastructure, such as base stations. Sensor nodes have limited computational resources and battery power; in particular, the nodes are incapable of public-key encryption. The network topology may or may not be known in advance by the system designer, and individual nodes have no knowledge of the network beyond their immediate neighbors.

We assume that each node i receives a set of keys $\mathcal{K}_i$. Two nodes can exchange messages if and only if they are within radio range of each other and their key sets overlap ($\mathcal{K}_i \cap \mathcal{K}_j \neq \emptyset$). We define $\mathcal{K}_{ij} \triangleq \mathcal{K}_i \cap \mathcal{K}_j$. When two nodes communicate, their encryption key is a hash of all the keys in $\mathcal{K}_{ij}$. The metrics and definitions described in this paper can be extended to other key agreement schemes, however. We leave this as a topic for future work.

B. Adversary Model

We assume that the network is deployed in the presence of an adversary. The adversary may have several different goals: he may wish to eavesdrop on sensitive information, inject false packets, or carry out flooding or other denial of service attacks. Each of these goals is made possible by capturing cryptographic keys. In addition to eavesdropping capability, we assume that the adversary has full knowledge of the key IDs held by each node (although not the keys themselves), and that the adversary knows the network and routing topology. The adversary is also capable of capturing nodes, thereby gaining access to all the node's cryptographic keys. Since all keys are drawn from a common key pool, these keys may be used to secure other links in the network. Furthermore, although the adversary can capture nodes, he only has a limited capacity to do so, and therefore must intelligently choose which nodes provide the most benefit.

C. Properties of Link Key Security Metric

Before defining the properties of a link key security metric, we introduce the concept of a redundant key.

Definition 1: A key k is *redundant* to link (i,j) if there exists a key $k' \in \mathcal{K}_{ij} \setminus \{k\}$ satisfying $\mathcal{N}_{k'} \subset \mathcal{N}_k$.

Since $\mathcal{N}_{k'} \subset \mathcal{N}_k$, capturing key k' implies the capture of key k. Thus k does not contribute anything to the security of the link, since capturing the other keys securing the link will automatically result in the compromise of key k.

We define a link key security metric as follows.

Definition 2: A *link key security metric* (LKSM) is a function f that takes as input the tuple $((i,j),\mathcal{K}_1,\ldots,\mathcal{K}_n)$, where (i,j) is a communication link in the network, and outputs a positive real number. f satisfies the following properties:

1) For any link (i,j) and any key $k \notin \mathcal{K}_{ij}$, we have that
$$f((i,j),\mathcal{K}_1,\ldots,\mathcal{K}_i,\ldots,\mathcal{K}_j,\ldots,\mathcal{K}_n) \leq f((i,j),\mathcal{K}_1,\ldots,\mathcal{K}_i \cup \{k\},\ldots,\mathcal{K}_j \cup \{k\},\ldots,\mathcal{K}_n) \quad (1)$$
We have equality if and only if k is redundant to (i,j).
2) For any link (i,j), any key $k \in \mathcal{K}_{ij}$, and any node a with $k \notin \mathcal{K}_a$, we have that
$$f((i,j),\mathcal{K}_1,\ldots,\mathcal{K}_a \cup \{k\},\ldots,\mathcal{K}_n) \leq f((i,j),\mathcal{K}_1,\ldots,\mathcal{K}_a,\ldots,\mathcal{K}_n) \quad (2)$$
with equality if and only if k is redundant to (i,j).

The first property states that adding a key to a link cannot decrease the security of the link. This is because the link retains its original keys, and the security of those keys is not reduced simply by adding a new key to the link. Adding a redundant key k does not change the security of the link, since any set of node captures by the adversary that recovers all of $\mathcal{K}_{ij}$ will also recover $\mathcal{K}_{ij} \cup \{k\}$.

The second property states that, if a node is given one of the keys comprising a link, then the security metric of the link should decrease. Adding a key to a node elsewhere in the network weakens that key, and hence the link as a whole, by providing another option for the adversary to capture that key.

These properties are motivated by the adversary model. Since the adversary knows the key distribution, the adversary's attacks will be based on the frequency of key reuse in the network. The first property reflects the fact that links secured by relatively few keys will be more vulnerable to attack. Furthermore, keys that are held by many nodes will be quickly exposed in an attack, and therefore links that rely on these keys will have limited security, motivating the second property.

In the next section, we present our metric and prove that it satisfies the desired properties.

III. Proposed Link Key Security Metric

Our proposed LKSM is defined as follows. Suppose we choose a node uniformly at random from $\mathcal{N}$, add its keys to a list of keys found so far, and then repeat the process until all the keys in $\mathcal{K}_{ij}$ have been found. The metric is defined as the expected number of nodes collected in this way until $\mathcal{K}_{ij}$ has been completely recovered. We state the definition more formally here.

Definition 3: Let $\{N_1, N_2, \ldots\}$ be a sequence of independent, identically distributed random variables, each of which is uniformly distributed on the set $\{1, \ldots, n\}$. We then define, for each $k \in \mathcal{K}$ and $(i,j) \in E$,

$$T_k = \min\{l : k \in \mathcal{K}_{N_l}\} \tag{3}$$

$$T_{ij} = \max_{k \in \mathcal{K}_{ij}} T_k, \phi(i,j) \triangleq \mathbf{E}(T_{ij}) \tag{4}$$

$\phi(i,j)$ is the value assigned to link (i,j) by our LKSM.

Definition 3 has the following interpretation. At each time step, we collect one node uniformly at random and add its keys to a set of captured keys. For any key k, the random variable T_k defined in (3) represents the number of time steps until k has been captured. Since the link is not captured until all keys have been captured, T_{ij}, the time to capture (i,j) is defined as the maximum value of T_k over all $k \in \mathcal{K}_{ij}$. We assign $\phi(i,j)$ to be the expected capture time.

This definition quantifies the security of the keys used by each link. If a key is held by relatively few nodes (and is therefore more secure), it will on average take more random collections until a node holding the key is found. Adding a key in $\mathcal{K}_{ij}$ to another node decreases the expected time, and therefore the security, by creating more opportunities for collecting that key. Adding a new key to $\mathcal{K}_{ij}$ will increase the expected time by increasing the number of keys that must be collected.

A. Verification of LKSM Properties

We now prove that the metric ϕ in Definition 3 satisfies conditions 1 and 2 in Definition 2.

Lemma 1 (First Property of Link Metric): For any link (i,j) and any key $k \notin \mathcal{K}_{ij}$, we have that

$$\begin{aligned}\phi((i,j), \mathcal{K}_1, \ldots, \mathcal{K}_i, \ldots, \mathcal{K}_j, \ldots, \mathcal{K}_n) \leq \\ \phi((i,j), \mathcal{K}_1, \ldots, \mathcal{K}_i \cup \{k\}, \ldots \mathcal{K}_j \cup \{k\}, \ldots, \mathcal{K}_n)\end{aligned} \tag{5}$$

with equality if and only if $\mathcal{N}_{k'} \subset \mathcal{N}_k$ for some $k' \in \mathcal{K}_{ij}$. In other words, ϕ satisfies the first condition of Definition 2.

Proof: Let $\mathcal{K}_{ij} = \{k_1, \ldots, k_p\}$, and let $\{N_1, N_2, \ldots\}$ be defined as in Definition 3. Define $T = \max\{T_{k_1}, \ldots, T_{k_p}\}$, the time to collect all the keys in $\mathcal{K}_{ij}$. Furthermore, define $\mathcal{K}'_{ij} = \mathcal{K}_{ij} \cup \{k\}$ and $T' = \max\{T_{k_1}, \ldots, T_{k_p}, T_k\}$. Equivalently, we have $T' = \max\{T, T_k\}$, and as a result $T \leq T'$. We now have two cases, the case where k is redundant and the case where k is non-redundant.

<u>*Case 1:*</u> If $\mathcal{N}_{k_l} \subset \mathcal{N}_k$ for some l, then we have $T_k \leq T_{k_l}$. This is because any node holding k_l will also hold key k, and therefore it cannot take more time to collect k. Thus $\max\{T, T_k\} = T$, and so $T = T'$, which implies that $\mathbf{E}(T) = \mathbf{E}(T')$.

<u>*Case 2:*</u> Suppose that for every $l \in \{1, \ldots, p\}$, there exists $n_l \in \mathcal{N}_{k_l} \setminus \mathcal{N}_k$. Define event A by

$$A \triangleq \{N_1 = n_1, \ldots, N_p = n_p\} \tag{6}$$

Since every sequence of node collections is feasible, A occurs with nonzero probability. When A occurs, all of the keys in $\mathcal{K}_{ij}$ have been collected within the first p rounds, but k has not yet been collected. This gives[1]

$$T'\mathbf{1}_A > T\mathbf{1}_A. \tag{7}$$

We have

$$\begin{aligned}\mathbf{E}(T) &= \mathbf{E}(T\mathbf{1}_A) + \mathbf{E}(T\mathbf{1}_{A^c}) &\text{(8a)}\\ &\leq \mathbf{E}(T\mathbf{1}_A) + \mathbf{E}(T'\mathbf{1}_{A^c}) &\text{(8b)}\\ &< \mathbf{E}(T'\mathbf{1}_A) + \mathbf{E}(T'\mathbf{1}_{A^c}) &\text{(8c)}\\ &= \mathbf{E}(T') &\text{(8d)}\end{aligned}$$

Inequality (8b) comes from the fact that $T \leq T'$, while the strict inequality (8c) follows from (7). Hence we have $\mathbf{E}(T)$ strictly less than $\mathbf{E}(T')$ if and only if k is a non-redundant key. ■

Lemma 2 (Second Property of Link Metric): For any link (i,j), any key $k \in \mathcal{K}_{ij}$, and any node a with $k \notin \mathcal{K}_a$, we have that

$$\begin{aligned}\phi((i,j), \mathcal{K}_1, \ldots, \mathcal{K}_a \cup \{k\}, \ldots, \mathcal{K}_n) \leq \\ \phi((i,j), \mathcal{K}_1, \ldots, \mathcal{K}_a, \ldots, \mathcal{K}_n)\end{aligned} \tag{9}$$

with equality if and only if there exists $k' \in \mathcal{K}_{ij}$ with $\mathcal{N}_{k'} \subset \mathcal{N}_k$. In other words, ϕ satisfies the second condition of Definition 2.

Proof: Let $\mathcal{K}_{ij} = \{k_1, \ldots, k_p\}$, and assume without loss of generality that $k = k_p$. Define $T = \max\{T_{k_1}, \ldots, T_{k_p}\}$ to be the time to collect all keys in $\mathcal{K}_{ij}$. Now, suppose that we add key k to node a. Let $T' = \max\{T_{k_1}, \ldots, T_{k_p}\}$ when $k \in \mathcal{K}_a$.

First, we have $T' \leq T$. This is because, since no keys have been removed from any nodes, any sequence $(N_1, \ldots, N_T)$ that collects all keys when $k \notin \mathcal{K}_a$ will also collect all keys when $k \in \mathcal{K}_a$. Now, depending on whether k is redundant, we have two cases.

[1] $\mathbf{1}_A$ is the indicator function of event A, i.e. it is 1 when A occurs and 0 otherwise.

Case 1: If $\mathcal{N}_{k_l} \subset \mathcal{N}_k$ for some $l \neq p$, then $T_k = T_{k_p} \leq T_{k_l}$, and so $T = \max\{T_{k_1}, \ldots, T_{k_{p-1}}, T_{k_p}\} = \max\{T_{k_1}, \ldots, T_{k_{p-1}}\}$. Furthermore, $\mathcal{N}_{k_l} \subset (\mathcal{N}_k \cup \{a\})$, and therefore $T' = \max\{T_{k_1}, \ldots, T_{k_{p-1}}\}$ as well. Since the sets $\mathcal{N}_{k_l}$ for $l < p$ are unchanged, this implies that $T = T'$.

Case 2: Suppose that, for every $l < p$, there exists $n_l \in \mathcal{N}_{k_l} \setminus \mathcal{N}_{k_p}$. Define the event A to be

$$A \triangleq \{N_1 = n_1, \ldots, N_{p-1} = n_{p-1}, N_p = a\} \tag{10}$$

Now, if $a \in \mathcal{N}_{k_p}$, then all keys in $\mathcal{K}_{ij}$ have been collected. Otherwise, additional nodes must be gathered, and so

$$T\mathbf{1}_A > T'\mathbf{1}_A \tag{11}$$

Since $T' \leq T$ in general, this equation implies that $\mathbf{E}(T') \leq \mathbf{E}(T)$. ■

B. Computation of Proposed LKSM

We can find the expected time to recover all keys securing a link (i, j) through a random walk formulation. Consider the set $\mathcal{P}(\mathcal{K}_{ij})$, the set of subsets of $\mathcal{K}_{ij}$. We express our metric in terms of a random walk on $\mathcal{P}(\mathcal{K}_{ij})$. After l steps, the current state of the walk will be the set $\mathcal{S}$ given by

$$\mathcal{S} = \bigcup_{r=1}^{l} (\mathcal{K}_{N_r} \cap \mathcal{K}_{ij}), \tag{12}$$

the set of keys in $\mathcal{K}_{ij}$ held by nodes $N_1, \ldots, N_l$. At step $l+1$, a transition from $\mathcal{S}$ to $\mathcal{T}$ occurs if N_{l+1} is in the set $\mathcal{N}(\mathcal{S}, \mathcal{T})$, defined by

$$\mathcal{N}(\mathcal{S}, \mathcal{T}) \triangleq \{a \in \mathcal{N} : \mathcal{K}_a \cap \mathcal{K}_{ij} = \mathcal{T} \setminus \mathcal{S}\} \tag{13}$$

I.e., if N_{l+1} contains the keys in $\mathcal{T} \setminus \mathcal{S}$ but no additional keys in $\mathcal{K}_{ij} \setminus \mathcal{S}$, which would cause a transition to a different state. Since nodes are collected uniformly at random, the transition probability is given by $\frac{|\mathcal{N}(\mathcal{S},\mathcal{T})|}{|\mathcal{N}|}$.

If we start our walk at the set $\emptyset$, where no keys have been collected yet, then the value of our proposed LKSM will be equal to the expected number of steps before the walk reaches state $\mathcal{K}_{ij}$, the state in which all keys have been found.

Let P_{st} be the probability of a transition from state s to state t, and let H_{st} be the expected number of steps in a random walk starting at s before state t is attained. Then we have

$$H_{st} = 1 + \sum_{k \neq s} P_{sk} H_{kt} + P_{ss} H_{st} \tag{14}$$

Gathering terms, we then have

$$H_{st} = \frac{1 + \sum_{k \neq s} P_{sk} H_{kt}}{1 - P_{ss}} \tag{15}$$

Our goal is to compute H_{st} when state s is $\emptyset$ and state t is $\mathcal{K}_{ij}$. In the worst case, this will require computing the time to reach state $\mathcal{K}_{ij}$ from all other possible states. Although we can find H_{st} by solving a matrix equation, computation time is reduced by taking advantage of the structure of the transition matrix. Suppose we are at the state corresponding to set $\mathcal{S}$. Since collecting additional nodes can only add to the list of keys found, it is only possible to transition to states with greater cardinality than $\mathcal{S}$, or to $\mathcal{S}$ itself if no additional keys are found. We can therefore compute the expected times of all states $\mathcal{S}$ satisfying $|\mathcal{S}| = |\mathcal{K}_{ij}| - 1$ first. Using this information, we can then compute the expected times of all sets satisfying $|\mathcal{S}| = |\mathcal{K}_{ij}| - 2$, and so on, until we can compute the expected time starting from state $\emptyset$. An algorithmic description can be found in Table II.

TABLE II
ALGORITHM FOR EXACT COMPUTATION

```
Input: Key distribution $\mathcal{K}_1, \ldots, \mathcal{K}_n$.
for all $(i,j) \in E$
  $\mathcal{K}_{ij} \leftarrow \mathcal{K}_i \cap \mathcal{K}_j$
  $s \leftarrow 0$
  while $s \leq |\mathcal{K}_{ij}|$
    for all $\mathcal{S} \subset \mathcal{K}_{ij}$, $|\mathcal{S}| = s$
      for all $\mathcal{T} \subset \mathcal{K}_{ij}$, $\mathcal{T} \supseteq \mathcal{S}$
        $P(\mathcal{S},\mathcal{T}) \leftarrow \frac{1}{|\mathcal{N}|} |\bigcap_{k \in \mathcal{T} \setminus \mathcal{S}} \mathcal{N}_k|$
    $s \leftarrow s + 1$
  $q \leftarrow |\mathcal{K}_{ij}|$
  while $q \geq 0$
    for all $\mathcal{S} \subset \mathcal{K}_{ij}$, $|\mathcal{S}| = q$
      $H(\mathcal{S},\mathcal{K}_{ij}) \leftarrow \frac{1}{1-P(\mathcal{S},\mathcal{S})} \left(1 + \sum_{\mathcal{S} \subsetneq T} P(\mathcal{S},\mathcal{T}) H(\mathcal{T},\mathcal{K}_{ij})\right)$
    $q \leftarrow q - 1$
  $\phi(i,j) \leftarrow H(\emptyset, \mathcal{K}_{ij})$
```

C. Asymptotic Runtime Analysis

We now derive the computation time required to evaluate the metric. For each edge, we first find the state transition matrix and then the relevant values of H_{st}. In order to find the state transition matrix, we must find the number of nodes holding all the keys in $\mathcal{T} \setminus \mathcal{S}$ for all $\mathcal{S}, \mathcal{T} \subseteq \mathcal{K}_{ij}$ satisfying $\mathcal{S} \subseteq \mathcal{T}$. If $|\mathcal{S}| = \ell$, there are $2^{|\mathcal{K}_{ij}| - \ell}$ sets containing $\mathcal{S}$. Furthermore, for every ℓ, there are $\binom{|\mathcal{K}_{ij}|}{\ell}$ subsets of size ℓ. Thus the time to find the transition matrix is equal to

$$O\left(\sum_{\ell=0}^{|\mathcal{K}_{ij}|} \left[\binom{|\mathcal{K}_{ij}|}{\ell} 2^{|\mathcal{K}_{ij}| - \ell}\right]\right) \tag{16}$$

In the worst case, we need to find the expected time to reach $\mathcal{K}_{ij}$ from every state $\mathcal{S} \subseteq \mathcal{K}_{ij}$. The expected time of each state depends on the expected times of the states containing it, so the workload is

$$O\left(\sum_{\ell=0}^{|\mathcal{K}_{ij}|} \left[\binom{|\mathcal{K}_{ij}|}{\ell} 2^{|\mathcal{K}_{ij}| - \ell}\right]\right) \tag{17}$$

This must be done for every link, giving an overall workload of

$$O\left(\sum_{(i,j) \in E} \left(\sum_{\ell=0}^{|\mathcal{K}_{ij}|} \left[\binom{|\mathcal{K}_{ij}|}{\ell} 2^{|\mathcal{K}_{ij}| - \ell}\right]\right)\right) \tag{18}$$

When $|\mathcal{K}_{ij}|$ is not very large on average, this computation is still feasible. For larger values of $|\mathcal{K}_{ij}|$, however, the total computation time becomes prohibitively large. To avoid this problem, we have developed an algorithm that approximates our LKSM.

IV. Approximate Computation of LKSM

We introduce the following non-exponential approximation algorithm to compute the LKSM values. We first select a fixed number of trials L and the number of rounds per trial, r. During the j-th round of trial l, we select a value at random from $\{1, \ldots, n\}$ and examine the keys held by the node with that index. For each key k, if k has not yet been collected, then we set $T(l,k)$ (the number of rounds before key k is collected in trial l) equal to j. If k is not collected during the r rounds, we set $T(l,k) = r$ as an approximation. r should be chosen large enough to make this unlikely.

After the number of rounds reaches r, we move on to the next stage. For every link (i,j), we define the approximate metric value for link (i,j) based on round l to be $T_{ij}^{(l)} \triangleq \max_{k \in \mathcal{K}_{ij}} T(l,k)$. Our estimate of the metric is then the average of the approximations computed in each round, given by

$$T'_{ij} \triangleq \frac{1}{L} \sum_{l=1}^{L} T_{ij}^{(l)} \tag{19}$$

This algorithm can be found in Table III.

Each trial represents a sample path of the random variables $\{N_1, N_2, \ldots\}$ defined above, and so also represents a sample path of the random variables $\{T_k\}_{k \in \mathcal{K}}$. Since we are approximating an infinite sequence of node collections, the number of rounds r must be sufficiently large so that every key used in the network is gathered. In our simulations, we found that choosing $r = O(n)$ leads to an accurate approximation. By averaging over a sufficiently large number of trials L, we get an estimate of $\mathbf{E}(T_{ij})$ for all i and j (see the error analysis provided below).

TABLE III
Approximate LKSM Algorithm

```
Input: Key distribution $\mathcal{K}_1, \ldots, \mathcal{K}_n$
$T(i,k) \leftarrow r,\ i = 1, \ldots, L,\ k = 1, \ldots, |\mathcal{K}|$
for $l = 1, \ldots, L$
  for $j = 1, \ldots, r$
    $s \overset{R}{\leftarrow} \{1, \ldots, n\}$
    for $k \in \mathcal{K}_s$
      if $T(l,k)$ equals $r$
        $T(l,k) \leftarrow j$

for each $(i,j) \in E$
  for $l = 1, \ldots, L$
    $V(l) \leftarrow \max_{k \in \mathcal{K}_{ij}} T(l,k)$
  $\phi(i,j) \leftarrow$ mean of $V(l)$
```

A. Asymptotic Runtime Analysis

During each trial, we generate $r = O(n)$ random numbers, each representing a node index. For each of these indices, we update the collection times for this trial. Since there are L trials, the total workload is $O(Lnm)$.

After this phase is complete, we then estimate T_{ij} for every link (i,j). The workload for link (i,j) is $O(L|\mathcal{K}_{ij}|)$, for an overall computation time of

$$O\left(L\left(nm + \sum_{(i,j)\in E} |\mathcal{K}_{ij}|\right)\right) \tag{20}$$

B. Approximation Error

Since the approximation algorithm does not actually find the expected time but rather estimates it via Monte Carlo methods, it will produce some approximation error, which we analyze in this section. The approximation algorithm conducts a series of L independent trials, each trial representing a different sample path of the sequence $\{N_1, N_2, \ldots\}$ in Definition 3. For each link (i,j), trial l gives a random variable $T_{ij}^{(l)}$, which is that trial's value of T_{ij}. The output of the algorithm is an overall approximation T'_{ij}, defined above in (19). We assume that r, the number of rounds, is sufficiently large so that $T_{ij}^{(l)}$ has the same distriubtion as T_{ij}. The Central Limit Theorem tells us that, for L sufficiently large, the distribution of T'_{ij} converges to $N(\mu, \frac{\sigma^2}{L})$, where μ and σ^2 are the mean and variance of T_{ij}, respectively. Since we are trying to estimate $\mathbf{E}(T_{ij})$, the variance $\sigma_e^2 = \frac{\sigma^2}{L}$ will determine the error of the estimate. From the properties of the normal distribution, T'_{ij} will be within σ_e of $\mathbf{E}(T_{ij})$ with 95% probability. Therefore we treat σ_e as a bound on the absolute error of the estimate.

To simplify the analysis, we assume that the $T_{ij}^{(l)}$'s have a geometric distribution with mean $\mathbf{E}(T_{ij})$. This is true when $\mathcal{K}_{ij}$ consists of a single non-redundant key, and it is a reasonable approximation in general because $T_{ij}^{(l)}$ represents the number of independent trials that occur before a condition is satisfied. For a geometric random variable with mean μ, the variance is given by $\mu^2 - \mu$. We then have

$$\sigma_e = \frac{\sqrt{\mu^2 - \mu}}{\sqrt{L}} < \frac{\sqrt{\mu^2}}{\sqrt{L}} = \frac{\mu}{\sqrt{L}} \tag{21}$$

The relative error

$$\frac{|T'_{ij} - \mathbf{E}(T_{ij})|}{\mathbf{E}(T_{ij})} \tag{22}$$

is therefore bounded above, with 95% probability, by $\frac{\mu}{\mu\sqrt{L}} = \frac{1}{\sqrt{L}}$. We therefore have that the relative error is determined only by the number of trials and is independent of the network parameters, implying that the approximation error scales well with network size. These results agree with the simulation data supplied below.

V. Simulation Study

In order to demonstrate the feasibility of the proposed LKSM, the run time of the approximation algorithm and the deviation from ideal behavior were simulated. The results are presented below.

A. Simulation setup

A network of n nodes, distributed uniformly at random within a fixed deployment area, was simulated for different values of the parameters specified in Table IV. The deployment area was a square of area 1, and the node radio range was equal

TABLE IV
LIST OF SIMULATION PARAMETERS

Parameter	Description	Setting in runtime analysis	Setting in error analysis
L	Number of trials	Fixed at $L = 250$	Varies
n	Number of nodes	Varies	Fixed at $n = 200$
r	Number of rounds/trial	Fixed at $r = n$	Fixed at $r = n$
P	Size of key pool	Varies	Varies
m	Size of each node's set of keys	Fixed at $m = 30$	Fixed at $m = 30$

to 0.15. Keys were distributed according to the probabilistic key distribution scheme [2]. In each case, the LKSM defined in Section III was computed for each communication link in the network.

Simulations were implemented using Matlab on a computer with an Intel Core2 Duo 3GHz processor and 3.25GB of RAM.

B. Runtime of Approximate Algorithm

The run time (in seconds) of the approximate algorithm is presented below (Fig. 1) for various network sizes. The run time represents the average amount of time to calculate the metric for every link in the network. The key pool size was varied as a way of changing the average size of $|\mathcal{K}_{ij}|$, since a smaller key pool will lead to more overlap. Even for a relatively large network with a great deal of key overlap, it only takes on the order of one minute to compute the metric for the entire network on a desktop PC.

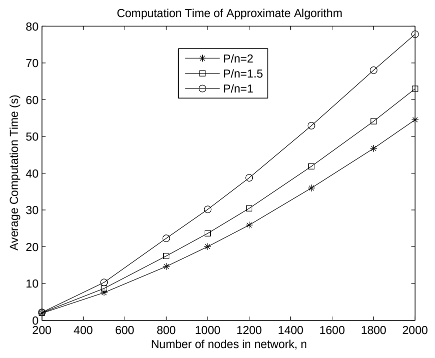

Fig. 1. Run time of approximate algorithm

C. Error of Approximate Algorithm

The following plot (Fig. 2) shows the average percentage error of the calculations produced by the approximate algorithm. The difference between approximate and actual drops off quickly, making the approximate algorithm both accurate and computationally feasible.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we proposed the concept of a *link key security metric* that can be used to evaluate the resilience of key distribution schemes to adversarial attack. We presented one feasible LKSM, along with two algorithms for computing it. The first algorithm computes the exact value of our metric and is suitable for smaller networks or networks with limited key reuse. The second algorithm produces an approximate result and is designed for large networks with more key overlap.

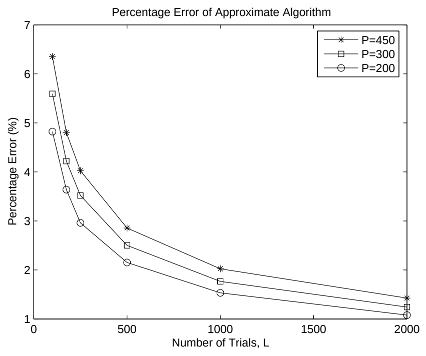

Fig. 2. Error of approximate algorithm

These results are part of a broader goal of developing security metrics that can be considered jointly with performance metrics such as throughput, error rate, and latency. Thus the main goals of future work will be to develop metrics for other adversarial attacks, such as jamming, cloning, and attacks on routing schemes, and to extend definitions of link vulnerability to vulnerability of network flows.

Another direction for future work will be to analyze key distribution schemes using this metric, and to use the metric to understand how networks degrade under different kinds of adversarial attack.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE communications magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[2] 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*. ACM New York, NY, USA, 2002, pp. 41–47.

[3] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2003, pp. 197–215.

[4] M. Ramkumar and N. Memon, "An efficient random key pre-distribution scheme for manet security," *IEEE Journal on Selected Areas of Communication*, 2005.

[5] D. Liu, P. Ning, and R. Li, "Establishing pairwise keys in distributed sensor networks," *ACM Transactions on Information and System Security (TISSEC)*, vol. 8, no. 1, pp. 41–77, 2005.