Establishing Pair-wise Keys in Heterogeneous Sensor Networks

Patrick Traynor, Heesook Choi, Guohong Cao, Sencun Zhu and Tom La Porta
Networking and Security Research Center
Department of Computer Science and Engineering
The Pennsylvania State University

***Abstract*— Many applications that make use of sensor networks require secure communication. Because asymmetric-key solutions are difficult to implement in such a resource-constrained environment, symmetric-key methods coupled with a priori key distribution schemes have been proposed to achieve the goals of data secrecy and integrity. These approaches typically assume that all sensors are similar in terms of capabilities, and hence deploy the same number of keys in all sensors in a network to provide the aforementioned protections. In this paper we demonstrate that a probabilistic unbalanced distribution of keys throughout the network that leverages the existence of a small percentage of more capable sensor nodes can not only provide an equal level of security but also reduce the consequences of node compromise. We demonstrate the effectiveness of this approach on small networks using a variety of trust models and then demonstrate the application of this method to very large systems. The approach and analysis presented in this paper can be applied to all protocols that use probabilistic keys including those that employ broadcast mechanisms, hash functions or polynomials for the generation of keys.**

Keywords: Probabilistic Key Distribution, Heterogeneous Sensor Networks, Key Management

I. Introduction

The wide-scale deployment of wireless ad hoc sensor networks is becoming more common. These networks typically cannot rely upon centralized administration or a pre-established infrastructure. Instead, group-wide communication is accomplished through the collaboration of the sensor nodes to form a network. The nodes in the network then forward packets on behalf of each other to desired destinations. This allows for the creation of networks in locations such as wilderness, rural and hostile settings. Because of the often critical nature of the data collected, many sensor network applications require secure communication.

Sensor networks typically consist of a large number (hundreds to a few thousand [1]) of very simple nodes. Commercially available sensors, such as the Berkeley MICA2 mote, are characterized by their limited processing capability (8-bit, 4MHz processor), tiny memory (128 KB program memory) and small size [4]. Because of these minimal resources, the implementation of asymmetric cryptographic algorithms is not efficient on such a platform [7]. Instead, we approach the problem using symmetric-key cryptography. A challenge with this approach is the distribution of shared secret keys to communicating nodes in an environment with limited infrastructure.

Overly simplistic key management approaches do not provide adequate confidence for such a system. For example, using a single key throughout the entire network compromises the communications of all nodes when any one node is captured. The converse, in which each node stores $n-1$ keys (one for each of n nodes in the network), incurs a growth rate of O(n^2) and therefore does not scale to large networks. Lastly, assigning keys specific to an a priori position fails to take into account the potential for mobility or insertion of new nodes, and is not robust against potential node misplacement.

A well received solution that has been extended by several researchers is to distribute a certain number of randomly selected keys in each of the nodes throughout the network [7], [3], [5], [9], [19]. Using this scheme, one can achieve a known probability of connectivity within a network. These previous efforts have assumed a deployment of homogeneous nodes, and have therefore suggested a balanced distribution of random keys to each of the nodes to achieve security. Likewise, the analysis of those solutions relies on assumptions specific to a homogeneous environment.

We propose leveraging the existing hierarchy found in many sensor networks in terms of node capability. Keys are predeployed according to an unbalanced distribution, i.e., deploy far more keys in the more capable nodes, and fewer keys in the less capable nodes. For example, in a military setting, many simple sensor nodes may be deployed in a field of operation along with a small set of more powerful, more secure nodes, perhaps in attended vehicles. We show that, for many communication scenarios, this solution leads to more secure systems while preserving the same levels of connectivity.

In this paper we make the following contributions: First, we introduce the concept of unbalanced random key predeployment and derive probabilities for achieving a key match between neighboring nodes. Second, we define communication models in a sensor network with heterogeneous nodes and evaluate the connectivity of these networks against trade-offs of the number of keys required in each node; the evaluation is through several mathematical models and supporting simulation. Third, as part of the communication models and evaluation we define various network trust models to reflect the requirements of various applications. We evaluate the robustness of our scheme for a number of communication and trust models in which both the size of the network and the confidence in our neighbors varies. We then discuss the application of the combination of the trust models and

unbalanced key distribution schemes to large scale sensor networks. Fourth, we evaluate the benefits of the unbalanced key distribution strategy in terms of memory requirements and resiliency in the face of node compromise when compared to a balanced key distribution strategy [7]. We also give a brief analysis for the application of the unbalanced method to the q-Composite scheme [3]. Finally, we simulate and analyze the results of both balanced and unbalanced key distribution schemes in an environment in which the transmission range of the nodes is smaller than the area covered by the neighborhood of nodes.

The method and analysis presented in this paper may be applied to any protocol that effects probabilistic keying, including broadcast mechanisms [7], [3], hash functions [19] or polynomials [2], [9] to generate keys.

The rest of this paper is organized as follows: in Section II we provide a brief review of related work and an overview of the unbalanced key distribution strategy; in Section III we develop the analysis for network connectivity; in Section IV we present both analytical and simulation results comparing key distribution strategies, including connectivity probabilities and average path lengths required to establish session keys; in Section V we discuss the results, including a detailed discussion on the impact of the compromise of one or more sensor nodes.

II. Overview of Unbalanced Method

In this section, we present a basic overview of previous work. We then introduce our network, trust and key distribution models, the latter of which is based upon the previous work of Eschenauer and Gligor [7]. Results are discussed in Section IV.

A. Related Work

Previous work on pre-deployment of keys in sensor networks has assumed all nodes are of equal capability, and hence distribute an equal number of keys to each node [7]. In this scheme, a large pool of P keys is generated, from which k are randomly selected, without replacement, for each sensor node. Two nodes may communicate to directly establish a session key if they have a key match. The probability that two nodes with the same number of random keys, k, share at least one key is:

$$P[Match] = 1 - \frac{((P-k)!)^2}{P!(P-2k)!} \text{ [7]} \quad (1)$$

If nodes do not have a key match, they may still establish a session key through one or more intermediate nodes with which they each have a common key [12].

The basic protocol supports key distribution, key revocation, and re-keying. These components are largely re-used in other work, including our own. We summarize them here for readability; please refer to Eschenauer, et al [7] for details.

1) Key Distribution/Initialization: The key distribution protocol has three phases - key pre-distribution, shared-key discovery and session-key establishment. The key pre-distribution phase takes place offline prior the deployment of a sensor network. In summary, k keys randomly selected from a pool of P keys are deployed in each sensor node.

The *shared-key discovery phase* begins upon network deployment. Sensor nodes determine their neighbors, and through a protocol as simple as the clear text broadcast of the key identifiers, discover which of these neighbors can communicate securely with the initial keys. It is also possible to implement more secure methods of determining the keys of neighbors including the use of hash functions [19], encrypted broadcasts [7], [3], and the use of polynomials [2]. A shared key is established between two nodes when the receiving sensor obtains a response to a challenge issued using the identifier of the shared key from the initial broadcast.

The *session-key establishment phase* attempts to create a secure communication link for pairs of nodes within wireless transmission range that lack a shared key from the previous phase. If it is possible to communicate between a pair of nodes by using a multi-hop path through secure and trusted neighbors, then a key can be generated at one of the endpoints and passed through this path such that the two end nodes can communicate directly. At the completion of this final phase, all nodes within transmission radius of each other should be able to communicate directly to whatever reliability the network designers have specified.

2) Revocation of Keys: When the compromise of a node is detected, it is important to remove those keys associated with the captured entity from all other nodes in the network. Methods to accomplish this goal can take a number of forms including a signed revocation message from some controller node or the result of a distributed election algorithm. For the purposes of this work, we will assume that some revocation method exists but do not specify its mechanics.

3) Re-keying: In addition to normal key expirations, in a setting where the network topology is rapidly changing due to high mobility or a high number of node failures or additions, it may be necessary to reestablish session-keys when a node's level of connectivity to its neighbors falls below a certain threshold. Connections can simply be established by running both the shared-key discovery and session-key establishment phases.

Since this original work, several other variations of this scheme have been suggested to strengthen this method. In Chan et al. [3], the idea of requiring q keys to match between neighbors, as opposed to one, is proposed to make it more difficult to compromise communications. Additionally, session keys can be established over multiple disjoint paths to reduce the probability of an intruder gaining a key. The authors also propose allowing keys to be distributed to nodes in pairs so that keys may be associated with specific nodes, thus allowing authentication.

Other authors propose different mechanisms for storing or generating keys to reduce memory requirements [5], [13]. In Zhu, et al. [17], the authors propose an erasure-based pre-deployment method to create secure channels over which

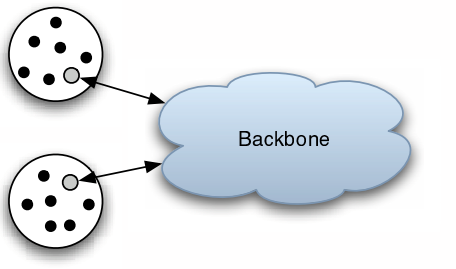

Fig. 1. A new model for sensor networks. Above, groups of nodes are divided into neighborhoods by their natural surroundings (mountains, buildings, etc). Inter-neighborhood communication is possible by sending messages from the sensing/Level 1 nodes through a gateway/Level 2 node (gray), which may have access to a backhaul to a backbone network.

group keys are distributed. Du, et al. [6] leverage a priori knowledge of the deployment of nodes (position) to further improve on the efficiency of pre-deployment schemes. Perrig, et al. leverage a modified version of the TESLA [10] authenticated broadcast protocol in SPINS [11], a suite of security protocols for sensor networks.

Eschenauer, et al. [7] leverage the fact that the network is homogeneous and large to derive equations to determine the overall connectivity of a network. Because our new network model violates many of the assumptions used in previous work to establish network connectivity, the majority of the equations used to describe the system cannot be used. In the following subsection we examine a new network model.

B. Network Model

The previous work on random key pre-deployment in sensor networks has assumed either a grid or very large random-graph arrangement such that all neighbors within the transmission radius of a given node are reachable. Communication between adjacent nodes is therefore limited only by key matching. This model is not always realistic for a number of reasons. Primarily, it fails to take in to account that signal-blocking barriers including hills, buildings and walls may exist between neighboring nodes. Because the assumptions of topology and topography used in most previous approaches are violated in realistic settings, we propose a new network model.

Figure 1 shows a network model with two main differences from that used in previous work. First, the landscape over which a sensor network is placed contains features that segregate nodes into exclusive neighborhoods. Because nodes are still distributed through the same methods discussed in previous papers (e.g. dropped from an airplane), there is no way to determine a node's neighbors a priori. Due to the random nature of the deployment, the potential for node mobility and addition of nodes at a later time, assigning keys for specific neighbors is not possible. Second, instead of a homogenous composition of nodes, the network now consists of a mix of nodes with different capabilities and missions. The sensing or Level 1 (L1) nodes are assumed to be very limited in terms of memory and processing capability, and perform the task of data collection. Level 2 (L2) nodes have more memory and processing ability. These nodes are equipped with additional keys, and take on the role of routers and gateways between networks. In addition to tamper-resistant casings [8], L2 nodes are assumed to be equipped with a fast encryption/deletion algorithm to protect their supplementary keys from compromise if they are captured. In Section V-B, however, we relax this assumption and examine the effects of L2 compromise.

In this model we consider two cases. First, each node is able to communicate with all nodes in its isolated neighborhood by direct transmission if both possess a common shared key for communication. We then consider cases in which the transmission radius of a node may be less than the radius of the neighborhood. In either case, communication with nodes outside of a local neighborhood is possible only by sending messages through a "gateway" L2 node which has access to a backhaul for some backbone network.

C. Keying and Trust Models

The previous work in this area has assumed that, unless compromised, a node can always trust its neighbors as long as a secure relationship can be established. In this paper, we have created three keying and trust models to which both the balanced and unbalanced key distribution schemes are applied:

- **Backhaul** - In this scenario, an L1 node only trusts an L2 node in terms of both sharing data and establishing session keys. Accordingly, L1 nodes will send data only to an L2 node, and will only establish a session key if it has a direct key match with the L2 node.
- **Peer-To-Peer with Limited Trust** - Two L1 nodes wish to share data and must therefore establish a session key. They trust only each other, or neighboring L2 nodes. Because the L1 nodes are neighbors, they can communicate the data directly. However, if they do not have a direct key match, the assistance of neighbors becomes necessary. Because their trust in their neighbors is limited, they will only use an L2 node to assist in establishing a session key.
- **Peer-To-Peer with Liberal Trust** - This scenario is similar to the limited trust scenario except that L1 nodes will trust other L1 nodes in the neighborhood in addition to the node with which they will be sharing data. Again, because the two L1 nodes wishing to correspond are neighbors, they can communicate directly. If they cannot directly establish a session key, they will use either the L2 nodes or other L1 nodes in the neighborhood with which a key is already shared to assist in indirect key establishment. We consider cases in which the number of nodes allowed to assist in key establishment is either unlimited or limited to one.

Inter-neighborhood communication via the backbone is assumed to be secure.

Lastly, nodes assisting in the indirect establishment of keys are assumed to immediately delete the session keys generated or learned through this process. Source and destination nodes also rekey so as to further avoid the effects of compromise. A full discussion of this topic is given in Section V.

III. Analysis of Connectivity

The following section discusses the methods and associated overhead for establishing pair-wise keys with a node's neighbors. The goal of these protocols is to establish a direct, secure link between neighboring source and destination nodes. In so doing, this solution for key distribution allows systems to operate securely while minimizing the security burden placed upon the least capable nodes in the system. In the following, n is the number of L1 nodes in a neighborhood, and g is the number of L2 nodes in a neighborhood, where applicable.

A. Unbalanced Key Distribution

Our scheme for the unbalanced distribution of keys throughout a wireless sensor network builds upon the previously described balanced approach of Eschenauer, et al [7]. Given the same generated key pool of size P, we store a key ring of size k keys in each sensor (L1) node, and a key ring of size m keys in each L2 node, where $m \gg k$. The equation for the probability of an L2 and L1 having at least one key in common is given by Eq. 2.

$$P[Match] = 1 - \frac{(P-k)!(P-m)!}{P!(P-m-k)!} \tag{2}$$

The full derivation of the above equation is available in Appendix A. Using Eq. 2, the sizes of k and m can be varied according to the preference of the system designer.

B. Determining Connectivity

We now determine network connectivity for each of the three trust models.

1) Backhaul Connectivity: To enable backhaul communication, a key match must exist between an L1 node and an L2 gateway node. In cases of a single L2 node being present in a neighborhood, we simply apply Eq. 2. If more than one L2 node is present in a neighborhood, two cases exist. In the first, all of the L2 nodes may act as gateways. In the second, only one L2 node will serve as a gateway to the backbone network. In this case, an L1 node may have a key match with the L2 gateway node, or because L2 nodes are trusted, may establish a session key through a path of one or more of the other L2 nodes.

Because the number of keys in the L2 nodes is very high, both theory and practice demonstrate that L2 nodes are able to communicate with each other with a far greater than five-nines reliability; therefore secure links between L2s will always be assumed. In either case, the probability that any L1 node can communicate with the gateway is:

$$P[Conn] = 1 - (1-U)^g \tag{3}$$

where U is defined by Eq. 2.

2) Peer-To-Peer with Limited Trust: In this case, an L1 node desires to communicate with another L1 node in its neighborhood. To do this, it either requires a direct key match with the peer L1 node, or it must be able to establish a session key through a path of one or more L2 nodes. Because L2 nodes are assumed to be more secure due to the presence of key-protecting algorithms, all L2 nodes can be used. The probability of connectivity is therefore:

$$P[Conn] = 1 - (1-B)(1-(1-(1-U)^g)^2) \tag{4}$$

B is defined by Eq. 1, U by Eq. 2 and g is the number of neighboring L2 nodes. A detailed derivation is available in Appendix B.

3) Peer-To-Peer with Liberal Trust: When using Peer-To-Peer with Liberal Trust to establish connectivity, in addition to using a path of L2 nodes to establish a session key, L1 nodes may establish session keys using a path through a sequence of L1 nodes that provide mutual key matching. Previous studies have considered the use of up to $n-1$ hops for the establishment of a session-key. Allowing such a possibly long path to establish session keys has the following two drawbacks. First, the potential latency caused by allowing up to $n-1$ hops for initializing secure communication may be unacceptably high for the network to complete its mission, especially in cases of high mobility in which nodes may be required to establish new session keys often in the middle of active communication. Second, the introduction of multiple hops may increase the chance that compromised but undetected intermediaries are able to eavesdrop on the actual session-key establishment. While not in the direct path of future packet exchanges, compromised adjacent nodes that assisted with keying may still be able to overhear and decrypt the communications of uncompromised neighbors. Because each hop along an indirect keying path is able to decrypt the data from the previous hop, the probability that at least one compromised node is in a path increases as that path becomes longer.

To estimate the level of connectivity in networks which allow an unlimited number of hops to initialize a session key, we make use of an observation from our simulations. Given a neighborhood of at least 10 nodes and sufficient keys to achieve a relatively high level of connectivity (three-nines), nodes with at least one connection to a neighbor are fully connected to their neighbors. Over the course of 10,000 simulations, we recorded that partitions in the network were limited in size to single nodes. Therefore, for networks with balanced key distributions, Eq. 5 may be used as an approximation for network connectivity when there is no hop limit for establishing keys.

$$P[Conn] = 1 - (1-B)^{n-1} \tag{5}$$

where B is defined in Eq. 1. This equation simply represents the probability that a node is not isolated.

To estimate the connectivity when limiting the number of hops through which a key may be established to one, we determined the expected number of nodes that any L1 node can communicate with either directly or through one hop. For networks with balanced key distributions, this value is given by Eq. 6.

$$\begin{aligned} E^1_{L1} \;=\; & B*(n-1) \\ & +\sum_{i=1}^{n-1} \binom{n-1}{i} B^i (1-B)^{n-1-i} E_1 \end{aligned} \tag{6}$$

where B is defined by Eq. 1 and $E_1 = (n-1-i)(1-(1-B)^i)$. For networks with unbalanced key distributions, the value is given by Eq. 7.

$$\begin{aligned} E^1_{L1} \quad &= E_0 + \sum_{k=1}^{g} \binom{g}{k} U^k (1-U)^{g-k} (1-B)^{n-1} E_1 \\ &+ \sum_{k=1}^{g} \binom{g}{k} \sum_{i=1}^{n-1} \binom{n-1}{i} U^k (1-U)^{g-k} B^i (1-B)^{n-1-i} E_2 \\ &+ \sum_{i=1}^{n-1} \binom{n-1}{i} B^i (1-B)^{n-1-i} (1-U)^g E_3 \end{aligned} \tag{7}$$

where ,

$$\begin{aligned} E_0 &= B * (n-1), \\ E_1 &= (n-1)(1-(1-U)^k), \\ E_2 &= [(n-1-i)(1-(1-U)^k)] \\ &+ \quad [(n-1-i)(1-U)^k(1-(1-B)^i)], \\ E3 &= (n-1-i)(1-(1-B)^i). \end{aligned}$$

The derivations of Eqs. 6 and 7 are given in Appendix C.

Given E_{L1}, the expected connectivity of the neighborhood is calculated by Eq. 8.

$$P[Conn] = \frac{E^1_{L1}}{n+g-1} \tag{8}$$

C. q-Composite Analysis

As mentioned in Section II-A, Chan et al. [3] extended the basic scheme by requiring nodes to share at least q keys with each other. In so doing, it becomes more difficult for an attacker to compromise communications as q instead of one key must first be captured. With the balanced scheme, however, an increase in q leads to a significant increase in the number of keys stored by all L1 nodes. The unbalanced approach can reduce the burden of the q-Composite scheme on L1 nodes while retaining its security advantages.

Eq. 9 gives the probability that two nodes containing key rings of differing sizes share exactly i keys is:

$$p(i) = \frac{\binom{P}{i}\binom{P-i}{(m-i)+(k-i)}\binom{(m-i)+(k-i)}{m-i}}{\binom{P}{m}\binom{P}{k}} \tag{9}$$

The probability that two nodes share at least q keys with each other is therefore:

$$1 - \sum_{i=0}^{q-1} p(i) \tag{10}$$

A full derivation of the above is given in Appendix D.

IV. Experimental Results

In the following simulations, the default number of neighboring nodes is set to 40 such that a direct comparison to previous work [7], [6], [9] can be drawn. Simulations with 60 neighbors [3] yield similar results due to the low probability of direct match between L1 nodes.

The simulator itself was written in C++ and designed only to determine whether or not keys could be established between neighboring nodes given a specific trust model. No communications protocols were directly incorporated; rather, nodes directly compared their pseudo-randomly assigned keys (direct key establishment) and keyed neighbors (indirect key establishment). Accordingly, any of the methods for determining the key IDs held by neighboring nodes including via broadcast or hashing could be implemented on top of this system. In both small and large network simulations, 12.5% of the nodes in the network were designated as L2s.

The simulations for small networks, detailed below in Section IV-A, contain a total of 40 nodes, all of which are within range of each other. Keys and their associated identifiers are assigned as discussed in Section III so as to simulate the random composition of a neighborhood. In the simulations for large networks, 1,000 nodes were placed randomly throughout a grid such that the average number of neighbors was 40. In order to ensure that all nodes had an equal chance of being connected, nodes placed on the edges of the network increased their transmission radii to raise their average number of neighbors to 40 as well. This could have instead been accomplished with the implementation of a mobility-based coverage scheme [15]. Nodes first established connectivity with their neighbors under the limited trust model and then attempted to see if they could reach all other nodes throughout the network.

A. Small Sensor Networks

In this subsection we determine the number of keys required for the balanced and unbalanced key distribution strategies to achieve targeted network connectivity for the various communication and trust models using equations 1-8.

Extensive simulation was carried out to verify the validity of these equations. Each simulation uses a sensor network of 40 nodes, each within direct transmission range of all other nodes so as to establish similar conditions to previous papers. Further experiments discussed in Section IV-C were then conducted by varying the transmission radii of nodes. For the unbalanced scenarios, unless otherwise noted, five L1 nodes are replaced with five L2 nodes. All cases were run an order of magnitude more times than the minimum requirement to determine connectivity (10,000 iterations for 3-9s, etc). Lastly, data about which keys were used to connect nodes was recorded to better understand the effect of a compromise.

1) Backhaul: In Figure 2, we plot the number of keys required in L2 nodes versus the number of L2 nodes in the network for various values of keys in L1 nodes to achieve five-nines connectivity using Eq.3. This chart illustrates the flexibility in engineering the number of keys deployed when using the unbalanced key distribution strategy. When using balanced key distribution, all nodes are required to have 328 keys to ensure five-nines connectivity. Compared to the 30 keys required in L1 nodes when 711 keys are deployed in L2 nodes, a potentially large savings in keys is illustrated when the unbalanced key distribution is implemented.

For this case, we verify Eq. 3 by comparing it against simulations considering a scenario in which 30 keys are

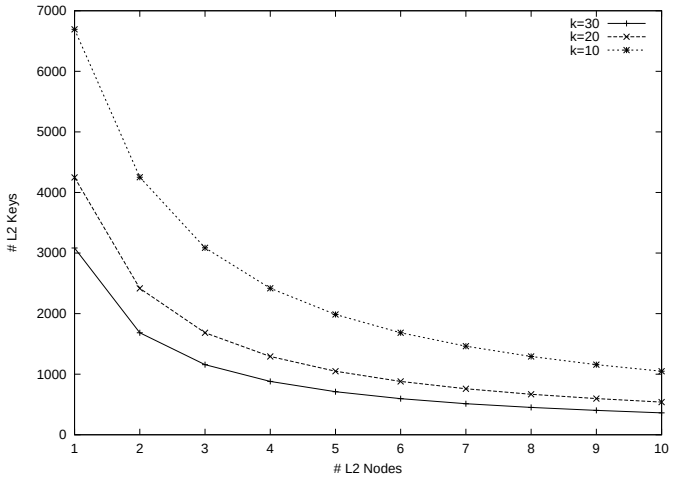

Fig. 2. Theoretical values for the number of keys in L2 nodes to establish 5-9s connectivity for the backhaul case with 10, 20 and 30 keys in each L1 node.

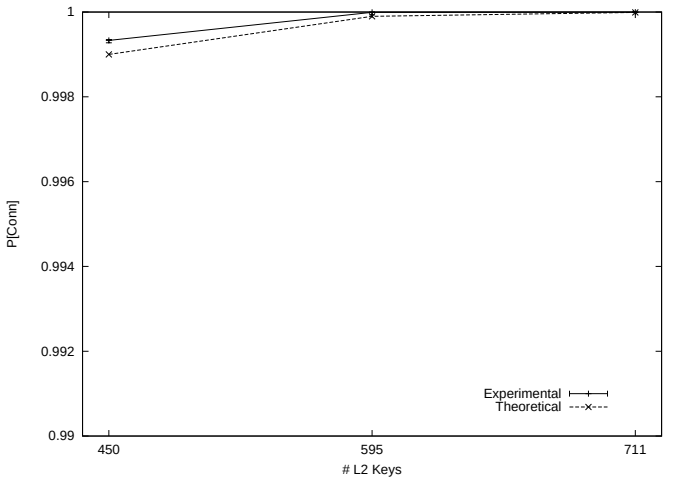

Fig. 3. A comparison of the theoretical and experimental results for the backhaul case. The simulations for the remaining trust models yielded similar results.

deployed in L1 nodes, and the number of keys deployed in L2 nodes is varied. These results are shown in Figure 3. For the three cases tested, the number of keys deployed in the L2 nodes is 450, 595, and 711, corresponding to an expected level of connectivity of three-, four-, and five-nines, respectively, according to Eq. 3. In all three cases, the average values obtained by the simulations are within 0.03% of those calculated with Eq. 3.

2) Peer-To-Peer with Limited Trust: Figure 4 shows the number of L2 keys required to achieve five-nines connectivity, versus the number of L2 nodes deployed in the network for various values of keys deployed in L1 nodes according to Eq. 4. When using balanced key distribution, the number of keys required in each node is 328. Again, we see a large savings in terms of number of keys deployed in L1 nodes when using unbalanced key deployment.

To verify Eq. 4, we ran simulations with 30 keys provisioned in L1 nodes, and varied the number of keys in five L2 nodes. The number of keys in each L2 node is 465, 620 and 750 keys, corresponding to connectivity levels of three-, four- and five-

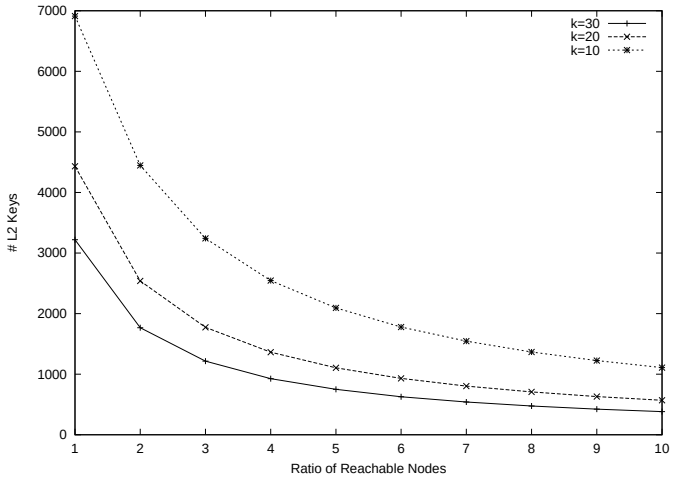

Fig. 4. Theoretical values for the number of keys in L2 nodes to establish 5-9s connectivity for the P2P limited-trust case with 10, 20 and 30 keys in each L1 node.

TABLE I
NUMBER OF REQUIRED KEYS, 40 NODE NETWORK WITH 5 L2S

Scenario	Unbalanced		Balanced
	L2 Keys	L1 Keys	L1 Keys
Backhaul	711	30	328
P2P Limited	750	30	328
P2P Liberal, 1-hop	689	30	83
P2P Liberal, n-hop	515	30	54

nines, respectively. The average difference obtained through simulation was within 0.05% of that obtained by Eq. 4.

3) Peer-To-Peer with Liberal Trust: In this scenario we examined peer-to-peer communication models with liberal trust, including both cases in which the path length is limited to one hop and in which the path length is unlimited.

For the 1-hop case, the values obtained from Eq. 6 and Eq. 7 were verified using simulation. The average differences obtained by the simulations are within 0.05% of those obtained from Eq. 6 and Eq. 7. For the balanced case, each L1 node requires 83 keys; for the unbalanced case each L1 node requires 30 keys if five L2 nodes are deployed, each with 689 keys.

For the unlimited hop case Eq. 5 was verified via simulation. The results obtained by Eq. 5 were within 0.05% of the average simulation results. Using Eq. 5, for the balanced case, each L1 node requires 54 keys to achieve five-nines connectivity; in the unbalanced case, to achieve five-nines connectivity, each L1 node requires 30 keys if five L2 nodes, each with 515 keys, are deployed.

From these results, we make the following observation: because the number of possible paths through the network is high when the path length is unlimited, even with a small number of keys in L1 nodes, there is a high probability of finding at least one path over which a session key may be established. In this scenario, there is not a large benefit to using the unbalanced key distribution solely to limit the number of keys in L1 nodes. Even in the case in which the path length is limited to a single hop, there is limited benefit in terms of number of keys deployed when using the unbalanced

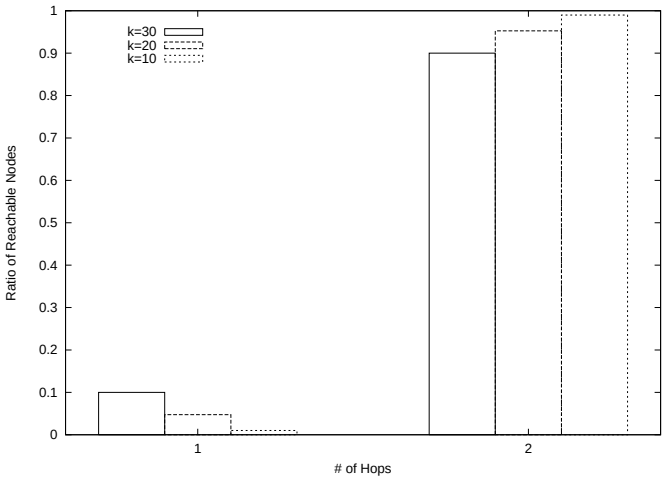

Fig. 5. Path length to neighboring nodes using the unbalanced key distribution scheme with limited trust for large sensor networks.

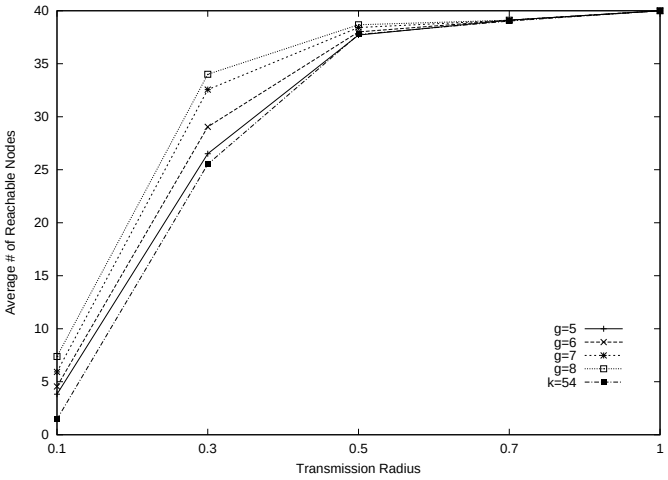

Fig. 6. The effects of altering the transmission radius of randomly positioned nodes as a percentage of the size of their neighborhood on the ability of any node to communicate across the network.

distribution strategy; however, as discussed in Section V, there are still benefits in terms of resiliency.

Table I shows a summary of the number of keys required in L1 and L2 nodes for the balanced and unbalanced key distribution strategies to achieve five-nines network connectivity. For the backhaul and limited trust cases in which key matches with specific nodes are essential, the unbalanced key distribution strategy provides large savings in terms of the number of keys deployed in L1 nodes. Notice that the limited trust case takes more keys than the backhaul case because messages are required to travel an additional hop. For the liberal trust cases in which more potential session key establishment paths exist, the savings is not as great; however, as illustrated in Section V-B, there is still a benefit in terms of resiliency in the face of compromised nodes.

B. Large Sensor Networks

In order to verify connectivity across a large network, we ran a simulation with 1,000 nodes 20 times, each iteration with different seeds for the random number generator. Like the previous simulations, we randomly designated 12.5% of the nodes as L2s and the remaining portion as L1 nodes. Each L1 and L2 node is equipped with 30 and 750 keys, respectively. This distribution of keys showed in both simulation and theory to connect a small (40 nodes) network with five-nines confidence. In all 20 iterations of the simulation, each of the 1,000 nodes was able to establish a path between themselves and all other nodes in the network.

The average number of hops for L1 nodes to establish a session-key with a neighbor node was then analyzed. Previous work has shown the balanced key distribution strategy results in the need for up to five hops to establish a key match [7] under these circumstances. In this case, the path over which the key match takes place often exits the 40-node neighborhood (i.e. transmission radius) of the source and destination, and then re-enters the neighborhood.

For the unbalanced scheme, we tested peer-to-peer with limited trust, which restricts the total number of hops to a maximum of two. Figure 5 summarizes these results. It can be seen that as the number of keys in an L1 node decreases from 30 to 10, it becomes far more likely that it will take two hops to establish a session key. This is because as the number of keys in an L1 node diminishes, it is less likely that there will be an L1-L1 key match, and hence a hop through an L2 node is required. It should be noted that this hop count limit also exists in small sensor networks.

There are several benefits to limiting the number of hops required to establish a key. First, the latency required to initialize communication is lowered. Second, the number of messages exchanged is reduced, limiting network congestion. Because of extremely limited bandwidth [4], network initialization time can greatly be reduced by such a reduction in messaging [14]. Finally, the number of nodes that can learn the session key are limited, thus reducing the chance of a compromised node gleaning useful information.

C. Networks with Varying Transmission Radius

All previous work, including the results presented so far in this paper, has assumed that all nodes within a neighborhood are within transmission range of each other. We now evaluate the impact on having smaller transmission radius than neighborhood radius. In these simulations we considered networks with balanced and unbalanced key distribution. In both cases we considered neighborhoods of 40 nodes, and varied the transmission range of a node between 10% and 70% of the neighborhood radius. We set number of keys in each node to be sufficient to give five-nines connectivity in a neighborhood of 40 nodes that are all within transmission range. For the balanced case this resulted in 54 keys/node. For the unbalanced network we considered cases of 5-8 L2 nodes in the network.

The results are shown in Figure 6 and indicate that with a small transmission radius, network connectivity is poor. In effect, the neighborhood has been reduced to only a few nodes. For the unbalanced case the probability of an L2 node being within range of an L1 node is small; because L1 nodes have so few keys in this case, there is a very low probability that they can securely connect to the network. As the radius

TABLE II
OVERVIEW OF COMPROMISE EFFECTS.

	Connectivity of Compromised Node(s)		
	Unbalanced (m=750, k=30)		Balanced (k=83)
# Nodes Compromised	L2 Compromised Connectivity	L1 Compromised Connectivity	L1 Compromised Connectivity
0	0	0	0
1 L2 or 1 L1	0.904	0.086	0.501
2	0.992	0.165	0.752
3	0.9995	0.238	0.878
4	0.99998	0.304	0.940
5	0.999999	0.365	0.971
Key Pooling Results	4 L1's and 1 L2		5 L1's
	0.9207		0.9708
Random Compromise	0.18588		0.501

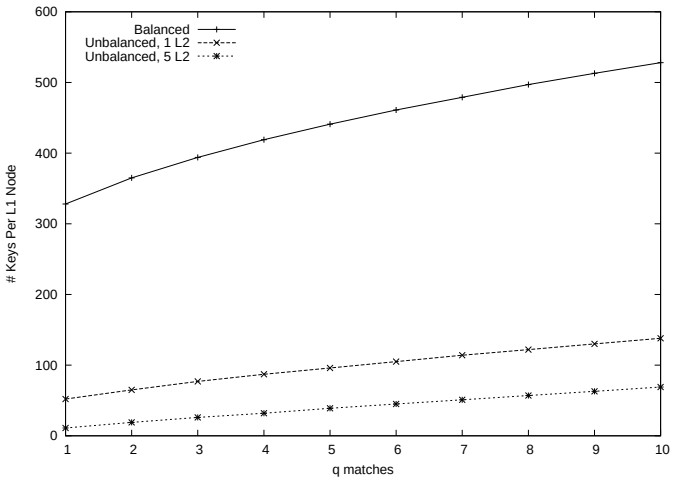

Fig. 7. A comparison of the number of keys that must be stored in L1 nodes for the P2P Limited model. In the unbalanced scenarios, the number of keys stored in an L2 node is set to 2000. Additionally, the change in L1 keys is compared against the presence of one and five L2 nodes. Notice that the rate of growth in L1 keys for both unbalanced cases is less than half that of the balanced.

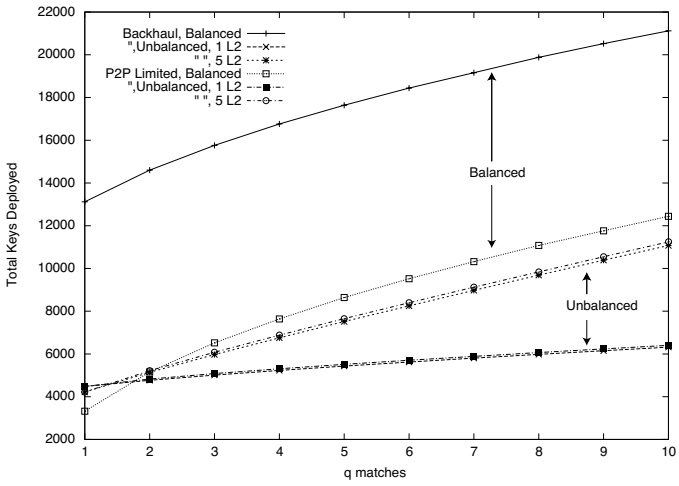

Fig. 8. The total number of keys deployed in a 40 node neighborhood for the backhaul and P2P Limited trust scenarios with varying values for q. The unbalanced cases place 83 keys in the L1 nodes and alters the number of keys in the L2 nodes.

grows, the unbalanced network provides higher connectivity than the balanced network until, at higher transmission radii, the performance converges. These results illustrate the need to understand the neighborhood size in terms of transmission radius in order to deploy an adequate number of keys to achieve target connectivity.

D. q-Composite Scheme

The application of the unbalanced keying method to the q-composite scheme [3] yields a number of benefits.

Figure 7 compares the number of keys that must be stored in L1 nodes in order to implement varying values of q for the balanced and unbalanced key distributions. For a direct comparison to be made, the number of keys in each L2 node is fixed at 2,000. Not only is the number of keys that must be stored in each L1 node significantly smaller when using unbalanced key distribution ($1/6 - 1/4$), but the rate at which keys increase in direct proportion to q is less than half that of the balanced scheme. The balanced scenarios demonstrate a significant increase in the memory burden of the most constrained nodes in the system. Cases in which only a single L2 and up to five L2s working together show that these nodes, which can easily store more keys due to increased resources, are able to help a network retain the advantages of the q-Composite scheme while reducing its increased requirements.

Decreasing the number of keys stored in each of the L2 nodes yields similar results, making the balanced case represent the ceiling scenario for L1 key storage.

Figure 8 gives further insight into the neighborhood-wide savings from the unbalanced approach. All L1 nodes in the unbalanced scenarios received 83 keys, which was chosen to demonstrate that the q-Composite model can be implemented using the same number of keys required for the basic, balanced approach. Barring a network where $q = 1$, fewer total keys are distributed throughout a neighborhood using the unbalanced over the balanced mechanism for implementing the q-Composite scheme. Globally, fewer total resources are therefore necessary to implement this scheme.

V. NODE COMPROMISE ANALYSIS

In the following subsections we calculate and evaluate the impact of node compromise. Recall that all communicating nodes establish a new key per session, even if they have a direct key match. Further, all nodes that assist in establishing session keys during the indirect key matching phase delete the

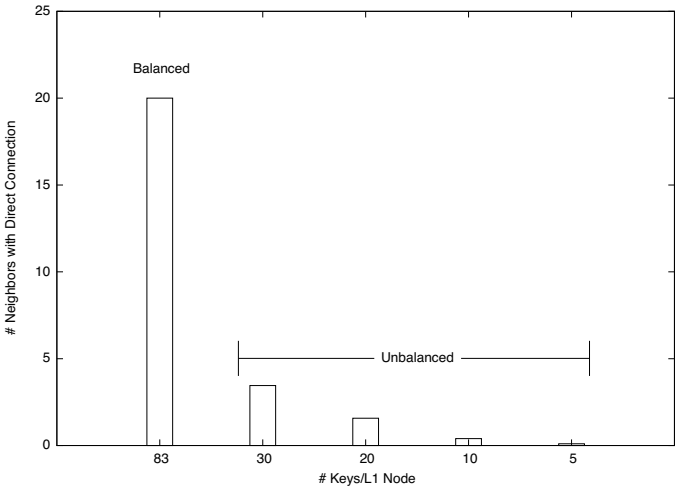

Fig. 9. Average number of neighbors with which a single compromised node can communicate without using a L2 node given a neighborhood of 40 neighbors (calculated via Eq. 2) The results demonstrate that the probability of establishing a session key without the assistance of L2 nodes is unlikely.

session keys they assign once key initialization is complete; in addition, the communicating end points re-key immediately. Therefore, compromised nodes cannot eavesdrop on any ongoing communication other than that which is directly terminated on the compromised node. The greater danger is that these compromised nodes may establish new session keys to inject false data into the network, or assist in establishing new session keys for neighbors and be able to eavesdrop on new sessions. The magnitude of these dangers is proportional to the number of nodes with which the compromised nodes may establish session keys. Accordingly, we analyze the ability of compromised nodes to communicate with their neighbors. The results are summarized in Table II.

In the following we analyze the cases of peer-to-peer with liberal trust limited to one intermediate node, and deploy all nodes with the keys necessary to achieve 0.99999 connectivity under normal conditions.

A. Level 1 Node Compromise

Due to the physically insecure nature of sensor networks, it is crucial to recognize the potential for nodes being interfered with or captured. For reasons of economics, L1 nodes are the least likely to contain sophisticated, tamper-resistant components. Accordingly, because the detection of node compromise and subsequent revocation take time if they occur at all, the consequences of exposed keys from the compromise of an L1 node are investigated.

Properly functioning nodes not only establish session keys with all of their neighbors but also delete all session keys generated to assist adjacent nodes with indirect keying. In so doing, a sensor network built under these requirements effectively implements "perfect resilience" to compromise in that the ongoing direct communications between other nodes are secure from a captured node. The damage done by an enemy in this case is limited to the injection of data into the network or in learning new session keys being established. It is therefore more important to determine how well connected a compromised node can become in order to fully appreciate the effects of node capture. Connectivity is a good measure of the amount of false traffic a node may insert into the network and how capable it will be to assist in the establishment of new keys.

While the actual process of compromise detection may not yet be an exact science, there are a number of characteristics [16], [18] that can be used to indicate such an event. In a network implementing the balanced scheme, it may be difficult to determine that a compromise has occurred due to the distributed nature of authority; however, because a large portion of key establishments must transpire with the assistance of an L2 node when using the unbalanced scheme, the number of opportunities to detect a compromise is greatly increased as a single node is given access to a greater pool of behavioral data. It would therefore be advantageous for the attacker to attempt to establish keys directly with its L1 neighbors and avoid using L2 nodes. Figure 9 shows the difficulty of starting sessions without involving an L2 node.

In Figure 9, the average number of L1 neighbors with which a compromised L1 node has a direct key match in a network of 40 nodes is shown. The number of keys in each node in this figure corresponds to that required to achieve five-nines connectivity in a peer-to-peer network with liberal trust and a limit of one-hop to establish session keys. The case of 83 keys deployed in L1 nodes corresponds to balanced key distribution; the other cases correspond to unbalanced key distribution. The results show that the probability of establishing a session key without the assistance of L2 nodes is unlikely. In all of the unbalanced cases except for $k = 30$, the average number of neighbors with which a direct connection is initially established is approximately equal to or less than one.

Therefore, in order to achieve secure connections with its neighbors, a compromised L1 node would almost certainly have to communicate through an L2 node. Because these nodes have access to a much larger number of keying requests, patterns indicating irregular keying requests are more likely to be detected.

B. Level 2 Node Compromise

Because of the increased resources available to L2 nodes, it is reasonable to assume that these devices are also equipped with tamper-resistant mechanisms. Regardless of the reliability of these protective components, no system is absolutely insulated from the possibility of compromise. We therefore examine the unlikely occurrence of an L2 node being captured by an adversary.

Due to the number of keys in its memory, the compromise of an L2 node is potentially more serious. If an attacker is able to devise a way to specifically identify and compromise L2 nodes, the worst case analysis of the proportion of neighbors with which keys can be established is 0.904, 0.992, 0.9995, 0.99998 and 0.999999[1] in the case of peer-to-peer liberal trust limited to one hop. Clearly, the compromise of even a single

[1]One to five compromised L2 nodes each with 750 keys, respectively, from Eq. 2

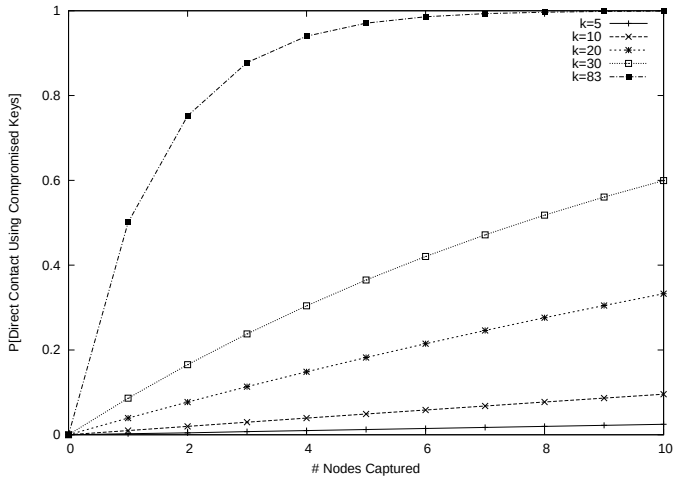

Fig. 10. The worst-case scenario for undetected multiple node compromise. As each node is compromised, its keys are added into a large pool which can be used to eavesdrop on other secure communications throughout the network.

L2 node may compromise a large portion of a network if its contained keys are not quickly revoked.

However, if the appearance of L1 and L2 nodes is made similar and there is no easy way to differentiate between them after the initialization phase, then an attacker would randomly choose a node to compromise. Unless a traffic analysis attack were to occur during the network initialization period, the flow of traffic across could not be used to reveal node capabilities.

In a network with five L2 nodes out of 40, the calculated average fraction of nodes with which a key can be directly established given a *random* single node compromise is 0.18588. For the balanced approach this value is significantly higher, 0.501. These values were generated through Eqs. 1 and 2. We observe that the network using the unbalanced key distribution is more robust to random node capture.

Like their L1 neighbors, L2 nodes delete session keys generated while assisting adjacent nodes in the process of indirect keying. In addition, if L1 nodes re-key amongst themselves after receiving a session key from an L2 node, the compromise of that L2 node only compromises the communications directly between that L2 and the surrounding L1s and not any of the messages strictly between L1 nodes. The greater danger is in the fact that the compromised L2 node may be potentially involved with assisting in establishing many new session keys.

C. Impact of Pooling Compromised Keys

While the number of keys stored within a single L1 node may not be significantly large, it is essential to understand how the capture and pooling of multiple key rings affect the probability of an attacker establishing secure connections with its neighbors. Consider the scenario which represents a peer-to-peer liberal trust communication scenario in which key match paths are limited to one hop. Figure 10 uses the one-hop liberal trust case to demonstrate the worst-case scenario for undetected node compromises. In this scenario we assume that none of the nodes that are captured share any common keys, meaning that the maximum number of keys has been captured. This provides ceiling values for the damage that can be done should this situation arise.

The balanced case, in which 83 keys are deployed in each L1 node to achieve five-nines connectivity, is the most vulnerable. In this scenario, if an attacker is able to compromise only five nodes and pool their keys, they will have a key match with any neighbor with a probability of 0.971. In an unbalanced case with five L2 nodes, each with the requisite number of keys to ensure five-nines connectivity, if an attacker compromises 10 L1 nodes (25% of the network), it will have a key match with its neighbors with a probability of 0.60, 0.33, 0.096 and 0.02 for L1 key deployments of 30, 20, 10, and 5 keys, respectively. Lastly, if the attacker were to instead expose four unbalanced L1s and one L2, the key match probability would be 0.9207, slightly less than the balanced result.

An additional criticism of pre-distribution schemes is that copies of each key are stored in multiple nodes, therefore making the system less secure. However, with keys distributed in an unbalanced, uniform random fashion over a network of 1,000 nodes (12.5% being L2s), each key is likely to be located in approximately 1% of the nodes on average. An attacker would therefore have to physically compromise almost 100 nodes (12.5% being L2s) in order to locate a specific key. If an attacker is able to compromise nearly 100 nodes in a network without being detected, the system is likely facing far more critical problems. We therefore reassert that this mechanism is appropriate for key management in sensor networks.

These results illustrate that even though the key savings benefit of the unbalanced scheme may seem low in this case (83 vs. 30 keys in an L1 node), the unbalanced system is much more robust in many situations to node compromise, including the pooling of keys from multiple captured nodes.

VI. Conclusion

Through both theoretical analysis and simulations, we have demonstrated the power of an unbalanced key distribution for a variety of network, trust and key management models. The unbalanced scheme not only reduces the number of transmissions necessary to establish session-keys but also reduces the effects of both single and multiple node captures. This approach also allows an administrator to tune the security component to their specific needs. Lastly, the unbalanced scheme allows for even the most memory constrained platforms, from sensor nodes to RFID tags, to hold enough keys to establish secure connections for communication.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, August 2002.

[2] Carlo Blundo, Alfredo De Santis, Amir Herzberg, Shay Kutten, Ugo Vaccaro, and Moti Yung. Perfectly-secure key distribution for dynamic conferences. *Advances in Cryptology (CRYPTO)*, 740:471–486, 1992.

[3] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2003.

[4] Crossbow. Wireless sensor networks. http://www.xbow.com/Products/Wireless_Sensor_Networks.htm.

[5] W. Du, J. Deng, S. Han, and P.K. Varshney. A pairwise key predistribution scheme for wireless sensor networks. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2003.

[6] W. Du, J. Deng, Y.S. Han, S. Chen, and P.K. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *Proceedings of IEEE Infocom*, 2004.

[7] L. Eschenauer and V. Gligor. A key management scheme for distributed sensor networks. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, November 2002.

[8] J. Hubaux, L. Buttyn, and S. Capkun. Security, testbeds and applications: The quest for security in mobile ad hoc networks. In *Proceedings ACM International Symposium on Mobile ad hoc networking & computing (MobiHoc)*, October 2001.

[9] D. Liu and P. Neng. Establishing pairwise keys in distributed sensor networks. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2003.

[10] A. Perrig, R. Canetti, D. Tygar, and D. Song. The tesla broadcast authentication protocol. In *RSA Cryptobytes*, 2002.

[11] A. Perrig, R. Szewczyk, V Wen, D. Culler, and D. Tygar. Spins: Security protocols for sensor networks. In *Proceedings of the International Conference on Mobile Computing and Networks*, 2001.

[12] J. Spencer, editor. *The Strange Logic of Random Graphs*, volume 22. Springer-Verlag, 2002.

[13] A. Stajano and R. Anderson. The ressurecting duckling: Security issues for ad hoc networks. In *Proceedings of the International Workshop on Security Protocols*, 1999.

[14] P. Traynor, R. Kumar, H. Bin Saad, G. Cao, and T. La Porta. LIGER: A Hybrid Key Management Scheme for Heterogeneous Sensor Networks. Technical Report NAS-TR-0008-2005, Network and Security Center Technical Report, Penn State University, 2005. http://www.cse.psu.edu/∼traynor/TR-NAS-0008-2005.pdf.

[15] G. Wang, G. Cao, and T. La Porta. Mobility-assisted sensor deployment. In *Proceedings of IEEE Infocom*, 2004.

[16] F. Ye, H. Luo, S. Lu, and L. Zhang. Statistical en-route detection and filtering of injected false data in sensor networks. In *Proceedings of IEEE Infocom*, 2004.

[17] S. Zhu, S. Setia, and S. Jajodia. LEAP: efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2003.

[18] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2004.

[19] S. Zhu, S. Xu, S. Setia, and S. Jajodia. Establishing pair-wise keys for secure communication in ad hoc networks: A probabilistic approach. In *Proceedings of the IEEE International Conference on Network Protocols (ICNP)*, November 2003.

APPENDIX A

The probability that two key rings will share at least one key is 1 - P[two nodes share no keys]. Unlike the balanced case, where a key ring of size k is removed initially, a key ring of size m, which is stored in the L2 node, is selected first. The number of key rings of size k that do not share at least one key with the key ring of size m is therefore:

$$\frac{(P-m)!}{k!(P-m-k)!}$$

The probability that no key is shared between the two rings is the ratio of the number of rings with no match to the total number of possible rings. The probability of at least one key being shared between two rings is therefore:

$$1 - \frac{k!(P-k)!(P-m)!}{P!k!(P-m-k)!}$$

APPENDIX B

The probability that an L1 node *can not* connect to any L2 node is:

$$(1-U)^g$$

The probability that two nodes *can* each contact at least one L2 node is therefore:

$$(1-(1-U)^g)^2$$

Given that a path across L2s can be established, the probability that two L1 nodes are connected via the limited trust model is therefore:

$$P[Conn] = 1-(1-B)(1-(1-(1-U)^g)^2)$$

which is simply: $1 - \overline{P[Direct]} * \overline{P[Indirect]}$.

APPENDIX C

For the unbalanced network case, the expected number of neighbors that any node can talk to within one hop is:

$$\begin{aligned} E_0 \quad &+ \quad \sum_{k=1}^{g}\binom{g}{k}U^k(1-U)^{g-k}(1-B)^{n-1}E_1 \\ &+ \quad \sum_{k=1}^{g}\binom{g}{k}\sum_{i=1}^{n-1}\binom{n-1}{i}U^k(1-U)^{g-k}B^i(1-B)^{n-1-i}E_2 \\ &+ \quad \sum_{i=1}^{n-1}\binom{n-1}{i}B^i(1-B)^{n-1-i}(1-U)^g E_3 \end{aligned} \tag{11}$$

where,

$$\begin{aligned} E_0 &= B*(n-1), \\ E_1 &= (n-1)(1-(1-U)^k), \\ E_2 &= [(n-1-i)(1-(1-U)^k)] \\ &+ [(n-1-i)(1-U)^k(1-(1-B)^i)], \\ E3 &= (n-1-i)(1-(1-B)^i). \end{aligned}$$

E_x is the expected number of nodes in each case that we analyze, and U is defined in Eq. 2. Now, we explain each term in detail.

1st Term: E_0 is the average number of L1 nodes that an L1 source node can talk to directly. E_0 is determined by multiplying the number of neighboring nodes by B, the probability that two L1s can converse:

$$E_0 = B*(n-1) \tag{12}$$

2nd term: E_1 accounts for the source node having a direct connection with at least one L2 node and no direct connections to other L1 nodes. The expected number of nodes the source can talk to is:

$$\sum_{k=1}^{g}\binom{g}{k}U^k(1-U)^{g-k}(1-B)^{n-1}E_1 \tag{13}$$

$$E_1 = (n-1)(1-(1-U)^k)$$

$T(i)$ is the expected number of additional nodes with which node i may communicate.

$$\begin{aligned} T_{L2}(0) &= (n-1) \\ T_{L2}(1) &= T_{L2}(0) - UT_{L2}(0) = (1-U)T_{L2}(0) \\ &\cdots \\ T_{L2}(k-1) &= T_{L2}(k-2) - UT_{L2}(k-2) \\ &= (1-U)^{k-1}T_{L2}(0) \end{aligned}$$

$$\begin{aligned} E_1 &= UT_{L2}(0) + UT_{L2}(1) + UT_{L2}(2) + \cdots + UT_{L2}(k-1) \\ &= U\sum_{l=0}^{k-1}T_{L2}(l) \\ &= U\sum_{l=0}^{k-1}(1-U)^l T_{L2}(0) \\ &= U(n-1)\sum_{l=0}^{k-1}(1-U)^l \\ &= U(n-1)\frac{(1-U)^k-1}{(1-U)-1} = (n-1)(1-(1-U)^k) \end{aligned}$$

3rd term: E_2 covers nodes that have direct connections to k L2 nodes and i L1 nodes. We represent this as:

$$\sum_{k=1}^{g}\binom{g}{k}\sum_{i=1}^{n-1}\binom{n-1}{i}U^k(1-U)^{g-k}B^i(1-B)^{n-1-i}E_2 \quad (14)$$

$$E_2 = [(n-1-i)(1-(1-U)^k)]+[(n-1-i)(1-U)^k(1-(1-B)^i)]$$

The total expected number of nodes consists of two cases: those reached through L2 nodes (E_{L2}) and those through L1 nodes (E_{L2}). We will derive each case separately and compute the final total expected number of nodes with which the source can talk.

$$E_2 = E_{L2} + E_{L1}$$

Each L2 node then computes the percentage of the remaining unreached L1 nodes with which it can communicate. This process recursively adds percentages for L2 nodes as follows:

$$\begin{aligned}
T_{L2}(0) &= (n-1-i)\\
T_{L2}(1) &= T_{L2}(0) - UT_{L2}(0) = (1-U)T_{L2}(0)\\
T_{L2}(2) &= T_{L2}(1) - UT_{L2}(1) = (1-U)T_{L2}(1)\\
&= (1-U)^2T_{L2}(0)\\
&\cdots\\
T_{L2}(k-1) &= T_{L2}(k-2) - UT_{L2}(k-2)\\
&= (1-U)T_{L2}(k-2) = (1-U)^{k-1}T_{L2}(0)
\end{aligned}$$

The number of L1 nodes we can expect to communicate with through L2 nodes which are directly connected to the source L1 node is then:

$$\begin{aligned}
E_{L2} &= UT_{L2}(0) + UT_{L2}(1) + \cdots + UT_{L2}(k-1)\\
&= U(1-U)^0T_{L2}(0) + U(1-U)^1T_{L2}(0)\\
&+ \cdots + U(1-U)^{k-1}T_{L2}(0)\\
&= UT_{L2}(0)\sum_{l=0}^{k-1}(1-U)^l\\
&= U(n-1-i)\frac{(1-U)^k-1}{(1-U)-1}\\
&= (n-1-i)(1-(1-U)^k)
\end{aligned}$$

After considering the L2 nodes, each L1 adds its contribution to the total percentage by calculating the probability that it can communicate with the remaining unreached L1 nodes. The process recursively adds percentages for L1 nodes as follows:

$$\begin{aligned}
T_{L1}(0) &= (n-1-i) - (n-1-i)(1-(1-U)^k)\\
&= (n-1-i)(1-U)^k\\
T_{L1}(1) &= T_{L1}(0) - BT_{L1}(0) = (1-B)T_{L1}(0)\\
T_{L1}(2) &= T_{L1}(1) - BT_{L1}(1) = (1-B)T_{L1}(1)\\
&= (1-B)^2T_{L1}(0)\\
&\cdots\\
T_{L1}(i-1) &= T_{L1}(i-2) - BT_{L1}(i-2) = (1-B)^{i-1}T_{L1}(0)
\end{aligned}$$

The number of expected nodes through directly connected L1 nodes is then:

$$\begin{aligned}
E_{L1} &= BT_{L1}(0) + BT_{L2}(1) + \cdots + BT_{L2}(i-1)\\
&= B(1-B)^0T_{L1}(0)\\
&+ B(1-B)^1T_{L1}(0) + \cdots + B(1-B)^{i-1}T_{L1}(0)\\
&= BT_{L1}(0)\sum_{l=0}^{i-1}(1-B)^l\\
&= B(n-1-i)(1-U)^k\frac{(1-B)^i-1}{(1-B)-1}\\
&= (n-1-i)(1-U)^k(1-(1-B)^i)
\end{aligned}$$

4th term: In the final term, we examine the case in which the source L1 node does not have any connections to L2 nodes and has i direct connections to L1 nodes. The equation is:

$$\sum_{i=1}^{n-1}\binom{n-1}{i}B^i(1-B)^{n-1-i}(1-U)^gE_3 \quad (15)$$

$$E_3 = (n-1-i)(1-(1-B)^i)$$

$T(i)$ is the expected number of additional nodes that node i may communicate with:

$$\begin{aligned}
T(0) &= n-1-i\\
T(1) &= T(0) - BT(0)\\
&\cdots\\
T(i-1) &= T(i-2) - BT(i-2)
\end{aligned}$$

The total number of expected nodes is then:

$$E_3 = BT(0) + BT(1) + \cdots + BT(i-1)$$

This can be simplified to:

$$\begin{aligned}
E_3 &= B\sum_{k=0}^{i-1}T(k)\\
&= B(n-1-i)\sum_{k=0}^{i-1}(1-B)^k\\
&= B(n-1-i)\frac{(1-B)^i-1}{(1-B)-1}\\
&= (n-1-i)(1-(1-B)^i)
\end{aligned}$$

For the balanced network case, the second and third term can be removed from Eq. 8 because there are no L2 nodes. In the fourth term, $(1-U)^g$ becomes 1 because of the same reason. Therefore, for the balanced network case, we can get the following equation (same as Eq. 6) for the expected number of nodes that any node can talk to within one hop is:

$$E_{L1}^1 = E_0 + \sum_{i=1}^{n-1}\binom{n-1}{i}B^i(1-B)^{n-1-i}E_1 \quad (16)$$

where, $E_0 = B*(n-1)$ and $E_1 = (n-1-i)(1-(1-B)^i)$.

APPENDIX D

Let $p(i)$ be the probability that two nodes share exactly i keys in common. The number of ways in which a key ring of size k and one of size m can be chosen from a pool P are $\binom{P}{k}$ and $\binom{P}{m}$, respectively. There are also $\binom{P}{i}$ ways in which two nodes can chose i keys in common. After i common keys have been selected, there remains $(m-i)+(k-i)$ key rings that must still be constructed from the remaining pool $(P-i)$. The number of ways to distribute these remaining keys between key rings of size k and m is $\binom{(m-i)+(k-i)}{m}$. The probability that two nodes share exactly i keys in common is:

$$p(i) = \frac{\binom{P}{i}\binom{P-i}{(m-i)+(k-i)}\binom{(m-i)+(k-i)}{m-i}}{\binom{P}{m}\binom{P}{k}}$$

The probability that two nodes share at least q keys with each other is therefore:

$$1 - \sum_{i=0}^{q-1}p(i)$$