Random Key Predistribution Schemes for Sensor Networks*

Haowen Chan Adrian Perrig Dawn Song
Carnegie Mellon University
{haowenchan, perrig, dawnsong}@cmu.edu

Abstract

Key establishment in sensor networks is a challenging problem because asymmetric key cryptosystems are unsuitable for use in resource constrained sensor nodes, and also because the nodes could be physically compromised by an adversary. We present three new mechanisms for key establishment using the framework of pre-distributing a random set of keys to each node. First, in the q-composite keys scheme, we trade off the unlikeliness of a large-scale network attack in order to significantly strengthen random key predistribution's strength against smaller-scale attacks. Second, in the multipath-reinforcement scheme, we show how to strengthen the security between any two nodes by leveraging the security of other links. Finally, we present the random-pairwise keys scheme, which perfectly preserves the secrecy of the rest of the network when any node is captured, and also enables node-to-node authentication and quorum-based revocation.

1 Introduction

Wide-spread deployment of sensor networks is on the horizon. Networks of thousands of sensors may present an economical solution to some of our challenging problems: real-time traffic monitoring, building safety monitoring (structural, fire, and physical security monitoring), military sensing and tracking, distributed measurement of seismic activity, real-time pollution monitoring, wildlife monitoring, wildfire tracking, etc. Many applications are dependent on the secure operation of a sensor network, and have serious consequences if the network is compromised or disrupted.

In sensor network security, an important challenge is the design of protocols to bootstrap the establishment of a secure communications infrastructure from a collection of sensor nodes which may have been pre-initialized with some secret information but have had no prior direct contact with each other. We refer to this problem as the *bootstrapping problem*. A bootstrapping protocol must not only enable a newly deployed sensor network to initiate a secure infrastructure, but it must also allow nodes deployed at a later time to join the network securely. The difficulty of the bootstrapping problem stems from the numerous limitations of sensor networks. We discuss these limitations in detail in Section 2.2; some of the more important ones include the inability to utilize existing public key cryptosystems (since the expensive computations involved could expose the power-constrained nodes to a denial-of-service attack), the inability to pre-determine which nodes will be neighbors after deployment, and the inability of any node to put absolute trust in its neighbor (since the nodes are not tamper resistant and are vulnerable to physical capture).

Eschenauer and Gligor recently proposed a random key predistribution scheme to address the bootstrapping problem. Its operation is briefly described as follows. A random pool of keys is selected from the key space. Each sensor node receives a random subset of keys from the key pool before deployment. Any two nodes able to find one common key within their respective subsets can use that key as their shared secret to initiate communication [10]. We review their approach (which we call the *basic* random key scheme) in Section 4.

In this paper, we propose three new mechanisms in the framework of random key predistribution to address the bootstrapping problem. First, we propose the *q-composite random key predistribution scheme*, which achieves greatly strengthened security under small scale attack while trading off increased vulnerability in the face of a large scale physical attack on network nodes. We will explain why this trade-off is a desirable one. Second, we present the *multi-path key reinforcement scheme*, which substantially increases the security of key setup such that an attacker has to compromise many more nodes to achieve a high probability of compromising any given communication. Finally, we propose the *random-pairwise keys scheme*, which assures that, even when some number of nodes have been compro-

*We gratefully acknowledge funding support for this research. This work was made possible in part by a gift from Bosch Research. This paper represents the opinions of the authors and does not necessarily represent the opinions or policies, either expressed or implied, of Bosch Research.

mised, the remainder of the network remains fully secure. Furthermore, this scheme enables node-to-node mutual authentication between neighbors and quorum-based node revocation without involving a base station. Node-to-node mutual authentication here refers to the property that any node can ascertain the identity of the nodes that it is communicating with.

To the best of our knowledge, no previous security scheme for sensor networks supports efficient node-to-node authentication without involving a base station. We give a detailed analysis of each proposed scheme and show under which situations our schemes can be used to achieve maximum security.

The remainder of the paper is organized as follows. We describe the problem area and present evaluation criteria for successful bootstrapping protocols in Section 2. We summarize our notation in Section 3. We then give an overview of the basic random key scheme by Eschenauer and Gligor in Section 4. We describe our q-composite random key predistribution scheme in Section 5, and our multi-path key reinforcement scheme in Section 6. We present our third scheme, the random-pairwise keys scheme in Section 7. Finally, we discuss related work in Section 8, and summarize our results in Section 9.

2 Problem statement and evaluation metrics

In this section, we first discuss the topology and architecture of a typical sensor network. We then list the technical properties of typical sensor networks that makes the bootstrapping problem a challenge. Finally, we present the goals and evaluation metrics for a successful sensor network security bootstrapping scheme.

2.1 Sensor network architecture

A typical sensor network has hundreds to several thousand sensor nodes. Each sensor node is typically low-cost, limited in computation and information storage capacity, highly power constrained, and communicates over a short-range wireless network interface. Most sensor networks have a base station that acts as a gateway to associated infrastructure such as data processing computers. Individual sensor nodes communicate locally with neighboring sensors, and send their sensor readings over the peer-to-peer sensor network to the base station. Sensors can be deployed in various ways, such as physical installation of each sensor node, or random aerial scattering from an airplane. In this paper we assume that any sensor network is only deployed by a single party, i.e. sensor nodes deployed by multiple independent untrusted parties are not part of the same network.

Generally, sensor nodes communicate over a wireless network. A typical sensor network forms around one or more *base stations*, which connect the sensor network to the outside network.

The communication patterns within a sensor network fall into three categories: node to node communication (e.g., aggregation of sensor readings), node to base station communication (e.g., sensor readings), base station to node communication (e.g., specific requests).

An example of a sensor node's hardware configuration is the Berkeley Mica Motes. They feature a 8-bit 4 MHz Atmel ATmega 128L processor with 128K bytes program store, and 4K bytes SRAM. The processor only supports a minimal RISC-like instruction set, without support for multiplication or variable-length shifts or rotates. The ISM band radio receiver communicates at a peak rate of 40Kbps at a range of up to 100 feet.

The deployment density and the overall size of the network can vary depending on the application. In this paper, we are examining very large sensor networks (> 1000 nodes) with a sizable communication range (> 20 neighboring nodes within communication range) and possibly multiple base stations. We focus on large networks because they cannot rely on existing non-scalable solutions for small networks such as base-station authentication. Due to their smaller overall statistical variance, they are uniquely suited to the random key approaches that we propose in this paper.

2.2 Sensor network limitations

The following characteristics of sensor networks complicate the design of secure protocols for sensor networks, and make the bootstrapping problem highly challenging. We discuss the origins and implications of each factor in turn.

- *Impracticality of public key cryptosystems.* The limited computation and power resources of sensor nodes often makes it undesirable to use public-key algorithms, such as Diffie-Hellman key agreement [9] or RSA signatures [20]. Currently, a sensor node may require on the order of tens of seconds up to minutes to perform these operations [7, 8]. This exposes a vulnerability to denial of service (DoS) attacks.

- *Vulnerability of nodes to physical capture.* Sensor nodes may be deployed in public or hostile locations (such as public buildings or forward battle areas) in many applications. Furthermore, the large number of nodes that are deployed implies that each sensor node must be low-cost, which makes it difficult for manufacturers to make them tamper-resistant. This exposes sensor nodes to physical attacks by an adversary. In the worst case, an adversary may be able to undetectably take control of a sensor node and compromise the cryptographic keys.

- *Lack of a-priori knowledge of post-deployment configuration.* If a sensor network is deployed via random scattering (e.g. from an airplane), the sensor network protocols cannot know beforehand which nodes will be within communication range of each other after deployment. Even if the nodes are deployed by hand, the large number of nodes involved makes it costly to pre-determine the location of every individual node. Hence, a security protocol should not assume prior knowledge of which nodes will be neighbors in a network.

- *Limited memory resources.* The amount of key-storage memory in a given node is highly constrained; it does not possess the resources to establish unique keys with every one of the other nodes in the network.

- *Limited bandwidth and transmission power.* Typical sensor network platforms have very low bandwidth. For example, the UC Berkeley Mica platform's transmitter has a bandwidth of 10 Kbps, and a packet size of about 30 bytes. Transmission reliability is often low, making the communication of large blocks of data particularly expensive.

- *Over-reliance on base stations exposes vulnerabilities.* In a sensor network, base stations are few and expensive. Hence it may be tempting to rely on them as a source of trust. However, this invites attack on the base station and limits the application of the security protocol.

2.3 The problem of bootstrapping security in sensor networks

Based on the limitations described in Section 2.2, a bootstrapping scheme for sensor networks needs to satisfy the following requirements:

- Deployed nodes must be able to establish secure node-to-node communication.

- The scheme should be functional without involving the base station as an arbiter or verifier.

- Additional legitimate nodes deployed at a later time can form secure connections with already-deployed nodes. This implies that bootstrapping information must always be present and cannot simply be erased after deployment to prevent compromise in the event of capture.

- Unauthorized nodes should not be able to establish communications with network nodes and thus gain entry into the network.

- The scheme must work without prior knowledge of which nodes will come into communication range of each other after deployment.

- The computational and storage requirement of the scheme must be low, and the scheme should be robust to DoS attacks from out-of-network sources.

2.4 Evaluation metrics

Sensor networks have many characteristics that make them more vulnerable to attack than conventional computing equipment. Simply assessing a scheme based on its ability to provide secrecy is insufficient. We present several criteria that represent desirable characteristics in a key-setup scheme for sensor networks.

- *Resilience against node capture*. We assume the adversary can mount a physical attack on a sensor node after it is deployed and read secret information from its memory. We evaluate a scheme's resilience toward node capture by estimating the fraction of total network communications that are compromised by a capture of x nodes *not including* the communications in which the compromised nodes are directly involved.

- *Resistance against node replication*. Whether the adversary can insert additional hostile nodes into the network after obtaining some secret information (e.g. through node capture or infiltration). This is a serious attack since the compromise of even a single node might allow an adversary to populate the network with clones of the captured node to such an extent that legitimate nodes could be outnumbered and the adversary can thus gain full control of the network.

- *Revocation*. Whether a detected misbehaving node can be dynamically removed from the system.

- *Scale*. As the number of nodes in the network grows, the security characteristics mentioned above may be weakened. We give a detailed definition of *maximum supportable network size* in Section 4.2.

Each solution we propose in this paper involves several steps. An *initialization* procedure is performed to initialize sensor nodes before they are deployed. After the sensor nodes are deployed, a *key setup* procedure is performed by the nodes to set up shared secret keys between some of the neighboring nodes to establish a secure link. This should form a connected graph of secure node-to-node links. Subsequently, neighbor-to-neighbor key establishment can be performed using the secure links in the initial graph created during key-setup.

3 Notation

For clarity, we list the symbols used in the paper below:

c	desired confidence level (probability) that the sensor network is connected after completing the connection protocol.
d	the expected degree of a node – i.e., the expected number of secure links a node can establish during key-setup.
m	number of keys in a node's key ring
n	network size, in nodes
n'	the expected number of neighbor nodes within communication radius of a given node
p	probability that two neighbor nodes can set up a secure link during the key-setup phase.
q	for the q-composite scheme, required amount of key overlap
S	key pool (set of keys randomly chosen from the total key space)
$\|S\|$	size of the key pool.
t	threshold number of votes after which a node will be revoked.

4 Background: overview of the basic random key predistribution scheme

Eschenauer and Gligor first proposed a random key-predistribution scheme [10]. In the remainder of this paper, we refer to their approach as the *basic scheme*. Let m denote the number of distinct cryptographic keys that can be stored on a sensor node. The basic scheme works as follows. Before sensor nodes are deployed, an *initialization phase* is performed. In the initialization phase, the basic scheme picks a random pool (set) of keys S out of the total possible key space. For each node, m keys are randomly selected from the key pool S and stored into the node's memory. This set of m keys is called the node's *key ring*. The number of keys in the key pool, $|S|$, is chosen such that two random subsets of size m in S will share at least one key with some probability p.

After the sensor nodes are deployed, a *key-setup phase* is performed. The nodes first perform key-discovery to find out with which of their neighbors they share a key. Such key discovery can be performed by assigning a short identifier to each key prior to deployment, and having each node broadcast its set of identifiers. Nodes which discover that they contain a shared key in their key rings can then verify that their neighbor actually holds the key through a challenge-response protocol. The shared key then becomes the key for that link.

After key-setup is complete, a connected graph of secure links is formed. Nodes can then set up *path keys* with nodes in their vicinity whom they did not happen to share keys with in their key rings. If the graph is connected, a path can be found from a source node to its neighbor. The source node can then generate a path key and send it securely via the path to the target node.

One needs to pick the right parameters such that the graph generated during the key-setup phase is connected. Consider a random graph $G(n, p_l)$, a graph of n nodes for which the probability that a link exists between any two nodes is p_l. Erdös and Rényi showed that for monotone properties of a graph $G(n, p_l)$, there exists a value of p_l over which the property exhibits a "phase transition", i.e. it abruptly transitions from "likely false" to "likely true" [21]. Hence, it is possible to calculate some expected degree d for the vertices in the graph such that the graph is connected with some high probability c, where $c = 0.999$, for example. Eschenauer and Gligor calculate the necessary expected node degree d in terms of the size of the network n as:

$$d = \left(\frac{n-1}{n}\right)(\ln(n) - \ln(-\ln(c))) \tag{1}$$

From the formula, $d = O(\log n)$. In our examples we expect d to be in the range of 20 to 50.

For a given density of sensor network deployment, let n' be the expected number of neighbors within communication range of a node. Since the expected node degree must be at least d as calculated, the required probability p of successfully performing key-setup with some neighbor is:

$$p = \frac{d}{n'} \tag{2}$$

Since the models of connectivity are probabilistic, there is always the chance that the graph may not be fully connected. This chance is increased if the deployment pattern is irregular or the deployment area has unpredictable physical obstacles to communication. It is difficult to anticipate such scenarios prior to knowing the specifics of the deployment area. To address this, if the network detects that it is disconnected, sensor nodes should perform *range extension*. This may involve increasing their transmission power, or sending a request to their neighbors to forward their communications for a certain number of hops. Range extension may be gradually increased until a connected graph is formed after key-setup. A useful way for a node to detect if a network is connected is by checking if it can perform multi-hop communication with all base stations. If not, range extension should be performed.

5 q-composite random key predistribution scheme

In the basic scheme, any two neighboring nodes need to find a single common key from their key rings to establish a

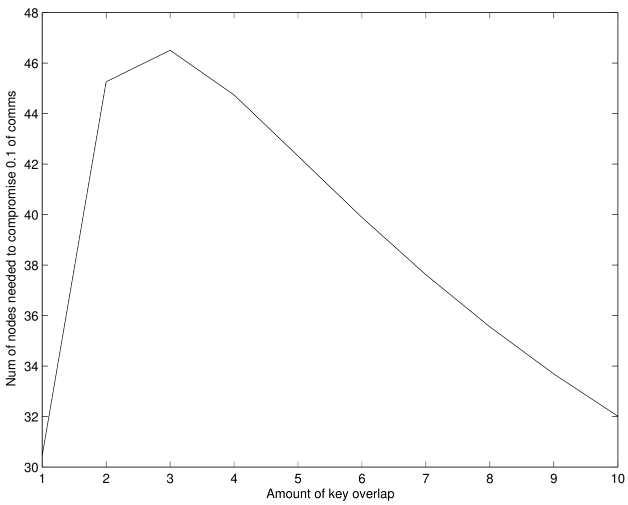


Figure 1. The expected number of nodes an adversary needs to capture before it is able to eavesdrop on any link with probability 0.1, for various amounts of key overlap q. Key ring size $m = 200$ keys, probability of connection $p = 0.5$.

secure link in the key-setup phase. We propose a modification to the basic scheme where q common keys $(q > 1)$ are needed, instead of just one. By increasing the amount of key overlap required for key-setup, we increase the resilience of the network against node capture.

Figure 5 reflects the motivation for the q-composite keys scheme. As the amount of required key overlap increases, it becomes exponentially harder for an attacker with a given key set to break a link. However, to preserve the given probability p of two nodes sharing sufficient keys to establish a secure link, it is necessary to reduce the size of the key pool $|S|$. This allows the attacker to gain a larger sample of S by breaking fewer nodes. The interplay of these two opposing factors results in an optimal amount of key overlap to pose the greatest obstacle to an attacker for some desired probability of eavesdropping on a link.

5.1 Description of the q-composite keys scheme

5.1.1 Initialization and key setup

The operation of the q-composite keys scheme is similar to that of the basic scheme, differing only in the size of the key pool S and the fact that multiple keys are used to establish communications instead of just one.

In the initialization phase, we pick a set S of random keys out of the total key space, where $|S|$ is computed as described later in Section 5.1.2. For each node, we select m random keys from S (where m is the number of keys each node can carry in its key ring) and store them into the node's key ring.

In the key-setup phase, each node must discover all common keys it possesses with each of its neighbors. This can be accomplished with a simple local broadcast of all key identifiers that a node possesses. While broadcast-based key discovery is straightforward to implement, it has the disadvantage that a casual eavesdropper can identify the key sets of all the nodes in a network and thus pick an optimal set of nodes to compromise in order to discover a large subset of the key pool S. A more secure, but slower, method of key discovery could utilize client puzzles such as a Merkle puzzle [16]. Each node could issue m client puzzles (one for each of the m keys) to each neighboring node. Any node that responds with the correct answer to the client puzzle is thus identified as knowing the associated key.

After key discovery, each node can identify every neighbor node with which it shares at least q keys. Let the number of actual keys shared be q', where $q' \geq q$. A new communication link key K is generated as the hash of *all* shared keys, e.g., $K = hash(k_1||k_2||\ldots||k_{q'})$. The keys are hashed in some canonical order, for example, based on the order they occur in the original key pool S. Key-setup is not performed between nodes that share fewer than q keys.

5.1.2 Computation of key pool size

We assume that we are required to take the sensor network's physical characteristics as a given parameter. Specifically, we are provided with a probability of full network connectivity c and the expected number of neighbors of each node n'. Via Equation 1, we first calculate d, the expected degree of any given node. This can be input to Equation 2 to calculate p, the desired probability that any two nodes can perform key-setup.

We now need to calculate the critical parameter $|S|$, the size of the key pool. If the key pool size is too large, then the probability of any two nodes sharing at least q keys would be less than p, and the network may not be connected after bootstrapping is complete. If the key pool size is too small, then we are unnecessarily sacrificing security. We would like to choose a key pool size such that the probability of any two nodes sharing at least q keys is $\geq p$. Let m be the number of keys that any node can hold in its key ring. We would like to find the largest S such that any two random samples of size m from S have at least q elements in common, with a probability of at least p.

We compute $|S|$ as follows. Let $p(i)$ be the probability that any two nodes have exactly i keys in common. Any given node has $\binom{|S|}{m}$ different ways of picking its m keys from the key pool of size $|S|$. Hence, the total number of ways for both nodes to pick m keys each is $\binom{|S|}{m}^2$. Suppose the two nodes have i keys in common. There are $\binom{|S|}{i}$ ways to pick the i common keys. After the i common keys have been picked, there remain $2(m-i)$ distinct keys in the two key rings that have to be picked from the remaining pool of $|S|-i$ keys. The number of ways to do this is $\binom{|S|-i}{2(m-i)}$. The $2(m-i)$ distinct keys must then be partitioned between the two nodes equally. The number of such equal partitions is $\binom{2(m-i)}{m-i}$. Hence the total number of ways to choose two key rings with i keys in common is the product of the aforementioned terms, i.e., $\binom{|S|}{i}\binom{|S|-i}{2(m-i)}\binom{2(m-i)}{m-i}$. Hence, we have

$$p(i) = \frac{\binom{|S|}{i}\binom{|S|-i}{2(m-i)}\binom{2(m-i)}{m-i}}{\binom{|S|}{m}^2} \quad (3)$$

Let $p_{connect}$ be the probability of any two nodes sharing sufficient keys to form a secure connection. $p_{connect} = 1-$ (probability that the two nodes share insufficient keys to form a connection), hence

$$p_{connect} = 1 - (p(0) + p(1) + \cdots + p(q-1)) \quad (4)$$

For a given key ring size m, minimum key overlap q, and minimum connection probability p, we choose the largest $|S|$ such that $p_{connect} \geq p$.

5.2 Evaluation of the q-composite random key distribution scheme

We evaluate the q-composite random key distribution scheme in terms of resilience against node capture and the maximum network size supported. We note that this scheme has no resistance against node replication since node degree is not constrained and there is no limit on the number of times each key can be used. The scheme can support node revocation via a trusted base station similar to the approach in [10].

5.2.1 Resilience against node capture in q-composite keys schemes

In this section we evaluate how the q-composite scheme improves a sensor network's resilience in the face of a node capture attack by calculating the fraction of links in the network that an attacker is able to eavesdrop on *indirectly* as a result of recovering keys from captured nodes. That is, we attempt to answer the question: For any two nodes A and B in the network, where neither A nor B have been captured by the attacker, what is the probability that the attacker can decrypt their communications using the subset of the key pool that was recovered from the nodes that were compromised.

We show that the q-composite key scheme strengthens the network's resilience against node capture when the number of nodes captured is low. Let the number of captured nodes be x. Since each node contains m keys, the probability that a given key has not been compromised is $(1-\frac{m}{|S|})^x$. The expected fraction of total keys compromised is thus $1-(1-\frac{m}{|S|})^x$. For any communication link between two nodes, if its link key was the hash of i shared keys, then the probability of that link being compromised is $(1-(1-\frac{m}{|S|})^x)^i$. The probability of setting up a secure link is $p = p(q) + p(q+1) + \ldots + p(m)$. Hence, we have that the probability that any secure link setup in the key-setup phase between two uncompromised nodes is compromised when x nodes have been captured is

$$\sum_{i=q}^{m}\left(1-\left(1-\frac{m}{|S|}\right)^x\right)^i \frac{p(i)}{p}$$

This equation also represents the fraction of additional communications (i.e., external communications in the network independent of the captured nodes) that an adversary can compromise based on the information retrieved from x number of captured nodes. Figure 2 shows how it varies with the number of nodes captured by the attacker.

We note that the scale of the x-axis shows absolute numbers of nodes compromised (i.e., independent of the actual total size of the network) while the y-axis is the fraction of the total network communications compromised. Hence, the schemes are not infinitely scalable - a compromise of x

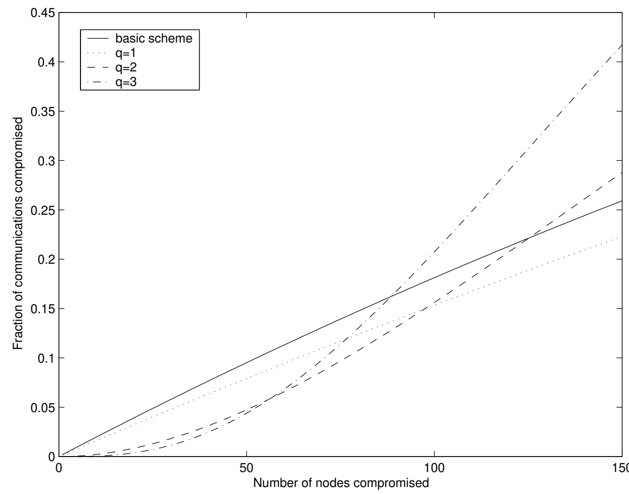


Figure 2. Probability that a specific random communication link between two random nodes A, B can be decrypted by the adversary when the adversary has captured some set of x nodes that does not include A or B. Key ring size $m = 200$, probability of key-setup $p = 0.33$.

number of nodes will always reveal a fixed fraction of the total communications in the network regardless of network size. A method to estimate the largest supportable network size of the various schemes is discussed in Section 5.2.2.

The q-composite keys scheme offers greater resilience against node capture when the number of nodes captured is small. For example, in Figure 2a, for $q = 2$, the amount of additional communications compromised when 50 nodes have been compromised is 4.74%, as opposed to 9.52% for the basic scheme. However, when large numbers of nodes have been compromised, the q-composite keys schemes tend to reveal larger fractions of the network to the adversary. By increasing q, we make it harder for an adversary to obtain small amounts of initial information from the network via a small number of initial node captures. This comes at the cost of making the network more vulnerable once a large number of nodes have been breached. This may be a desirable trade-off because small scale attacks are cheaper to mount and much harder to detect than large scale attacks. It is easy to mask an attack on a single node as a communications breakdown due to occlusion or interference; it is much harder to disguise an attack on many nodes as a natural occurrence.

The q-composite scheme removes the incentive for small scale attacks since the amount of additional information revealed in the rest of the network is greatly reduced. It forces the attacker to attempt large scale attacks which are expensive and more easily detectable.

5.2.2 Maximum supportable network sizes for the q-composite keys scheme

In this section we assess the scalability of the random key schemes we have presented thus far.

Since a fixed number of compromised nodes causes a *fraction* of the remaining network to become insecure, these random-key distribution schemes cannot be used for arbitrarily large networks. For example, based on Figure 2a, in the basic scheme, the capture of 50 nodes compromises approximately 9.5% of communications in the network. For a network of 10,000 nodes this translates to an approximate payoff of 10% of communications compromised for a cost to the attacker of capturing just 0.5% of total nodes, representing a relatively modest investment for a high payoff.

We can estimate a network's maximum supported size by framing the following requirement:

> *Limited global payoff requirement*: Suppose the adversary has captured some nodes, but is only able to break some fraction $f \leq f_m$ of all communications. We require that each subsequent node that is compromised to the enemy allows them to break as many links in the rest of the network, on expectation, as the average connectivity degree of a single node.

In other words, given that the network is still mostly secure ($f \leq f_m$), we would like that, on average, after capturing some node, the adversary does not learn more about the rest of the network than they learn about the communications of the node itself. Via this requirement, smaller scale attacks on a network must be mainly economically justified by the value of the individual nodes compromised rather than the amount of information that the captured keys can reveal in the rest of the network, thus limiting the incentive of an adversary to begin an attack. The maximum compromise threshold f_m intuitively represents the level of compromise past where the adversary gains an unacceptably high confidence of guessing the sensor readings of the entire network, and thus the network must be considered exposed and no longer secret. f_m will vary depending on the application and the correlation of different sensor readings.

Using the definition of limited global payoff, we can estimate the maximum allowable sizes for the networks such that our requirement holds true. For any number x of nodes compromised, we know that some fraction $f(x)$ of the remaining secure links created after key-setup have been compromised. Let x_m be the number of nodes compromised such that $f_m = f(x_m)$ of the other secure links created during key-setup has been compromised. f_m is a given parameter (see the definition of limited global payoff preceding). Let the average connectivity degree of a single node be d. The adversary thus holds an expected $x_m d$ connections in which the compromised nodes are directly involved. We require that the number of *additional* links compromised elsewhere in the network be less than this number of directly compromised links. There are $\frac{nd}{2}$ total links in the network. Hence, the requirement is that $(\frac{nd}{2} - x_m d) f_m \leq x_m d$. Simplifying,

$$n \leq 2x_m \left(1 + \frac{1}{f_m}\right) \quad (5)$$

Figure 3 shows the estimated maximum network sizes for the basic random keys scheme as well as for several parameters of the q-composite keys scheme. We note that the maximum network sizes scale linearly with key ring size m. For example, for $p = 0.33$, $f_m = 0.1$, and $m = 200$, the maximum network size for the 2-composite keys scheme is $1,415$ nodes while the maximum network size for the basic scheme is $1,159$ nodes.

These calculations are our proposed method of estimating the maximum supportable size of a network given that certain security properties hold. Alternative methods may exist that produce different network size estimations.

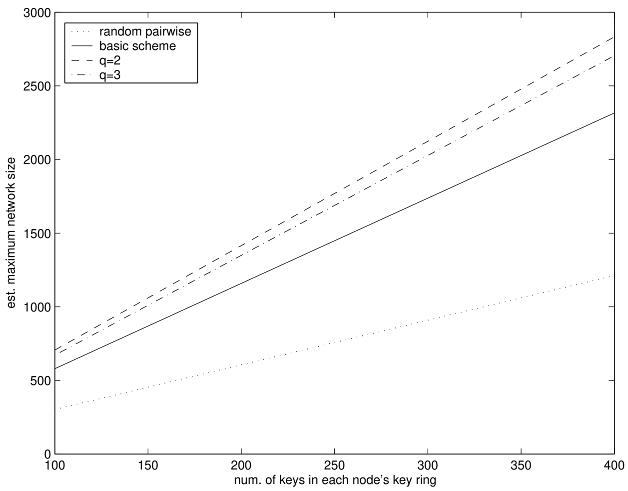


Figure 3. Maximum network sizes
$(p = 0.33, f_m = 0.1)$

6 Multipath Key Reinforcement

In this section we present *multipath key reinforcement*, a method to strengthen the security of an established link key by establishing the link key through multiple paths. This method can be applied in conjunction with the basic random key scheme to yield greatly improved resilience against node capture attacks by trading off some network communication overhead. We analyze the resulting scheme and explain why we discourage using multipath key reinforcement in conjunction with a q-composite scheme.

6.1 Description of multipath key reinforcement

The basic idea behind multipath key reinforcement was first explored by Anderson and Perrig [2]. We assume that initial key-setup has been completed (in the following examples, we assume the basic random key scheme was used for key-setup). There are now many secure links formed through the common keys in the various nodes' key rings. Suppose A has a secure link to B after key-setup. This link is secured using a single key k from the key pool S. k may be residing in the key ring memory of some other nodes elsewhere in the network. If any of those nodes are captured, the security of the link between A and B is jeopardized. To address this, we would like to update the communication key to a random value after key-setup. However, we cannot simply coordinate the key update using the direct link between A and B since if the adversary has been recording all key-setup traffic, it could decrypt the key-update message after it obtained k and still obtain the new communication key.

Our approach is to coordinate the key-update over multiple independent paths. Assume that enough routing information can be exchanged such that A knows all disjoint paths to B created during initial key-setup that are h hops or less. Specifically, $A, N_1, N_2, \ldots, N_i, B$ is a path created during the initial key-setup if and only if each link $(A, N_1), (N_1, N_2), \ldots, (N_{i-1}, N_i), (N_i, B)$ has established a link key during the initial key-setup using the common keys in the nodes' key rings. Let j be the number of such paths that are *disjoint* (do not have any links in common). A then generates j random values $v_1, \ldots, v_j$. Each random value has the same length as the encryption/decryption key. A then routes each random value along a different path to B. When B has received all j keys, then the new link key can be computed by both A and B as:

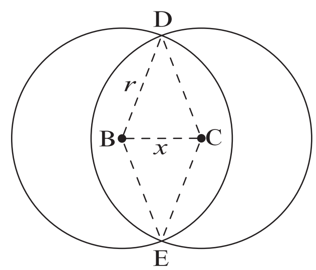
$$k' = k \oplus v_1 \oplus v_2 \oplus \ldots \oplus v_j$$

The secrecy of the link key k is protected by all j random values. Unless the adversary successfully manages to eavesdrop on all j paths, they will not know sufficient parts of the link key to reconstruct it.

The more paths we can find between two nodes A and B, the more security multipath key reinforcement provides for the link between A and B. However, for any given path, the probability that the adversary can eavesdrop on the path increases with the length of the path since if any one link on the path is insecure then the entire path is made insecure. Further, it is increasingly expensive in terms of communication overhead to find multiple disjoint paths that are very long. In this paper we will analyze the case where only paths of 2 links (only one intermediate node) are considered. We call this scheme the *2-hop multipath key reinforcement scheme*. This approach has the advantage that path discovery overhead is minimized: for example, A could exchange neighbor lists with B. Once they identify their common neighbors with which both of them share a key, A and B can perform key reinforcement using their secure links through these common neighbors. Furthermore, the paths are naturally disjoint and no further effort needs to be taken to guarantee this property. We will calculate the expected effectiveness of this scheme and evaluate its security properties in simulation.

6.2 Estimation of expected effectiveness of 2-hop multipath key reinforcement

In this section, we first calculate the expected number of common neighbors between two nodes in a random uniform planar deployment of sensors. We then derive a formula for the new expected probability for compromising a given link after multipath key reinforcement has taken place.



The figure above indicates the parameters to be used in our calculation. B and C denote two communicating sensor nodes. r is the communications range of each sensor node. We assume that each node has the same range for receiving and transmitting. x is the distance between two nodes.

For any given separation x, the area $A(x)$ within both nodes' communication radii is the area of the sectors BDE and CDE minus the area of the rhombus $BDCE$:

$$A(x) = 2r^2 \cos^{-1}\left(\frac{x}{2r}\right) - x\sqrt{r^2 - \frac{x^2}{4}}$$

The probability distribution function of the distance between two nodes within communication radius is given by $F(x) = P(distance < x) = x^2/r^2$. The probability density function is thus $f(x) = F'(x) = 2x/r^2$. The expected area of overlap is thus given by:

$$\begin{aligned} & \int_0^r A(x)f(x)dx \\ = & \int_0^r \left(2r^2 \cos^{-1}\left(\frac{x}{2r}\right) - x\sqrt{r^2 - \frac{x^2}{4}}\right) \frac{2x}{r^2}\, dx \\ = & \left(\pi - \frac{3\sqrt{3}}{4}\right) r^2 \; = \; 0.5865\pi r^2 \end{aligned}$$

We define the term *reinforcing neighbors* of two nodes sharing a secure link as the common neighbors with whom both nodes share a secure link. Since the expected area of overlap is 0.5865 of a single communication radius, the expected number of reinforcing neighbors is thus $0.5865p^2n'$ where p is the probability of sharing sufficient keys to communicate, and n' is the number of neighbors of each node. Via Equation 2, this can also be expressed as $0.5865\frac{d^2}{n'}$. As an example, for $d = 20$ and $n' = 60$ (i.e. $p = 0.33$), the expected number of reinforcing neighbors is 3.83.

In general, if a link is reinforced by k common neighbors, then the adversary must be able to eavesdrop on that link, as well as at least one link on each of the k 2-hop paths. If the adversary's base probability of compromising a link is b, then the probability of compromising at least one hop on any given 2-hop path is the probability of compromising hop 1 in the path plus the probability of compromising hop 2 in the path minus probability of compromising both hops in the path = $2b - b^2$. Hence, the final probability of breaking the link is now

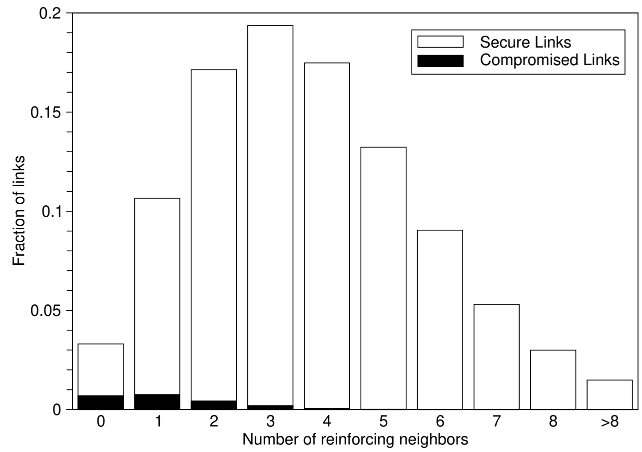


Figure 4. Reinforcement and compromise statistics for base compromise probability $b = 0.2$

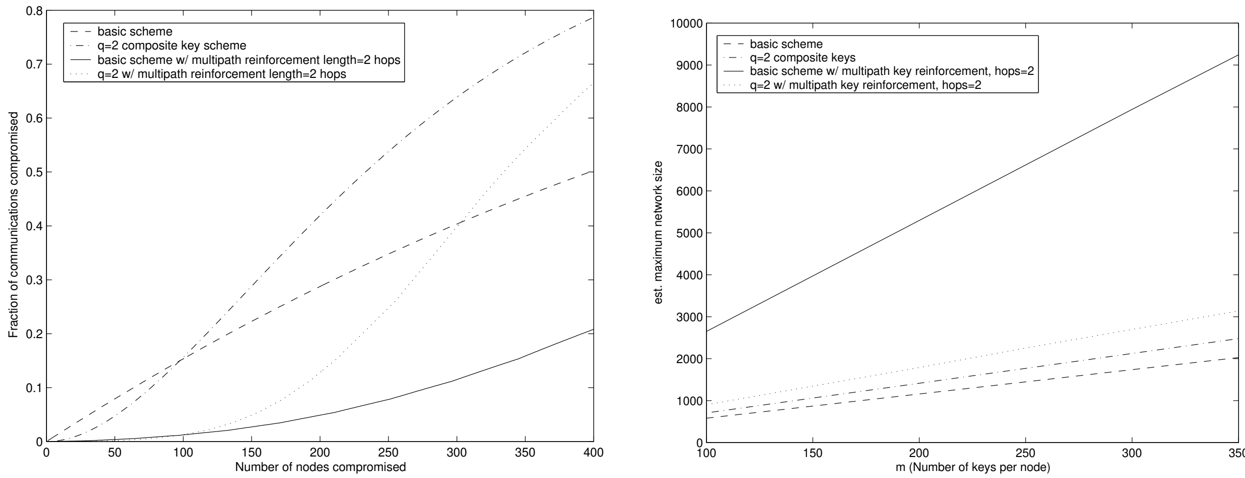
$$b' = b(2b - b^2)^k$$

For example, if the adversary has a base 0.1 chance of eavesdropping on a given link before reinforcement, for a link reinforced by 3 neighbors, the chance of eavesdropping after reinforcement improves to 6.86×10^{-4}, or about 1 in 1,458.

From the expected number of reinforcing neighbors we can estimate the expected network communications overhead of the 2-hop multipath reinforcement scheme. Each reinforcing neighbor represents an extra 2-hop communication to help reinforce a given 1-hop link. Hence, on average, the total additional communications overhead for key-reinforcement is at least $2 \times 0.5865p^2n'$ times more than the network communications needed for basic key-setup, not including additional communications for common-neighbor discovery. For example, for $p = 0.33$ and $n' = 60$, we can expect to see at least 7.66 times additional network traffic after key-setup is complete. Including common neighbor discovery, we estimate the final scheme to be approximately 10 times more expensive in network communications than the basic scheme in this case. Given that eavesdropping probabilities can be improved from 0.1 to $6.86 \times 10^{-}4$ (146 times improvement), this may be a good trade-off.

6.3 Evaluation of multipath key reinforcement

The effectiveness of 2-hop multipath key reinforcement is evaluated by simulating the random uniform deployment of 10,000 sensor nodes on a square planar field. The probability of any two nodes being able to establish a secure link is set at $p = 0.33$, and the deployment density is set



(a) Resistance against node capture

(b) Maximum network sizes

Figure 5. Multipath key reinforcement results ($m = 200, p = 0.33$)

such that the expected number of neighbors of each node was 60. The eavesdropping attack is modeled by iterating over each secure link and marking it as compromised with random chance based on the simulated probability of compromise c. A link is considered completely compromised only if it is compromised and all its reinforcement paths are also compromised.

Figure 4 reflects the relative distribution of the number of reinforcing neighbors for each link in the simulation. The results indicated reflect support for our calculated average of 3.83 reinforcing neighbors between any 2 nodes within communication distance. The figure also shows the distribution of reinforced links that were compromised by an adversary with a base 0.2 probability of compromising any link prior to reinforcement. In this simulation, links with more than 3 reinforcing neighbors did not suffer significant rates of compromise. The overall rate of compromise was lowered by an order of magnitude, from 0.2 to 0.022.

Figure 5a indicates the amount of communications compromised versus the number of nodes compromised, with and without key reinforcement for the various schemes. Successfully implementing multipath key reinforcement on the basic scheme enables it to outperform the q-composite scheme for $q \geq 2$ even when the q-composite scheme is supplemented by key reinforcement. The intuitive reason for this is that multipath key reinforcement acts similarly to the q-composite keys scheme in that it compounds the difficulty of compromising a given link by requiring the adversary possess multiple relevant keys to eavesdrop on a given link. The trade-off for this benefit in the q-composite case is a smaller key pool size; the trade-off for the multipath key reinforcement scheme is increased network overhead. Compounding both the schemes compounds their weaknesses - the smaller key pool size of the q-composite keys scheme undermines the effectiveness of multipath key reinforcement by making it easier to build up a critically large collection of keys.

Figure 5b shows the maximum network size of the basic scheme with multipath key reinforcement. The graphs show that multipath key reinforcement gives a significant boost to network size performance when implemented on the basic scheme, but has little effect with the q-composite scheme.

The cost of the improved security due to multipath key reinforcement is an added overhead in neighbor discovery and key establishment traffic. Whether this tradeoff is a good one will depend on the specific application as well as the deployment density characteristics of the sensor network.

While the analysis presented is for using multipath key reinforcement to secure links that have been formed after key-setup, the scheme can also be used to reinforce *path-keys* that are established between nodes that did not share keys during key setup. This will further improve the security of the system.

7 Random-pairwise keys scheme

In the random key schemes presented thus far, while each node can verify that some of its neighbors have certain secret keys and are thus legitimate nodes, no node can authenticate the identity of a neighbor that it is communicating with. For example, suppose node A shares some set of keys

K with node B and that they use these keys as the basis for securing a communications link. Because keys can be issued multiple times out of the key pool, other nodes, e.g., C, could also hold this set of secret keys K in its key ring. A cannot ascertain that it is really communicating with B and not C, since it knows nothing more about B than its knowledge of K. We define the property that we are seeking as follows:

> **Node-to-node authentication**. A protocol has the property of node-to-node authentication if any node can ascertain the identity of the nodes that it is communicating with.

This property is useful in supporting many security functions. For example, in detecting node misbehavior, it is essential that a node be certain of the misbehaving node's identity before taking any action. Node-to-node authentication can also allow individual sensor nodes to resist a node replication attack by keeping track of which node identities have already been inserted into the network and rejecting further connection attempts by that identity. As a final example, node-to-node authentication can shift security functions away from the base station by enabling nodes to autonomously perform revocations on misbehaving nodes, thus improving reaction times toward detectable network intrusions.

In this section, we propose a new key establishment protocol called the *random pairwise scheme* that possesses the key property of node-to-node authentication. The random pairwise scheme has the following properties:

- *Perfect resilience against node capture.* Any node that is captured reveals no information about links that it is not directly involved in.
- *Node-to-node identity authentication.* Nodes are able to verify the identities of the nodes with whom they are communicating. An adversary is unable to impersonate the identity of any node B unless B has already been captured.
- *Distributed Node Revocation.* With some added overhead in key storage, misbehaving nodes can be revoked from the network without involving a base station.
- *Resistance to node replication and generation.* The scheme can reduce the opportunity of node replication at some cost to node memory and communication setup overhead.
- *Comparable scalability.* The scheme can support a maximum number of nodes that is comparable to the number of nodes supportable by the basic scheme and q-composite schemes under the limited global payoff requirement framed in Section 5.2.2.

7.1 Description of the random pairwise scheme

Suppose a sensor network has a maximum of n nodes. A simple solution to the key-predistribution problem is the *pairwise* keys scheme where each node contains $n-1$ communication keys each being pairwise privately shared with one other node in the network.

The *random pairwise* keys scheme is a modification of the pairwise keys scheme based on the observation that not all $n-1$ keys need to be stored in the node's key ring to have a connected random graph with high probability. Erdös and Rényi's formula allows us to calculate the smallest probability p of any two nodes being connected such that the entire graph is connected with high probability c. To achieve this probability p in a network with n nodes, each node need only store a random set of np pairwise keys instead of exhaustively storing all $n-1$. Reversing the calculation, if a node can store m keys, then the maximum supportable network size is

$$n = \frac{m}{p} \tag{6}$$

Depending on the model of connectivity, p may grow slowly with n when n is large (intuitively, p cannot decrease as n goes toward infinity, since it is more likely that a large graph is disconnected than a smaller graph). Hence, n should increase with increasing m and decreasing p. The exact rates will depend on the deployment model.

The use of pairwise keys instead of purely random keys chosen from a given pool can give us node-to-node authentication properties if each node which holds some key k, also stores the identity (ID) of the other node which also holds k. Hence, if k is used to create a secure link with another node, both nodes are certain of the identity of each other since no other nodes can hold k.

7.1.1 Initialization and key-setup in the random pairwise keys scheme

Recall that the size of each node's key rings is m keys, and the probability of any two nodes being able to communicate securely is p. The random pairwise keys scheme proceeds as follows:

1. In the pre-deployment *initialization* phase, a total of $n = \frac{m}{p}$ unique node identities are generated. The actual size of the network may be smaller than n. Unused node identities will be used if additional nodes are added to the network in the future. Each node identity is matched up with m other randomly selected distinct node IDs and a pairwise key is generated for each pair of nodes. The key is stored in both nodes' key rings, along with the ID of the other node that also knows the key.

2. In the post-deployment *key-setup* phase, each node first broadcasts its node ID to its immediate neighbors. By searching for each other's IDs in their key-rings, the neighboring nodes can tell if they share a common pairwise key for communication. A cryptographic handshake is then performed between neighbor nodes who wish to mutually verify that they do indeed have knowledge of the key.

7.1.2 Multi-hop range extension

Since the node ID is just a few bytes, key discovery involves much less network traffic and computational overhead in the nodes than standard random-key predistribution. Hence the effective communication range of nodes for key setup can be extended beyond physical communication range by having neighboring nodes re-broadcast the node ID for a certain number of hops. Each hop that the node ID is rebroadcast effectively extends the range by approximately one communication radius, increasing the number of nodes that can hear the broadcast by a squared factor. The table below shows some intuition for number of reachable nodes in the case where the expected number of neighbors within communication range is 60.

local (0 hops)	1 hop	2 hops	3 hops
60	240	540	960

This has an impact on the maximum supportable network size n. Recall from Equation 2 that connection probability $p = \frac{d}{n'}$ where n' is the number of neighbors and d was computed via the required probability of graph connectivity. From Equation 6 we have that maximum network size $n = \frac{m}{p}$ where m is the key ring size. Hence

$$n = \frac{mn'}{d} \tag{7}$$

By increasing the effective communications radius, we also increase the number of neighbors n', hence the maximum supportable network size n also increases. Multihop range extension should be used with caution, however, because the rebroadcast is performed without verification or authentication. Hence, during the deployment phase, an adversary can send random node IDs into the network which will then be rebroadcast x times by the receiving nodes. This potential denial of service (DoS) attack could stop or slow the key-setup process since the sensor network is actively helping to amplify the range of the adversary's interfering transmissions. The potential damage due to this DoS attack can be reduced by limiting the number of hops of the range extension. If DoS is a serious concern then multihop range extension could be removed altogether; it is not required for the operation of the random pairwise scheme.

7.1.3 Support for distributed node revocation

In the random pairwise scheme, node revocation can be supported via base stations as described by Eschenauer and Gligor [10]. However, base station initiated revocation mechanisms may also slow the node revocation process due to the potential high latency between the nodes and the base-station. In revocation, fast response is particularly crucial since a detected attack must be sealed off from the network before it can do significant harm.

To reduce the disadvantages associated with a base-station dependent revocation protocol, we present a distributed node revocation scheme for the random pairwise scheme. Such a scheme is possible if we assume the existence of a mechanism in each sensor node that enables it to detect if neighbor nodes have been compromised. The scheme works by having neighboring nodes broadcast 'public votes' against a detected misbehaving node (we use the term *public vote* since the identity of the voter in this case need not be kept secret). If any node B observes more than some threshold number t of public votes against some node A, then B breaks off all communications with A. By listening on the network (like any other sensor node), the base station can relay the votes back to a physically secure location where the undeployed nodes are stored. There, any as-yet undeployed node identities react appropriately by erasing any pairwise keys associated with A from the undeployed nodes' key rings. This has the effect of permanently severing node A from the network.

It is a technical challenge to design a compact and efficient distributed public vote counting mechanism for sensor nodes.

In the following discussion, the set of nodes which can vote against node A are termed A's *voting members*. We require the voting scheme to have the following properties:

- Compromised nodes cannot revoke arbitrary nodes.
- No voting member of A is able to forge another member's vote against A.
- Each voting member of A must be able to verify the validity of a broadcast public vote against A.
- Broadcast public votes from one voting member reveal no information that would allow listeners to generate additional public votes.
- Broadcast public votes have no replay value.
- The method of propagating the broadcast to cover the entire network should not be vulnerable to denial of

service attack by a malicious node operating within the network.

As a first attempt, a simple scheme is as follows: Consider a node A, which, like every other node in the network, has m keys in its key ring. Since all the keys are issued to exactly two nodes and no two keys are issued to the same pair of nodes, we have exactly m nodes that share a pairwise key with node A. We call this set of m nodes the set of *voting members* of A. Each of these m voting members are assigned a random voting key k_i. Each voting member also knows the respective hashes of the voting keys of all the $m-1$ other voting members, i.e. $hash(k_j), j \neq i, 1 \leq j \leq m$. To cast a public vote against A, the node broadcasts k_i. All other voting members can verify the vote by computing $hash(k_i)$. Once k_i is verified, voting members can replace $hash(k_i)$ with k_i and a flag reflecting the fact that this vote has already been heard on the network.

One problem with this scheme is that each entry on the key ring now stores not only the pairwise key but also $m-1$ hash values and a voting key. Hence, if m pairwise keys are stored on the node, the memory requirement is $O(m^2)$.

In our scheme, we propose using a Merkle tree [17] to efficiently authenticate m hash values. Only a single verifying hash value (the root value of the Merkle tree) needs to be stored, but the voting information is now size $O(\log m)$, since each node now needs to reveal not just its secret voting key but also the hash values of the $\log m$ internal nodes in the Merkle tree that will allow the other voting members to authenticate the vote.

One consequence of using a Merkle tree mechanism is that it is now necessary to remember which nodes have already been received, in order to remove replay value of the votes. For each vote, the path to the root of the Merkle tree is unique and can be described in $\log m$ bits. Hence, only $\log m$ bits of storage per received vote is necessary. Also, a total of at most $t \log m$ bits is needed since only t votes need to be received before revocation occurs. t is generally chosen to be small, as described below.

Choice of the threshold value t. Let t be the minimum number of votes needed to revoke a node. t must be chosen low enough such that it is unlikely that any node has a degree $< t$ in the network, but high enough such that a collection of rogue nodes cannot cause the revocation of many legitimate nodes. For any of the m keys in a node's key ring, the probability that it is used is the probability that the other node which has this key is within communication radius. This probability is $\frac{n'}{n}$ since there are n' neighbors out of n total nodes, that will be within communication radius. The distribution of the degree of a node is hence binomial $(m, \frac{n'}{n})$. Since $n = \frac{mn'}{d}$(from Equation 7, where d is the expected degree of a node in terms of number of secure links created during key-setup), $\frac{n'}{n}$ simplifies to $\frac{d}{m}$. Hence we have that the degree of a node is binomial$(m, \frac{d}{m})$, the average is d and the variance is $d(1-\frac{d}{m})$. For key ring sizes sufficient to support a reasonably sized network, $\frac{d}{m}$ will be small. Hence the variance is close to the average d, i.e., the distribution is heavily skewed to the left.

The expected degree of a node d should increase slowly with network size n (from Equation 1, $d = O(\log n)$). Hence t should remain small (≤ 5) for the range of network sizes we are considering in this paper (1,000 to 10,000 nodes). Since t is small, we note that memorizing previously cast votes to prevent replay is not a significant memory cost.

One consequence of implementing such a voting scheme is that no node can have less than t neighbors, otherwise that node cannot be revoked. Since t was chosen such that it is unlikely that any node has degree $< t$ in the network, the scheme can be modified such that any node that is unable to form at least kt connections (where k is some small multiple, e.g. 2) on the network after the key-setup phase must be revoked. Such low-degree nodes can be detected via the degree-counting mechanism described in Section 7.1.4 below.

Even if this mechanism is in place, if an adversary can selectively compromise nodes without detection, then it may be possible to compromise a set of nodes that shield each other from revocation, e.g. compromise enough nodes around a misbehaving node such that only $t-1$ legitimate nodes are left to communicate with it. Another method of attack would be to only present detectable misbehavior to $t-1$ neighbors so as to prevent revocation. In such cases, proper revocation may still be possible depending on the sensitivity and accuracy of the detection mechanism. However, designing a node-level intrusion detection mechanism that has both high sensitivity and accuracy is an extremely challenging problem. Hence, base-station issued revocation mechanisms may still be used to limit the potential damage that can be caused by these sophisticated attacks. Distributed node revocation is best used as a fast-reaction system to respond to perceived node-capture attacks, rather than as a full counter-measure against a malicious node that has already entered the network.

Broadcast mechanism. Our public voting scheme relies on being able to propagate every public vote across the network to all voting members. However, having every node naively re-broadcast all votes heard on the open network presents a vulnerability to denial of service attack. In our scheme, only the voting members will re-broadcast any received public votes to each other, while all other nodes ignore the broadcast. This transmission is performed unencrypted, since public votes need not be secret once they are broadcast. Since there is no transmission control in an unencrypted broadcast, we require that each voting member that first receives a correctly verified vote perform a re-broadcast of the vote a fixed number of times at varying intervals in order to maximize the probability of a successful transmission to a neighboring voting member. We now show that every voting member will receive this broadcast with approximately the same very high probability c that the network is connected (in this analysis, we assume the broadcast transmission is perfect). We assume that αn nodes have been deployed where $0.5 < \alpha < 1$ (i.e. a significant fraction of the nodes have been deployed). Each voting member has an expected total of n' neighbors within range. There are about αm voting members that have been deployed. Each voting member can thus expect to find $(\frac{\alpha m-1}{\alpha n-1})n'$ other voting members within communications range. Since m and n are large and α is a large fraction, we can approximate this with $(\frac{\alpha m}{\alpha n})n'$ which simplifies to $\frac{mn'}{n}$. However, from Equation 7, this is exactly the degree d that is required to connect the graph with high probability c. Hence the network of voting members forms a random graph with almost the same probability of being connected as the original network of secure links (it may be slightly lower due to our approximation). The reason for this is that the voting members can perform unencrypted broadcast to every voting member within range, whereas the communication links established during key-setup between arbitrary neighbor nodes must be conditional on the sharing of a pairwise key. For any node A, the probability of an arbitrary node B being a relevant voting member is approximately $\frac{m}{n}$ which is exactly the probability of B sharing a pairwise key with A. Both the graphs are connected with high probability.

Resisting revocation attack. One possible weakness associated with distributed node revocation is that each node holds the potential to cast a vote against m other nodes. Since the total number of nodes $n = \frac{m}{p}$, this could represent a significant fraction of the node population. Hence only a fixed number of nodes need to be compromised without detection in order for them to revoke a significant proportion of the network, regardless of the network size.

To prevent widespread release of revocation keys by compromised nodes, we require that only nodes that have established direct communication with some node B have the ability to revoke B.

We do this by distributing the revocation keys to the voting members of B in a deactivated form, i.e. each voting member i stores its revocation key for B k_{Bi} masked (XORed) with some secret S_{Bi}. This deactivated key will not hash to the correct verifying value and is thus useless for voting. Node B knows all the activation secrets $S_{Bi}, 1 \leq i \leq m$. During the key discovery and setup phase, if node i wishes to complete key setup with node B, it requires node B to transmit its activation secret S_{Bi} (and vice-versa). Once node i has received S_{Bi} it unmasks k_{Bi} using S_{Bi}, and verifies that it was given the correct unmasking secret by performing vote verification on the unmasked k_{Bi} to see if it is a valid revocation key. Storage of m masking factors on node B takes only $O(m)$ space and is insignificant compared to the total $O(m \log m)$ space needed to store the voting and verification apparatus.

Such a policy of need-to-know key activation ensures that the majority of revocation keys recovered through node capture are in an unusable masked state. In order to use these revocation keys to revoke some node A the adversary now has to physically communicate with A and complete key-setup for up to t new connections.

Via this mechanism, the adversary's ability to attempt sabotage through this course of action is seriously limited by the implementation of schemes to limit node replication and node generation (see next Section 7.1.4). In general, since resistance against node replication imposes an upper limit d_{max} on the degree of a node, once a malicious node has collected d_{max} activation values for its revocation keys, further requests for activation values will be rejected by the other nodes in the network since it will be detected that this node is attempting to exceed its maximum allowed degree. Hence the number of revocation keys issuable by each compromised node is limited to d_{max}.

Even if we do not assume the implementation of schemes for resisting node replication, the requirement that the adversary establish physical (1-hop) communication with a target node is a strong disincentive to mount a DoS attack via revocation. For example, if disruption rather than subversion of the network is all that is desired by the adversary and the adversary has the ability to physically communicate with the target nodes, then a radio jamming attack is probably cheaper and more productive than a revocation attack.

The vote-activation mechanism presented above limits the damage an adversary can inflict by broadcasting node revocations. It does not completely eliminate the potential for such an attack. However, it does makes it less economically viable for an attacker to mount a revocation attack.

7.1.4 Resistance against node replication and node generation

In the event that node capture goes undetected by the network, it is desirable that the network be resistant against the addition of infiltrator nodes derived from captured nodes, especially considering that resistance may be required to prevent revocation attack on the network (see Section 7.1.3)

To limit the amount node replication possible on the network, the degree of any node can be limited. We know that the degree of a node on the network is approximately binomially distributed $(m, \frac{d}{m})$ with expectation d and variance close to d (see Section 7.1.3 for derivation). Hence very few nodes should have degree $\geq 3d$, for example. This implies that we can limit the degree of nodes to d_{max} where d_{max} is some small multiple of d, without disrupting network connectivity.

The expected degree d increases slowly with graph size n. For example, Equation 1 indicates $d = O(\log n)$. Hence d_{max} will generally be small compared with the total *potential* connectivity m.

Since the random-pairwise scheme allows us to have a notion of authenticated node identity, a method for node-degree counting for the random-pairwise scheme may be implemented with the public-vote counting scheme presented in Section 7.1.3. The operation of the degree-counting scheme is exactly identical. Each node contains a voting key and some way to verify valid voting keys. Each time a given node A forms a connection with some node B, A broadcasts its voting key for B and vice-versa. Each node can thus track the degree of all m of the nodes which share pairwise keys with it, and refuse to form new connections if the degree becomes too large.

One concern in this case is that we now need to memorize d_{max} number of cast votes instead of a small number t. Each vote still requires only $\log m$ bits to store since we only need to store its unique path in the Merkle tree, hence directly storing d_{max} votes may still be feasible. Otherwise, for applications with a relatively large d_{max}, we note that an m-bit bit field is sufficient to completely record all m votes since each bit could represent a unique path in the Merkle tree. Furthermore, since we only need to provide a rough bound for the number of votes heard, the bit field representation could be compressed using various lossy sparse-storage directory mechanisms such as the Coarse Vector [13] and Tristate [1] protocols.

7.2 Evaluation of the random keys scheme

Perfect resilience against node capture. Since each pairwise key is unique, capture of any node does not allow the adversary to decrypt any additional communications in the network besides the ones that the compromised node is directly involved in. This would be represented in Figure 2 as the line y=0.

Maximum supported network size. The limited global payoff requirement of Section 5.2.2 cannot be used to compute the maximum network size of the random pairwise keys scheme because global information revealed from local node capture is always 0. Rather, the maximum network size of a random pairwise keys scheme is fixed at design time by Equation 6.

The maximum supportable network size for a random pairwise key scheme without distributed node revocation or multihop range extension is shown in Figure 3. Figure 6 reflects the network sizes for the random pairwise scheme with all the features mentioned earlier including range extension. It can be seen that with a range extension of just two hops, we can get network sizes comparable to the other schemes in this case. Also, the $O(\log m)$ cost of including distributed node revocation does not significantly impact maximum network size.

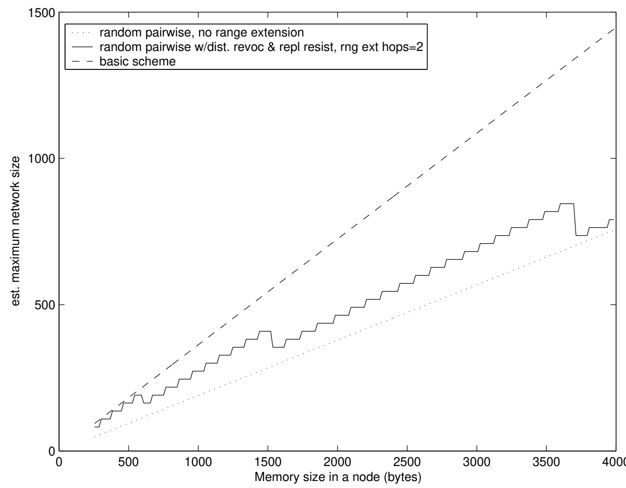


Figure 6. Network sizes for random pairwise key setup compared against the basic scheme with and without multipath key reinforcement. Link keys are 128bits, hash values are 80bits in this simulation. $p = 0.33, f_{threshold} = 0.1$

Given that the random pairwise scheme has perfect resilience against node capture and authentication features, this is a highly desirable result.

Resistance to revocation attack of distributed scheme. If resistance against node replication is implemented, then the theoretical number of nodes an attacker can revoke per successful node captured is $\frac{d_{max}}{t}$ which is kd where k is a small constant. Since any captured node will have an expected d links in any case, the number of nodes lost through the revocations due to a captured node is some small constant factor of the links directly lost through the compromise of the node. Furthermore, d grows only slowly with n, hence the attacker is unable to target a significant portion of the network for revocation if it has only compromised a small number of nodes. While a revocation attack amplifies the disruptive power of the attacker to some extent, it is unlikely that an attacker will find it economically attractive to obtain full control of a sensor node only to expend this successful intrusion in revoking a small constant number of other nodes (if denial of service is all that is desired by the attacker, physical destruction of each node is probably more economical). This is especially true considering that they must explicitly establish communications with every node that they wish to revoke.

8 Related Work

We first review work in establishing shared keys in mobile computing, then review work in sensor network key establishment.

Tatebayashi, Matsuzaki, and Newman consider key distribution for resource-starved devices in a mobile environment [23]. Park et al. [18] point out weaknesses and improvements. Beller and Yacobi further develop key agreement and authentication protocols [4]. Boyd and Mathuria survey the previous work on key distribution and authentication for resource-starved devices in mobile environments [6]. The majority of these approaches rely on asymmetric cryptography. Bergstrom, Driscoll, and Kimball consider the problem of secure remote control of resource-starved devices in a home [5].

Stajano and Anderson discuss the issues of bootstrapping security devices [22]. Their solution requires physical contact of the new device with a master device to imprint the trusted and secret information.

Carman, Kruus, and Matt analyze a wide variety of approaches for key agreement and key distribution in sensor networks [8]. They analyze the overhead of these protocols on a variety of hardware platforms.

Wong and Chan propose a key exchange for low-power computing devices [24]. However, their approach assumes an asymmetry in computation power, that is, one of the participants is a more powerful server.

Perrig et al. propose SPINS, a security architecture specifically designed for sensor networks [19]. In SPINS, each sensor node shares a secret key with the base station. To establish a new key, two nodes use the base station as a trusted third party to set up the new key.

We review the related work by Eschenauer and Gligor [10] in Section 4. Anderson and Perrig propose a key establishment mechanism for sensor networks based on initially exchanging keys in the clear [2]. Their key infection approach is secure as long as an attacker arrives after key exchange and did not eavesdrop the exchange.

Zhou and Haas propose to secure ad hoc networks using asymmetric cryptography [25]. Kong et al. propose localized public-key infrastructure mechanisms, based on secret sharing and multiparty computation techniques [14]. Such approaches are expensive in terms of computation and communication overhead.

Broadcast encryption by Fiat and Naor [11] is another model for distributing a shared key to a group of receivers. However, this model assumes a single sender, and that the sender knows the key pools of all receivers. Subsequent papers further develop this approach [3, 12, 15].

9 Conclusion

Efficient bootstrapping of secure keys is of critical importance for secure sensor network applications. Local processing of sensor data requires secure node to node communication. We present three efficient random key predistribution schemes for solving the security bootstrapping problem in resource-constrained sensor networks.

Each of these three schemes represents a different trade-off in the design space of random key protocols. The choice of which scheme is best for a given application will depend on which trade-off is the most appealing.

The q-composite scheme achieves significantly improved security under small scale attack at the cost of greater vulnerability to large scale attack. This increases the attacker's cost of mounting an attack since the option of harvesting a small number of keys in order to extract a random sample of the readings in the entire network is no longer appealing, thus forcing the attacker to perform a large scale node capture attack.

The (2-hop) multipath reinforcement scheme improves security at the cost of network communication overhead. Since the expected number of common neighbors is proportional to $\frac{1}{n'}$ (where n' is the expected number of neighboring nodes), this scheme performs best when the deployment density is sparse relative to the communication radius of the nodes. It also presents the best characteristics when the variation in deployment density is low (i.e. nodes are regularly dispersed).

The random pairwise scheme has the best security properties of the three schemes. It possesses perfect resilience against node capture attacks as well as support for node-based revocation and resistance to node replication. The properties come with the trade-off that the maximum supported network size is not as large as the other schemes.

10 Acknowledgments

We are very grateful to Virgil Gligor for his helpful discussions with us and his suggestions that helped improve the paper. We are also very grateful to Falk Herrmann for his feedback and discussions. Finally, we would also like to thank the anonymous reviewers for their comments and suggestions.

References

[1] Anant Agarwal, Richard Simoni, Mark Horowitz, and John Hennessy. An evaluation of directory schemes for cache coherence. In *Proceedings of the 15th Annual International Symposium on Computer Architecture*, pages 280–289, 1988.

[2] Ross Anderson and Adrian Perrig. Key infection: Smart trust for smart dust. Unpublished Manuscript, November 2001.

[3] Dirk Balfanz, Drew Dean, Matt Franklin, Sara Miner, and Jessica Staddon. Self-healing key distribution with revocation. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 241–257, May 2002.

[4] M. Beller and Y. Yacobi. Fully-fledged two-way public key authentication and key agreement for low-cost terminals. *Electronics Letters*, 29(11):999–1001, May 1993.

[5] Peter Bergstrom, Kevin Driscoll, and John Kimball. Making home automation communications secure. *IEEE Computer*, 34(10):50–56, Oct 2001.

[6] Colin Boyd and Anish Mathuria. Key establishment protocols for secure mobile communications: A selective survey. In *Australasian Conference on Information Security and Privacy*, pages 344–355, 1998.

[7] Michael Brown, Donny Cheung, Darrel Hankerson, Julio Lopez Hernandez, Michael Kirkup, and Alfred Menezes. PGP in constrained wireless devices. In *9th USENIX Security Symposium*, August 2000.

[8] David W. Carman, Peter S. Kruus, and Brian J. Matt. Constraints and approaches for distributed sensor network security. *NAI Labs Technical Report #00-010*, September 2000.

[9] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22:644–654, November 1976.

[10] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communication Security*, pages 41–47, November 2002.

[11] Amos Fiat and Moni Naor. Broadcast encryption. In *Advances in Cryptology – CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, 1994.

[12] J. Garay, J. Staddon, and A. Wool. Long-lived broadcast encryption. In *Advances in Cryptology — CRYPTO '2000*, pages 333–352, 2000.

[13] Anoop Gupta, Wolf-Dietrich Weber, and Todd Mowry. Reducing memory and traffic requirements for scalable directory-based cache coherence schemes. In *Proceedings of the 1990 International Conference on Parallel Processing (Vol. I Architecture)*, pages 312–321, 1990.

[14] Jiejun Kong, Petros Zerfos, Haiyun Luo, Songwu Lu, and Lixia Zhang. Providing robust and ubiquitous security support for mobile ad-hoc networks. In *9th International Conference on Network Protocols (ICNP'01)*, 2001.

[15] M. Luby and J. Staddon. Combinatorial bounds for broadcast encryption. In *Advances in Cryptology — EUROCRYPT '98*, pages 512–526, 1998.

[16] R. Merkle. Secure communication over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978.

[17] Ralph Merkle. Protocols for public key cryptosystems. In *1980 IEEE Symposium on Security and Privacy*, 1980.

[18] C. Park, K. Kurosawa, T. Okamoto, and S. Tsujii. On key distribution and authentication in mobile radio networks. In *Advances in Cryptology - EuroCrypt '93*, pages 461–465, 1993. Lecture Notes in Computer Science Volume 765.

[19] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. In *Seventh Annual ACM International Conference on Mobile Computing and Networks (MobiCom 2001)*, July 2001.

[20] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[21] J. Spencer. *The Strange Logic of Random Graphs*. Number 22 in Algorithms and Combinatorics. 2000.

[22] Frank Stajano and Ross Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Security Protocols, 7th International Workshop*, 1999.

[23] M. Tatebayashi, N. Matsuzaki, and D. B. Jr. Newman. Key distribution protocol for digital mobile communication systems. In *Advances in Cryptology - Crypto '89*, pages 324–334, 1989. Lecture Notes in Computer Science Volume 435.

[24] Duncan S. Wong and Agnes H. Chan. Efficient and mutually authenticated key exchange for low power computing devices. In *Advances in Cryptology — ASIACRYPT '2001*, 2001.

[25] Lidong Zhou and Zygmunt J. Haas. Securing ad hoc networks. *IEEE Network Magazine*, 13(6):24–30, November/December 1999.