PIKE: Peer Intermediaries for Key Establishment in Sensor Networks

Haowen Chan
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15221, USA
Email: haowenchan@cmu.edu

Adrian Perrig
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15221, USA
Email: perrig@cmu.edu

***Abstract*— The establishment of shared cryptographic keys between communicating neighbor nodes in sensor networks is a challenging problem due to the unsuitability of asymmetric key cryptography for these resource-constrained platforms. A range of symmetric-key distribution protocols exist, but these protocols do not scale effectively to large sensor networks. For a given level of security, each protocol incurs a linearly increasing overhead in either communication cost per node or memory per node. We describe Peer Intermediaries for Key Establishment (PIKE), a class of key-establishment protocols that involves using one or more sensor nodes as a trusted intermediary to facilitate key establishment. We show that, unlike existing key-establishment protocols, both the communication and memory overheads of PIKE protocols scale sub-linearly ($O(\sqrt{n})$) with the number of nodes in the network yet achieving higher security against node compromise than other protocols.**

I. Introduction

Secret communication is an important requirement in many sensor network applications, so shared secret keys are used between communicating nodes to encrypt data. Since predetermining the potential location or connectivity of sensor nodes in a large deployment can be impractical, we cannot simply preload each sensor node with the relevant shared keys. *Key establishment* protocols are used to set up the shared secrets, but the problem is complicated by the sensor nodes' limited computational capabilities, battery energy, and available memory. Hence, asymmetric cryptography such as RSA or Elliptic Curve cryptography (ECC) is unsuitable for most sensor architectures due to high energy consumption and increased code storage requirements.

Several alternative approaches have been developed to perform key management on resource-constrained sensor networks without involving the use of asymmetric cryptography.

KDC-based schemes rely on the presence of a resource-rich key distribution center (KDC) in the network to act as a trusted arbiter for key establishment.

Examples of such schemes include SPINS [17] and Kerberos [20]. In this class of protocols, the memory resource load on the sensor nodes is low since each node only needs to secure its communications with the KDC. For example, in SPINS, only a single symmetric key shared with the base station (KDC) needs to be stored in each sensor node. However, the protocol is dependent on every pair of communicating sensor nodes being able to communicate with the base station during the key establishment process. In a large multi-hop network, this creates a non-uniform communication pattern where the communication load is focused around the base station. As the nodes closest to the base station are obliged to forward most of the communications between the base station and the rest of the sensor network, they expend battery energy at a higher rate, thus often shortening the lifetime of the network. This focused communication load is proportional to the total number of nodes in the network. Furthermore, the base station can become a rich target for compromise, since it represents a single point of failure which can break the security of the entire network. The highly focused communication pattern also allows an adversary to easily perform traffic analysis to locate the base station for compromise.

Several other schemes use large stores of cryptographic information to ensure that any two nodes can perform key agreement directly without using intermediaries or asymmetric cryptography. The simplest such memory-intensive scheme is the full pairwise scheme [5], [7]. In this scheme, each node in a network of n nodes shares a unique pairwise key with every other node in the network. The memory overhead is $n-1$ cryptographic keys for every sensor node. Blom [2], Blundo et al. [3], and Leighton and Micali [13] propose other schemes which likewise guarantee that any two nodes will be able to perform key establishment, but each of these schemes also involves a $\Omega(n)$ high memory cost if we require that the system is secure against an adversary capable of compromising a certain fraction of the total number of nodes in the network.

Random key predistribution schemes (abbreviated as random-key schemes) are another major class of key-establishment protocols for sensor networks. Examples include schemes proposed by Eschenauer and Gligor [7], Chan et al. [5], Du et al. [6], and Liu and Ning [15]. These schemes take advantage of the fact that a random graph is connected with high probability if the average degree of its nodes is above a threshold. Hence, key establishment only needs to be performed probabilistically such that any two neighboring nodes have some probability p of successfully completing key establishment. The probability p is computed such that the average number of secure links established by each node is greater than the threshold number of secure links needed to achieve a connected network with high probability. Once the connected secure network is formed, *path keys* are then routed through the secure links to complete the process of key establishment between every neighbor. The main advantage of random key schemes is that communication costs per node are constant regardless of the total number of nodes in the network. Random key schemes incur a high memory overhead which increases linearly with the number of nodes in the network if the level of security is kept constant [5], [6]. Furthermore, since random key distribution is probabilistic in nature, it is only suitable for networks where the random graph model for connectivity holds. For example, in a network where nodes are not densely distributed, or in a network where node density is non-uniform, performing probabilistic key establishment could result in a disconnected graph due to the fact that a few critical pairs of nodes could not successfully perform key establishment. Since sparse networks with few redundant links tend to minimize sensor network hardware costs, they represent an important class of sensor network deployments for which random key distribution is unsuitable.

Contributions. Let n represent the number of nodes in the network.

- We present Peer Intermediaries for Key Establishment (PIKE), a key-distribution scheme based on using peer sensor nodes as trusted intermediaries.
- PIKE is designed to address the lack of scalability of existing symmetric-key key distribution schemes. All existing schemes incur linearly increasing costs ($\Omega(n)$) in either communications per node or memory per node. PIKE achieves a trade-off by achieving $O(\sqrt{n})$ overheads in both communications per node and memory per node. This is a highly desirable point in the design space of key-distribution protocols.
- PIKE establishes keys between any two nodes regardless of network topology or node density. This makes it applicable to a wider range of deployment scenarios than random key predistribution, which requires a network deployment with high, uniform node density.
- Besides the KDC approaches that have a high communication overhead, PIKE is more resilient than previous approaches against sensor node compromise.

II. Assumptions

We assume that an initial insecure communication infrastructure exists to perform key establishment. We assume nodes are globally addressable in this infrastructure, that is, any node can send communications through the network to any other node in the network. An example of such a globally addressable communications infrastructure is geographically addressed networking using GPSR as a routing protocol [12] and a geographic hash table (GHT) as a address lookup service [19]. Other examples of globally addressable routing infrastructures include ones proposed by Rao et al. [18] and Newsome and Song [16]. Both these schemes achieve globally addressable, efficient routing without requiring location information. Because the initial communications network is set up prior to security establishment, it is assumed to be insecure.

We assume the strong *node-compromise* attacker model adopted by the previously described key distribution schemes [2], [3], [5], [6], [7], [15]. Specifically, we assume that the attacker is capable of compromising a fraction of the total number of nodes in the network and exposing the secret information contained within them. We will model the effects of two forms of node compromise. In *passive node compromise* we assume that after node compromise, attacker can only launch passive attacks such as eavesdropping. In *active compromise* we assume that the attacker is also able to perform active attacks such as providing false routing metrics through the compromised node.

We assume that the sole goal of the adversary is the exposure of the keys established using the protocol.

III. The PIKE Protocol

In this section, we describe the basic PIKE protocol. We also describe various extensions to the protocol. We analyze and discuss their trade-offs. Throughout this paper, n represents the number of nodes in the network.

A. Motivation

We wish to address the lack of scalability of current symmetric-key key distribution protocols. KDC-based schemes incur $\Omega(n)$ focused communication load on the nodes nearest to the base station. Random Key predistribution schemes [5], [6] and other schemes such as those due to Blom [2], Blundo et al. [3], and Leighton and Micali [13] all incur at least $\Omega(n)$ memory cost in terms of key storage space in order to maintain security if it is assumed that the adversary is capable of compromising a fixed fraction of the total nodes. We wish to design a scheme that will incur sub-linear overheads in *both* focused communication load per node and memory per node, while retaining the property of resilience against the compromise of a fraction of the network.

B. Basic PIKE Protocol

The basic idea in PIKE is to use sensor nodes as trusted intermediaries to establish shared keys between nodes. Each node shares a different (unique) pairwise key with each of $O(\sqrt{n})$ other nodes in the network. The keys are deployed such that for any two nodes A and B, it is possible to find some node C in the network that shares a unique pairwise key with both A and B. A can then securely route the key establishment message through C to B. Since we only predistribute unique pairwise keys that are shared between exactly two nodes, the established key is secure if C has not been compromised by the adversary.

Let the maximum number of node IDs in the network be n. For brevity, we will assume that n is a perfect square; if n is not a perfect square we can always replace $\sqrt{n}$ with $\lceil\sqrt{n}\rceil$ wherever it occurs.

We associate each node with an ID of the form (x, y) where $x, y \in \{0, 1, 2, \ldots, \sqrt{n} - 1\}$. Each node (x, y) is then loaded with a secret key pairwise-shared solely with each node in the two sets of nodes below:

$$(i, y)\ \forall i \in \{0, 1, 2, \ldots, \sqrt{n} - 1\}$$

and

$$(x, j)\ \forall j \in \{0, 1, 2, \ldots, \sqrt{n} - 1\}$$

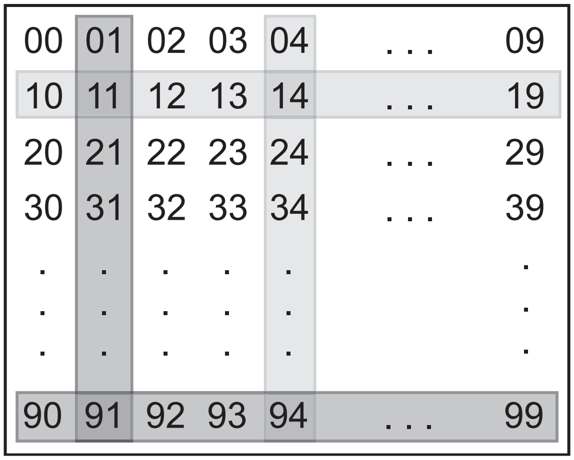


Fig. 1. Sample virtual ID space for 100 nodes. Each number represents a node ID. Dark and light shaded boxes indicate nodes which share unique keys with nodes 91 and 14 respectively. The keys are pairwise-shared and not common to a row; for example 91 and 01 share a key that is distinct from the key shared between 91 and 11. In the above diagram, either node 11 or node 94 can be used to establish keys between nodes 91 and 14. This grid of node IDs is a purely virtual space; there is no correspondence to the actual physical locations of the nodes.

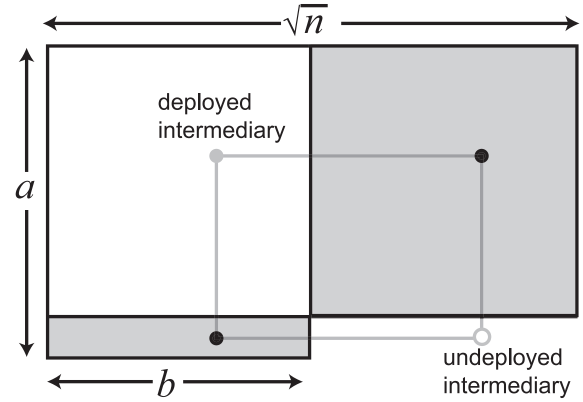


Fig. 2. Deployment order of node IDs. The largest node ID deployed is $(a-1, b-1)$. If $b = \sqrt{n}$ then there are always at least two intermediaries for any two nodes; otherwise there is a small (at most $\frac{1}{4(a-1)}$) probability that only one intermediary has been deployed.

Each key is unique and shared only between two nodes, hence they are called *pairwise* keys. For example, (x, y) will share a key $K_{(x,y),(1,y)}$ with $(1, y)$ and a different key $K_{(x,y),(2,y)}$ with $(2, y)$ and so on. Each node stores $2(\sqrt{n}-1)$ keys and the total number of unique keys generated is $n(\sqrt{n}-1)$.

Due to the way pairwise keys are shared, any two nodes A and B will be able to find two node IDs which will share a pairwise key with both A and B. Specifically, if node A has ID (x_A, y_A) and node B has ID (x_B, y_B), then the nodes with IDs (x_A, y_B) and (x_B, y_A) will share pairwise keys with both A and B.

Figure 1 shows an example of how the scheme works for a network of 100 nodes. The grid of node IDs is represented; there is no correlation with the actual physical locations of the deployed nodes. In this example, $\sqrt{n} = 10$ and hence the node IDs can be depicted as numbers from 0 to 99, where the node ID $d_1 \times 10 + d_2$ corresponds to the general form (d_1, d_2) described previously. The two sets of nodes that share keys with any given node A are the nodes that lie on the same row as A, and the same column as A, respectively. In the example, suppose node 14 wishes to perform key establishment with node 91. They both share unique pairwise keys with node 94 since node 94 lies on the same row as node 91 and the same column as node 14. Likewise, node 11 is a valid intermediary since it lies on the same column as node 91 and the same row as node 14. Note that if two neighboring nodes happen to have IDs that lie on the same row or same column then they already share a unique pairwise key and do not need to perform further key establishment. This square grid deployment provides two possible intermediaries for every pair of nodes. A less redundant version (where only one intermediary exists for every pair of nodes) can be constructed by using only node IDs from above the diagonal of the grid; however as we shall see in Section III-C, having redundant intermediaries results in lower average communication overhead.

Once the IDs of the possible intermediaries have been found, the node initiating the key establishment chooses an intermediary through which to route the key-establishment message. It bases this choice on which of the two intermediaries has the smallest routing metric (or heuristic) for the path that the message will take.

As an example, if geographic routing is used, the heuristic used is geographic distance between the nodes. Hence, if node A and node B wish to perform key establishment through one of the two possible intermediaries C_1 or C_2, they will choose $C = C_m$ such that

$$m = \underset{i \in 1,2}{\text{argmin}}\, d(A, C_i) + d(C_i, B)$$

where $d(F, G)$ denotes the physical distance between nodes F and G. Note that geographic routing is only an example; other routing mechanisms such as distance-vector could be used. In that case, other metrics such as the number of hops to reach the intermediary could be used to decide which intermediary to pick.

Once the intermediary C is chosen, A encrypts the new key to be shared with B using the key it shares with C and then sends it to C. C decrypts the key and re-encrypts it using the key it shares with B, and sends it to B. Finally, B sends a nonce and MAC to A to confirm the receipt of the key.

The process is summarized below. In this notation, K_{AB} refers to the unique pairwise key shared by A and B. N_B denotes a nonce generated by B. $A \rightarrow B$: M refers to a message M sent from A to B. $E_K\{M\}$ refers to a message M encrypted using key K. $\text{MAC}_K(M)$ refers to the message authentication code (MAC) for the message M, under the key K.

$$\begin{aligned}
A \rightarrow C:\ & E_{K_{AC}}\{A, B, K_{AB}\}, \text{MAC}_{K_{AC}}(E_{K_{AC}}\{A, B, K_{AB}\}) \\
C \rightarrow B:\ & E_{K_{BC}}\{A, B, K_{AB}\}, \text{MAC}_{K_{BC}}(E_{K_{BC}}\{A, B, K_{AB}\}) \\
B \rightarrow A:\ & E_{K_{AB}}\{A, B, N_B\}, \text{MAC}_{K_{AB}}(E_{K_{AB}}\{A, B, N_B\})
\end{aligned}$$

C. Estimated Communication Overhead and Security

If we assume that the nodes are uniformly distributed on a square, flat two-dimensional surface of area A, the expected distance between any pair of nodes in the network is $\frac{2}{3}\sqrt{A}$. For a deployment with uniform density, the number of hops required to traverse this distance is $O(\sqrt{n})$ [14]. Hence, we can approximate the hop-distance between any two nodes in the network as $\alpha\sqrt{n}$ where α is some constant that depends on the range of the nodes and the exact shape of the deployment area. Since the two intermediary nodes could lie anywhere in the network, the closer one is approximately $\frac{2}{3}\alpha\sqrt{n}$ hops away from the pair of neighboring nodes wishing to perform key-establishment. The total cost for each key establishment is thus approximately $2 \times \frac{2}{3}\alpha\sqrt{n} = \frac{4}{3}\alpha\sqrt{n}$ hops for the round trip. In a network where each node has an average of d neighbors, we can expect a total communication overhead of $\frac{4}{3}d\alpha\sqrt{n}$ per node.

In an ideally secure scheme where no intermediaries are used (such as one using asymmetric cryptography in a public-key infrastructure), the probability that any link between two nodes is compromised is the probability that either of its endpoint nodes are compromised. If some fraction f of nodes is compromised, then the probability of the link being compromised is $1-$(probability that neither endpoint was compromised) = $1-(1-f)^2$. This can be approximated by $2f$ when f is small.

In PIKE, a link is compromised only if either of its endpoint nodes is compromised, or if the intermediary node is compromised. Hence, if some random fraction f of the total number of nodes in the network is compromised, the fraction of total links compromised will be about $1-(1-f)^3$, which can be approximated by $3f$ if f is small. This compares favorably with the $2f$ for the ideal case.

D. Deployment order of node IDs

Thus far, we have described PIKE by assuming n to be the total number of nodes in the sensor network. However, since the scheme needs to allow for the expansion of the network, in fact n represents the maximum number of node IDs to be generated in the lifetime of the network. The actual number of nodes deployed in the network at any time is typically less than n. In order to maximize the number of intermediaries existing in the network for any two nodes, we stipulate that node IDs be deployed in the network in node ID order. Hence, the deployment order of node IDs would be $(0,0)$, $(0,1)$, $(0,2)$, $\ldots,(0,\sqrt{n})$ followed by $(1,0)$, $(1,1)$, $(1,2)$, $\ldots\,(1,\sqrt{n})$ and so on. Typically, we expect node deployments to be batched, hence the requirement is simply that each deployment batch should be of contiguous node IDs and no node IDs are skipped between batches.

By deploying the nodes in node ID order, we ensure that the space of deployed node IDs is roughly rectangular, hence the number of available intermediaries between any two nodes is usually two and at least one. Figure 2 reflects this reasoning. A pair of nodes shares only one intermediary if one lies in the shaded region on the left and the other lies in the shaded region on the right. The probability of this occurring is:

$$
\begin{aligned}
p &= \text{fraction of left shaded area} \times \text{fraction of right shaded area} \\
&= \frac{b}{m(a-1)+b} \times \frac{(m-b)(a-1)}{m(a-1)+b} \\
&< \frac{b}{ma} \times \frac{(m-b)(a-1)}{ma} \\
&= \frac{b^2(1-a)+bam}{(ma)^2} \\
&\quad \text{Since the above is maximized when } b = \tfrac{ma}{2(a-1)}, \\
&< \frac{1}{4(a-1)}
\end{aligned}
$$

Hence, for example, if $n = 4900$, and 2000 nodes were already deployed, then the probability of encountering only one intermediary is bounded by $\frac{1}{4(\frac{2000}{\sqrt{4900}}-1)} = \frac{1}{108}$, which is less than 1%. Hence, most of the nodes will have two intermediaries to choose from.

The above analysis assumes that there are no nodes that become inactive or unreachable. In Sections III-F, III-G and III-H we will describe methods that will increase the number of potential intermediaries, thus reducing the effects of failed nodes.

E. Memory Overhead

In a naive implementation of PIKE, each node needs a unique pairwise key to be shared with each of the other nodes on the same row and column as itself. Hence, $2(\sqrt{n}-1)$ keys will need to be stored on each node.

By extending an idea first proposed by Shih-I Huang [10], we can reduce the memory storage overhead by a factor of two. Consider a row of node IDs in the grid of node IDs. Each of these nodes needs to share a unique pairwise key with all the other nodes in the row. Since there are $\sqrt{n}$ nodes in the row let us relabel them 0, 1, 2, $\ldots$, $\sqrt{n}$ for clarity. We assign a unique secret K_a to each node $a \in \{0,\ldots,\sqrt{n}\}$. Suppose $\sqrt{n}$ is odd. For each other node $b = (a+i) \mod \sqrt{n}$ where $i \in \{1,\ldots,\frac{\sqrt{n}-1}{2}\}$, we assign the value $h(K_a||h(b))$ as the key for the pair (a,b), where h is a pre-image resistant cryptographic hash function. For each other node $c = (a-i) \mod \sqrt{n}$ where $i \in \{1,\ldots,\frac{\sqrt{n}-1}{2}\}$, we symmetrically assign the value $h(K_c||h(a))$ as the key for the pair (a,c). In this scheme, node a can use its secret K_a to dynamically generate the pairwise key it needs with node b if $b = (a+i) \mod \sqrt{n}$ where $i \in \{1,\ldots,\frac{\sqrt{n}-1}{2}\}$. Only the pairwise keys with the other $\frac{\sqrt{n}-1}{2}$ nodes need to be stored, thus reducing the memory overhead by a factor of two.

Figure 3 illustrates the manner in which the keys are generated for an example where $\sqrt{n} = 5$. In the diagram, each node can generate two pairwise keys using its secret K_i, while the other two pairwise keys are stored. The overall memory overhead is $\frac{\sqrt{n}-1}{2} + 1$ keys. We described the key derivation method for the case where $\sqrt{n}$ is odd. The method extends

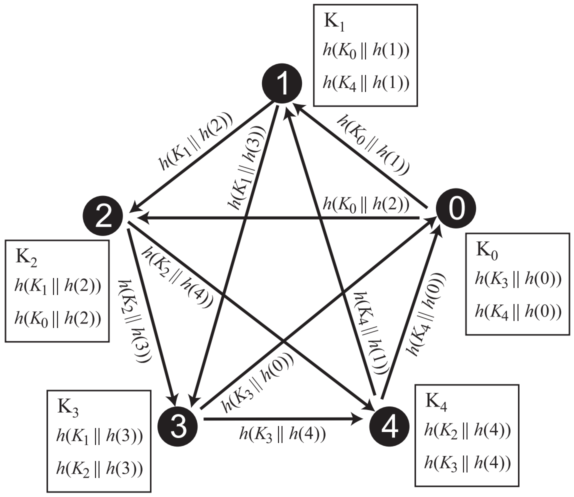


Fig. 3. Pairwise key derivation for 5 nodes. Nodes are indicated as black circles. Each edge represents a unique pairwise key between two nodes, and the derivation of the key is indicated alongside the edge. Boxes next to the nodes denote the keys that must be stored on that node. For $2k+1$ nodes, only $k+1$ keys need to be stored.

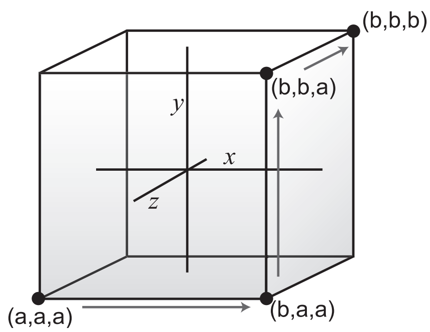


Fig. 4. Extension to three dimensions. Each axis now contains $\sqrt[3]{n}$ nodes, and each node now shares keys with $3(\sqrt[3]{n}-1)$ other nodes, corresponding to the x, y and z axes. Shown on the diagram is how a node with ID (a,a,a) can establish a key with node (b,b,b) using (b,a,a) and (b,b,a) as intermediaries.

directly to the case where $\sqrt{n}$ is even, yielding a memory overhead of $\frac{\sqrt{n}}{2}+1$ keys.

The method as described performs key determination for a given row. Determination of the keys for a column proceeds identically. The secret K_i can be re-used without loss of security, yielding a total overhead of at most $\sqrt{n}+1$ keys.

F. Extension to three or more dimensions

The basic PIKE scheme as described uses a two-dimensional grid of node IDs to perform key establishment using at most one intermediary. This basic idea can be extended directly into three or more dimensions.

Figure 4 illustrates the lay out of a cube of node IDs which allows PIKE to perform key establishment using at most two intermediaries. Specifically, for three dimensions, each node ID consists of a triple (a_1,a_2,a_3). Each node shares keys with each other node that lies on the same axis as itself, i.e. (i,a_2,a_3), (a_1,i,a_3), and (a_1,a_2,i) for $i\in\{0,\ldots,\sqrt[3]{n}\}$. Every pair of nodes needs to route their key establishment

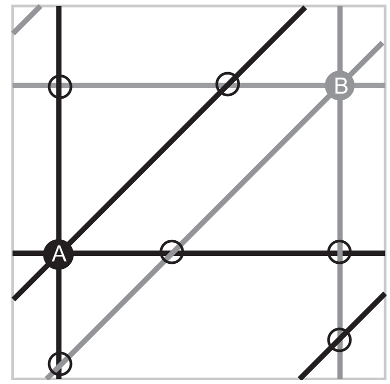


Fig. 5. Adding an extra xy axis. Dark lines are the nodes that share keys with node A, light lines are the nodes that share keys with node B. The 6 nodes that share keys with both A and B are highlighted with circles.

through at most two intermediaries, or sometimes just one intermediary if both nodes lie on the same plane. Node deployment likewise occurs in node ID order, ensuring that a roughly cuboid area is enclosed by the space of the deployed nodes. The selection of the two intermediaries for each link could be done completely at the initiating node, or in a greedy fashion with the closest node being chosen at each stage until the key-establishment packet reaches the destination.

The tradeoff induced by raising the dimensionality of the node ID space is a reduction of the memory overhead from $O(\sqrt{n})$ to $O(\sqrt[3]{n})$, while incurring an additional communication and security cost by requiring two intermediaries per key establishment instead of one. While it is straightforward to further extend the scheme to four or more dimensions, we found that for the network sizes we are considering (≤ 10000 nodes), raising the dimensionality to four or more dimensions does not offer a good trade-off.

G. Inclusion of additional axes of node IDs with shared keys

In the basic PIKE scheme, the closest of the two available intermediary nodes is chosen to assist in the routing of the key establishment message, thus resulting in an expected communication distance of around $\frac{2}{3}\alpha\sqrt{n}$ hops (see Section III-C for derivation). By increasing the number of available intermediaries to choose from to k, we can further reduce this expected distance to $\frac{2}{k+1}\alpha\sqrt{n}$. One way of doing this is to add an additional axis of nodes which share keys with each other.

Figure 5 illustrates the effect of adding an additional axis of nodes which share pairwise keys with each other. The new xy axis runs diagonal to both of the original axes, wrapping around the edge of the grid of node IDs. Specifically, for a node with ID (x,y), the new axis contains the nodes:

$$((x+i) \mod \sqrt{n}, (y+i) \mod \sqrt{n})\ \forall i\in\{0,1,2,\ldots,\sqrt{n}\}$$

Since the grid of node IDs is square, the new xy axis contains exactly the same number of nodes as the original x and y axes, i.e., $\sqrt{n}$ nodes.

To traverse the node ID space from one node to another, there are now three choices of axes to traverse first, followed by another choice of two axes to traverse next, yielding 6 possible intermediary nodes. In general, for q non-parallel axes on a two-dimensional plane, we will get $q(q-1)$ potential intermediary IDs. The effect is reduced by the fact that the entire ID space may not be deployed. For example, in Figure 5, four of the intermediaries lie on the rectangle bounded by A and B and so have probably been deployed, but the lower two intermediaries may not yet have been deployed.

Additional axes can also be added to the scheme with a three dimensional node ID space. In such a case, three additional xy, yz and xz axes could be added. As long as the first intermediary is chosen such that it lies on the same x, y, or z axis as the target, there will exist one useful added axis on the remaining plane that needs to be traversed.

In a 2D node ID space, having a total q axes reduces the expected communication overhead down to a minimum of $\frac{2}{q(q-1)+1}\alpha\sqrt{n}$ (depending on the extent of deployment of the node ID space) while increasing the memory overhead to $\frac{q\sqrt{n}}{2}+1$. Another effect of having additional axes is that it makes the scheme less vulnerable to dead or disconnected nodes. The addition of more axes also has the undesirable side-effect of making the scheme more vulnerable to active attacks. This effect will be explained in Section IV.

H. *Using newly established keys*

In the course of the protocol, new keys will be established between neighboring nodes. These keys could be used in an identical fashion to the original pairwise keys in order to facilitate the establishment of other keys. For example, suppose the node (a_1, a_2) has established a key with neighbor (b_1, b_2). Now (a_1, a_2) wishes to establish a key with another neighbor (b_1, b_3). Since (b_1, b_2) and (b_1, b_3) must share a key since they lie on the same row, (a_1, a_2) can use (b_1, b_2) as an intermediary to perform key establishment with (b_1, b_3). All communications are confined to the locality of (a_1, a_2) and thus the savings in communication overhead can be significant.

The undesirable effect of using the newly established keys is that security can be weakened significantly. A first generation key is dependent on the security of one intermediary. A second generation key could be established using up to two first generation keys, thus being effectively dependent on up to three intermediaries for its security. A third generation key could be dependent on the security of up to seven intermediaries, and so on. Some limit on the number of dependencies of any edge should be used in order to prevent the security of the later links from falling exponentially. Figure 6 illustrates an example where newly established keys are only used if they are dependent on the security of only one intermediary. Since the scheme used here is two-hop key establishment (i.e., a three-dimensional ID space), the maximum number of intermediaries upon which an edge could be dependent is limited to 5 (one extra for each of three edges + two for the actual intermediaries).

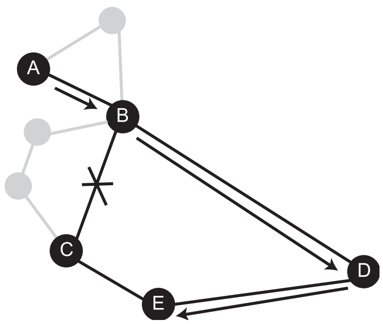


Fig. 6. Using established keys with a limit of one intermediary. A is establishing a key with E. The edge BC is not used since it involves two intermediaries. The final key establishment path $ABDE$ is indicated by arrows. The new edge AE is now dependent on the security of three intermediaries instead of the usual two.

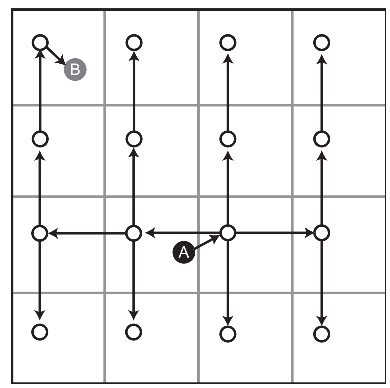


Fig. 7. GHT lookup with replication factor 4^2. Arrows indicate the pattern of dissemination of the location information of A.

I. *Implementation Details*

PIKE is dependent on having a globally addressable communication infrastructure in place prior to the establishment of security. For this paper, we chose to implement the communication infrastructure as GPSR [12] using a modified GHT [19] as an address lookup service. We chose geographic routing as an example of an initial communications infrastructure because GPSR and GHT are easy to implement in simulation. Also, geographic distance can be computed easily as an estimate of the routing cost to any potential intermediary, and the locations of arbitrary nodes can be discovered efficiently via a geographic hash table (GHT). In general, however, any globally addressable communications infrastructure could be used in place of geographic routing. We note that the overall communication overhead for our scheme operating over other communications infrastructures will likely be different from the figures presented in this paper.

A complete description of GPSR and GHT is beyond the scope of this paper. A brief sketch of the mechanics as applied to our scheme is as follows. When a node A is deployed, it maps its ID to a random geographical location h_{a0}. The location h_{a0} is associated with replication points $h_{a1}, h_{a2}, \ldots, h_{ak}$ where k is the data replication factor. A finds the replication point h_{ac} closest to its location, and publishes

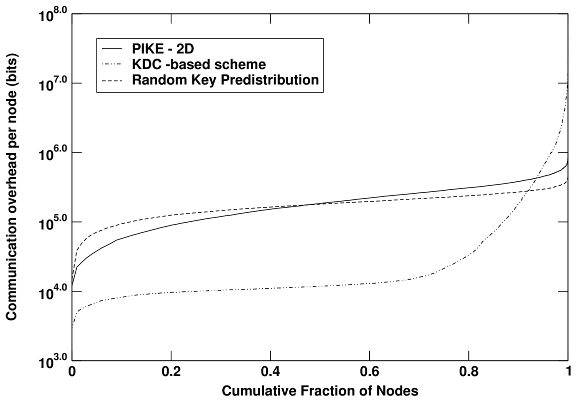


Fig. 8. Histograms of communication overhead per node, 5000 node network with 50 neighbors per unit disc communication radius. PIKE-2D: The basic PIKE scheme

A's physical location to the node closest to h_{ac}. The data is then disseminated to the other replication points. Figure 7 shows how the location data of A is disseminated.

Because we assume that sensor nodes are immobile, we modified GHT slightly to provide a more efficient structure for the dissemination of location information. We removed the hierarchical structure of GHT in favor of a uniform dissemination network since the data in this case has only one source and is unchanging, hence there are no data consistency issues.

When a node B is considering A as a possible intermediary in key establishment, it maps A's ID to find h_{a0}, and calculates the closest replication point. It then queries the closest replication point to discover the location information of A. Once A's location is known by B, it is stored for future reference. The communication overhead involved in location lookups is thus highly efficient. In our simulations, communications due to lookups constituted less than 25% of the total communication overhead.

IV. Evaluation Metrics

In evaluating the PIKE protocols, we will focus on three metrics: focused communication load, memory overhead, and resilience against node compromise. We describe each criteria in turn.

A. Focused Communication Load

Rather than measure the total communication overhead throughout the network, we choose to measure the average communication of the top 10% most heavily loaded nodes in the network. This is because the distribution of communication load caused by the key-establishment schemes we are measuring are typically very heavily skewed. For example, Figure 8 shows the distribution of communications for three schemes in a sample deployment of 5000 nodes. The y-axis is a log scale. As expected, the KDC-based scheme shows an extremely focused load on nodes that are close to the base station, with the most heavily loaded nodes having to communicate up to several hundred times more than the other nodes. PIKE and Random Key Predistribution both incur more evenly distributed load patterns.

In evaluating communication overhead, we are motivated by the energy consumption of the schemes and the resultant effect on node lifetimes. Under focused load, the most heavily loaded nodes will be exhausted first. When a significant fraction of the most heavily loaded nodes have been exhausted, the network could become disconnected, or could suffer from a cascade effect further focusing the communication load on the remaining nodes, resulting in the rapid failure of the remainder of the network. Hence, in evaluating the communication overhead of a scheme, a measure of how much load is focused on the most heavily loaded nodes is more descriptive than the total measure of communications throughout the network. For this reason, we will measure the average communication load of the 10% of the nodes which are the most heavily loaded. This fraction (10%) was chosen arbitrarily, however its exact value should not greatly affect the comparative results of the simulations.

B. Memory overhead

We measure the amount of additional memory that is required for facilitating the key-establishment scheme. This measure refers to the amount of memory that must be permanently dedicated to the key-establishment scheme even while the protocol is not taking place, i.e., it does not count the temporary storage that is needed during the execution of the protocol. For example, in the PIKE schemes, the memory overhead would correspond to the number of bytes needed to store the collections of pairwise keys on each node. For the random key predistribution schemes, it would likewise refer to the total amount of memory needed for the storage of the large collections of keys or cryptographic information on each node.

C. Resilience Against Node Compromise

In this metric we measure the fraction of all communications in the network that is exposed through various levels of node compromise. The fraction of communications compromised is defined as:

$$\frac{\text{total number of compromised links}}{\text{total number of links}}$$

In the above, a "link" refers to a secure communication link between two neighboring nodes. Clearly, any link will be compromised if either one of the principals is compromised. Hence, even an ideal security scheme will expose links at roughly twice the rate at which nodes are compromised. This baseline rate of compromise will be indicated on the graphs.

V. Simulation Details

In this section we will measure the communication overhead, memory overhead, and resilience against node compromise of two configurations of PIKE compared against a generic KDC-based scheme and the random key predistribution scheme described by Du et al. [6].

The networks were simulated on a flat, square deployment field. A unit-disc bidirectional communication model was assumed. Indicated on each set of results will be the total number of nodes in the network as well as the density of deployment, measured in nodes per unit disc. We assume communications are limited to 256 bit packets, including a packet overhead of 32 bits. Longer communications have to be broken up into smaller packets each with its own communication overhead. Node IDs are assumed to be 16 bits in length, keys and message authentication codes (MACs) are assumed to be 80 bits in length, and geographical information for geographical routing is assumed to be 20 bits in length.

The KDC-based Scheme

The KDC-based scheme was modified from SPINS [17] to induce the minimum possible amount of communication overhead under our assumptions. We use GPSR to perform communications through the network. The base station is positioned at the center of the deployment area and this position is known to all nodes. At the beginning of the protocol, each node securely reports its position to the base station. Once the base station has the position of every node, it computes the set of neighbors for each node using the unit disc model and generates a unique pairwise key for each pair of nodes known to be in communication radius. Each key is then sent securely with a MAC to each of the relevant nodes.

The Random Key Predistribution Scheme

We simulated the path-key establishment phase of the random key predistribution scheme described by Du et al. [6]. The probability of two neighbors forming a secure link was calculated using the formula:

$$p_{connect} = \frac{\left(\frac{n-1}{n}\right)(\ln(n) - \ln(-\ln(0.9999)))}{d} \qquad (1)$$

In the above, 0.9999 reflects the probability of the graph being connected, and d is the density of the network in terms of the number of neighboring nodes per unit disc.

In the Du et al. scheme, once the adversary compromises more than $x = \frac{m\omega}{\tau^2}$ nodes (where m is the number of keys stored in each node, and ω and τ are parameters decided by $p_{connect}$), the information leaked by the scheme rises exponentially with the number of nodes compromised. We picked 5% of the total number of nodes in the network as the critical point past which node compromise will begin to break the security of the random key scheme. Thus, we can assume $0.005n = \frac{m\omega}{\tau^2}$ and derive the minimum number of keys required m from the total network size n.

After the initial key establishment is performed, path-discovery must then be performed to discover the paths that path-keys must take from one neighbor to another. Since most neighbors can be reached within three secure hops [7], we implemented path-discovery as a distance-vector protocol with metrics limited to three hops or less. Specifically, each node first broadcasts its list of neighbors with which it shares a secret key, advertising a distance of one hop. Subsequently, each node advertises any improved route that it had learned in the previous round that is under three hops. Since metrics are limited to three hops or less, only two iterations are needed to complete route discovery. All communications are performed on secure links, hence all broadcasts are node-to-node and are encrypted and also authenticated with message authentication codes (MACs). After path-discovery is performed, path key-establishment is completed by routing key-establishment messages along the discovered paths.

The PIKE Schemes

We evaluated two implementations of PIKE. In the first, labeled in the graphs as "PIKE - 2D", we implemented the basic two-dimensional node ID scheme as described in Section III-B. In the second, labeled in the graphs as "PIKE - 3D", we implemented the scheme with a three-dimensional ID space (hence key establishment takes at most three hops of communication) as described in Section III-F. The 3D scheme is designed to incur a lower communications and memory overhead by sacrificing some security against active attacks. In this scheme, we selected intermediaries in a greedy fashion (i.e., the initiating node allows the first intermediary to choose the second intermediary). We also implemented the extension where we added additional axes of nodes which share pairwise keys. The extra axes were added as the xy, yz and xz axes as described in Section III-G. Finally we also allowed the 3D scheme to use newly established secure links to perform key establishment, as described in Section III-H. We only considered using newly established secure links which were dependent on the security of one or less intermediaries. The underlying communication infrastructure was GPSR with GHT as a lookup service. The degree of GHT replication was fixed at 7^2.

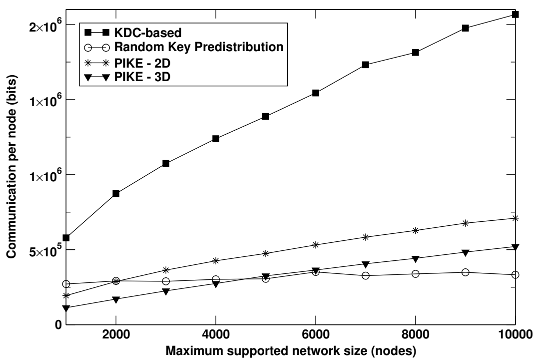
VI. Evaluation Results

For brevity, we shall refer to the PIKE scheme using the two-dimensional node ID space as the "PIKE-2D scheme" and the PIKE scheme using the three-dimensional node ID space as the "PIKE-3D scheme".

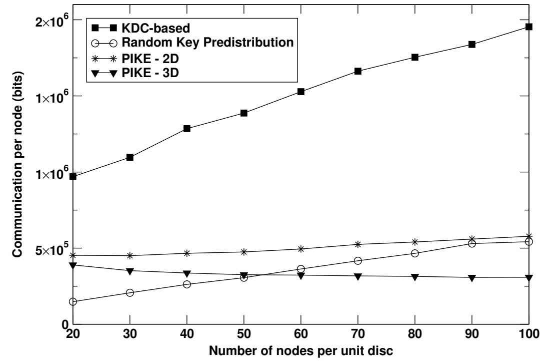
A. Communication Overhead

Figure 9 shows the average communications per node of the top 10% most communicating nodes of the various schemes in various deployments. "Network size" refers to the number of nodes in the network. "Network density" refers to the average number of nodes per unit disc communication area (i.e., the expected number of neighbors per node). The figures shown are the sum totals of bits sent and received.

Figure 9(a) shows communication overhead varying with network size. The PIKE schemes show a sub-linear trend in the communications overhead as the network size increases. This is as expected since the communication load is relatively well distributed over the network, and the expected communication overhead of any node is proportional to $\sqrt{n}$. The KDC-based scheme, on the other hand, shows a rapid increase in focused communication overhead as the network size rises. The amount of communication overhead incurred



(a) Communication vs network sizes, density = 50



(b) Communication vs network density, network size = 5000 nodes

Fig. 9. Average communications overhead of top 10% most heavily loaded nodes

by the random key predistribution scheme is mainly due to the broadcasts in the path-discovery phase. Since the distance vector is limited to three hops or less in all cases, this overhead remains relatively constant regardless of the number of nodes in the network.

Figure 9(b) shows communication overhead varying with network density. Naturally, as network density increases, the number of communicating neighbors increases and hence the amount of key establishment traffic needed also increases. This increase is reflected in the steady rise in communication overhead for the KDC-based scheme. For the PIKE schemes, the effect is counterbalanced by the fact that increasing network density while keeping the total number of nodes in the network constant is the same as decreasing the scale of the deployment area while keeping communication ranges constant. Hence it now takes fewer hops to traverse the same physical distance. This effect balances the need for increased key establishment traffic and results in communication overheads that only vary very slightly with network density. For the random key schemes, increasing the network density increases the number of broadcast messages needed to perform path-discovery to the distance of three hops. Hence, the communication overhead of random keying rises steadily with increasing network density, surpassing the overhead of PIKE-2D at high densities.

In summary, Figure 9 shows that the PIKE schemes incur low communication overheads (comparable with that of random key predistribution for moderate network sizes) and exhibit a sub-linear trend as the network size increases.

B. Memory Overhead

Figure 10 reflects the memory overheads for various parameters of sensor network deployment.

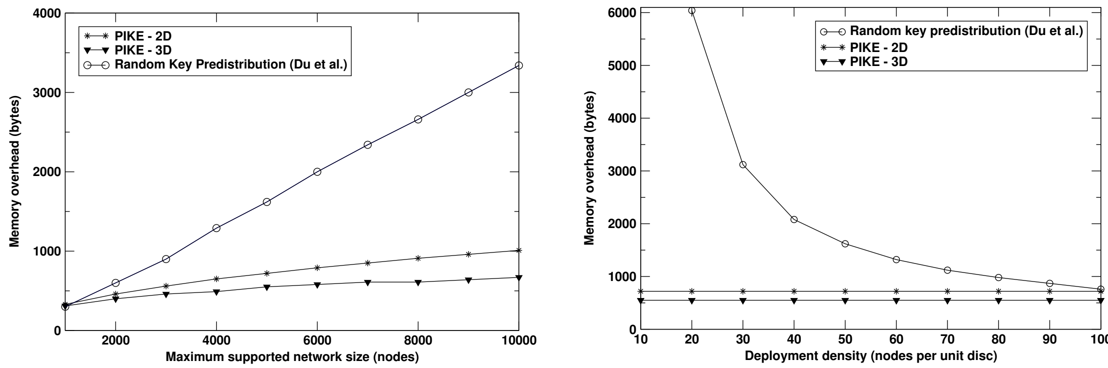
The memory overhead of the random key predistribution scheme was derived from the characteristics of the Du et al. [6] scheme as described in Section V, i.e. supporting node compromise levels of up to 5% of the total nodes. As can be seen, the memory overhead is directly proportional to the total number of nodes in the network.

For the PIKE-2D scheme, each node shares keys with $2(\lceil\sqrt{n}\rceil - 1)$ nodes. Under the key derivation scheme described in Section III-E, we require $\lceil\sqrt{n}\rceil + 1$ keys per node.

For the PIKE-3D scheme, each node shares keys with $6(\lceil\sqrt[3]{n}\rceil - 1)$ nodes. Hence, we require a total of $3\lceil\sqrt[3]{n}\rceil + 1$ keys per node.

Figure 10(a) reflects how the memory consumption of each scheme varies with network size. The memory consumption for the PIKE-2D and PIKE-3D schemes rise with the square root and cube root respectively of the total number of nodes in the network, while the memory consumption for the random key scheme rises linearly with the total number of nodes in the network. The PIKE schemes have a lower memory overhead than the random key scheme, and the difference becomes more pronounced as the number of nodes in the network increases.

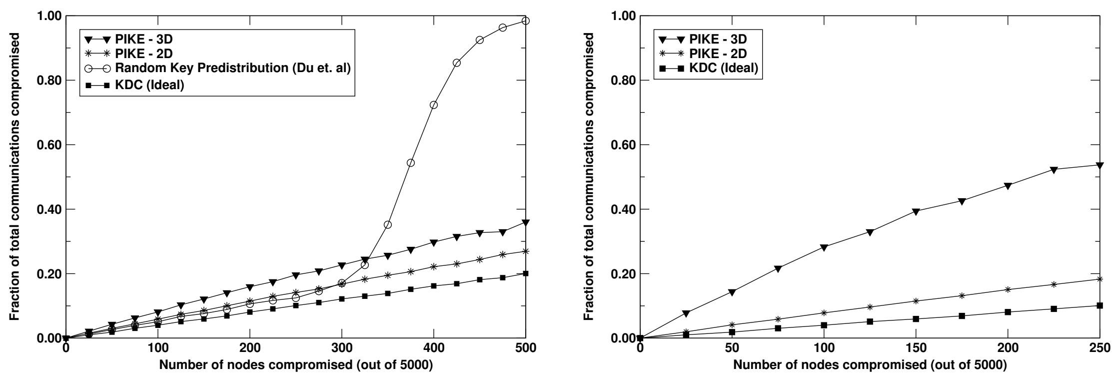
Figure 10(a) reflects how the memory consumption of each scheme varies with network density. The memory overhead of the PIKE schemes are not sensitive to the density of deployment. For the random key scheme, the amount of key storage needed can be reduced as density increases, since $p_{connect}$ as derived in Equation 1 is inversely proportional to the density of deployment. It can be seen that with very dense deployments, the memory needed for random keying can approach the amount needed for the PIKE schemes. However this is at the cost of incurring greater communication overheads as indicated on Figure 9(b). It is also worth noting that random key schemes are unable to support low node densities at any region in the network (e.g., less than 20 nodes) since they cannot guarantee the connectivity of the network formed by the initial secure links if the degree of each node is too low. Unlike the random key schemes, the PIKE schemes can support arbitrarily low node densities or non-uniform node densities.



(a) Memory vs network sizes, density = 50

(b) Memory vs network density, network size = 5000 nodes

Fig. 10. Memory overhead



(a) Passive attacks only

(b) Active attacks

Fig. 11. Security statistics for 5000 nodes, 50 nodes per unit disc. X-axes are different for (a) and (b).

C. Resilience Against Node Compromise

Figure 11 shows the security of the various schemes under simulation. We investigated the effect of node compromise in a network of 5000 nodes with a density of 50 nodes per unit disc communication area.

Figure 11(a) reflects the security of each scheme under passive node compromise. Under this attacker model, the attacker launches only passive eavesdropping attacks after node compromise. The baseline level of communication compromise is indicated by the line for the KDC-based scheme, where the adversary can only compromise a link if they control one of the principals. The baseline (ideal) fraction of communication compromise is approximately $2f$ where f is the fraction of the total number of nodes that has been compromised. We note that the KDC-based scheme only reaches ideal security if the KDC itself has not been compromised. If the KDC is compromised, then all security is lost. In the graphs (Figure 11(a)), it can be seen that both the PIKE-2D scheme and the PIKE-3D scheme exhibit the expected behavior of exposing links at a roughly linear rate similar to the ideal line. Because the PIKE-2D scheme uses one intermediary, the rate of communication exposure is approximately $3f$ where f is the fraction of nodes compromised. Similarly, since the PIKE-3D scheme uses two intermediaries, the rate of communication exposure is approximately $4f$. When less than 300 of 5000 nodes have been compromised, the random key predistribution scheme shows security slightly better than that of PIKE-2D. Once more than 300 nodes have been compromised, the amount of communications in the network that is exposed rises exponentially until the network is almost completely compromised when 500 nodes have been compromised. This is a characteristic of both the Du et al. [6] and the Liu and Ning [15] schemes. The point at which security begins to break is proportional the memory overhead for the scheme, but the inflection point remains a characteristic of the graph regardless of the amount of memory spent on the scheme. In comparison, the PIKE schemes maintain their resilience against node compromise throughout the range of nodes compromised. This kind of steady degradation of security makes it impossible for an adversary to compromise a large fraction of the network by compromising a small fraction of sensor nodes.

Figure 11(b) shows the security of the PIKE schemes under active compromise. For the KDC-based scheme, if we assume that the base station is secure against compromise, links can only be compromised if one of the principals is compromised, so the baseline of ideal security remains the same. For brevity, we did not investigate the possible effects of active attacks against the path-discovery phase of random key predistribution since many well-known routing attacks such as wormholes or black holes are directly applicable, and there are a variety of counter-measures of varying effectiveness [11].

The PIKE-3D scheme exhibits increased vulnerability if the adversary is active and present at the time of deployment. Since the initial communications infrastructure is insecure, the adversary could potentially spoof the distance metric in order to make its compromised nodes more likely to become chosen as intermediaries. For example, suppose A was choosing between nodes C_1 and C_2 as intermediaries to node B, and the adversary has compromised C_1 but not C_2. The adversary could respond to A's query about the location of C_1 with false information, causing it to appear as if C_1 is in the same locality as A when in fact it is distant. Hence, C_1 will be chosen as an intermediary for this key establishment and the adversary will gain control over the link AB. We model the maximum possible effect of this kind of metric spoofing by assuming that whenever a compromised node is a potential intermediary, it will always be chosen in favor of any uncompromised node. We would expect this form of attack to result in a linear increase in the rate of communication compromise when the number of nodes compromised is a small fraction of the total number of nodes. This is because for a link to be secure, now all the potential intermediaries for the link must be secure as well as the two end-points. Hence, for the 2D scheme, the rate of communication compromise would be approximately $4f$ since both the two end-points as well as the two nodes which are potential intermediaries must be secure. For the PIKE-3D scheme, the rate of communication compromise would be much higher since two intermediaries are used for each link, and furthermore there are more potential intermediaries due to the additional axes. Figure 11(b) reflects this trend. The PIKE-3D scheme has weak security under this attacker model, revealing 50% of the communications in the network after only 5% of the nodes have been compromised. However, the PIKE-2D scheme retains a good level of security under active attack. We note that the metric spoofing attack described above may not be feasible in all situations. For example, location spoofing could be detectable by an intrusion detection system that can detect if nodes are claiming to be in more than one place. Also, if the routing infrastructure is secured prior to performing key establishment, such an attack would be prevented.

D. Comparison of Trade-offs

Figure 12 shows the energy and memory trade-offs for various key establishment schemes. The estimated energy cost

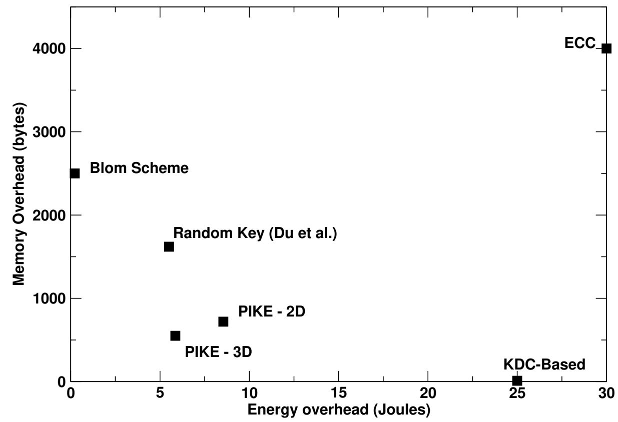


Fig. 12. Estimated energy and memory cost of various key establishment schemes for network size = 5000, density = 50 nodes per unit disc

of elliptic-curve based asymmetric cryptography (ECC) was based on a study performed by Hodjat and Verbauwhede [9] which measured the overhead of ECC-based Diffie-Hellman key agreement on a WINS sensor network. The amount of memory needed for ECC was based on the lesser of the figures reported in the implementation studies by Hasegawa al. [8] and Aydos et al [1]. Asymmetric cryptography is the most scalable and flexible of all the schemes as the overheads and implementation details do not change with network size or density. However, it also presents the largest amount of energy and memory overhead of all the schemes. Figures are also noted for Blom's scheme [2] as an example of a memory-intensive scheme (other schemes such as Leighton and Micali's scheme [13] incur roughly the same memory overheads). The figures are for parameters of the scheme set to provide perfect security until more than 250 nodes are compromised, after which security breaks down completely. Blom's scheme provides for a minimum use of energy resources at the cost of high memory overhead. For the the remaining four schemes which we investigated in simulation (the KDC-based scheme, the random key predistribution scheme and the deterministic schemes), energy consumption was estimated by multiplying the total amount of communications by an average communications cost of 18 μJ/bit [4]. This form of estimation is inaccurate since many hardware implementation factors can affect the consumption of power of communications. However it can give us a rough intuition of how communication energy trades off against computation energy. Code overhead for these schemes are not considered since the basic communication and symmetric-key cryptography routines used are a fixed requirement for all the schemes we are considering, including the ones using asymmetric-key cryptography (this is because asymmetric-key cryptography is too computationally costly for use in all network communications). It can be seen that the KDC-based scheme offers the least amount of memory overhead in return for a high memory cost. The PIKE schemes offer further attractive trade-offs in memory and energy overhead compared with the trade-offs offered by other schemes.

The PIKE schemes also offer further qualitative advantages over the other schemes. Because trust is decentralized in the deterministic schemes, there does not exist a single point of failure in the system where a successful compromise would breach the security of the entire network. This gives PIKE schemes an advantage over KDC-based schemes where the KDC must be made tamper-proof. Also, memory-intensive schemes like the ones proposed by Blom [2], Blundo et al. [3], Leighton and Micali [13] and their associated random-key hybrids described by Du et al. [6] and Liu and Ning [15] offer good security until a certain threshold of nodes have been compromised, after which the security of the network breaks down completely. This makes their parameters extremely difficult to pick since it is hard to decide in advance what fraction of a network an unknown future adversary is capable of compromising. In comparison, the PIKE schemes do not require such sensitive parameters and offer an advantage in that reasonable security still exists in the network even after significant amounts of node compromise, since communications are exposed gradually as nodes are compromised.

VII. Conclusion

PIKE (Peer Intermediaries for Key Establishment) is a class of schemes that uses other sensor nodes in the network as trusted intermediaries to perform key establishment between neighboring nodes. We describe several extensions and parameters to allow the scheme to achieve various trade-offs between communications, memory and security.

Of the two configurations of the PIKE scheme that we investigated, the PIKE-2D scheme offers higher resilience against node capture, in particular against active attacks where the routing metric to sensor nodes are spoofed by the adversary. The PIKE-3D scheme is less resilient against active attacks, but achieves a lower communication and memory overhead. Both schemes exhibit gradual exposure of the communications of the sensor network as nodes are compromised, which is a qualitative advantage over the sudden-breakdown exposure pattern of some memory intensive protocols and their related random key predistribution schemes.

We measured the performance of two configurations of the PIKE schemes in simulation with various sensor network deployment parameters. We show that the PIKE schemes involve lower memory storage requirements than random key distribution while requiring comparable communication overheads. PIKE is currently the only symmetric-key predistribution scheme which scales sub-linearly in both communication overhead per node and memory overhead per node while being resilient to an adversary capable of undetected node compromise.

PIKE enjoys a uniform communication pattern for key establishment, which is hard to disturb for an attacker. The distributed nature of PIKE also does not provide a single point of failure to attack, providing resilience against targetted attacks.

In contrast to the currently popular random-key predistribution mechanisms, PIKE has the advantage that key establishment is *not* probabilistic, so any two nodes are guaranteed to be able to establish a key.

References

[1] M. Aydos, T. Yanik, and C. Koc. An high-speed ECC-based wireless authentication protocol on an ARM microprocessor. In *The 16th Annual Computer Security Applications Conference*, pages 401–410, 2000.

[2] R. Blom. Non-public key distribution. In *Advances in Cryptology: Proceedings of Crypto '82*, pages 231–236, 1982.

[3] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly-secure key distribution for dynamic conferences. In Ernest F. Brickell, editor, *Advances in Cryptology - Crypto '92*, pages 471–486, Berlin, 1992. Springer-Verlag. Lecture Notes in Computer Science Volume 740.

[4] D. Carman, B. Matt, P. Kruus, D. Balenson, and D. Branstad. Key management in distributed sensor networking. In *DARPA Sensor IT Workshop*, 2000.

[5] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy*, May 2003.

[6] W. Du, J. Deng, Y. Han, and P. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proceedings of the Tenth ACM Conference on Computer and Communications Security (CCS 2003)*, pages 42–51, October 2003.

[7] L. Eschenauer and V. 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.

[8] T. Hasegawa, J. Nakajima, and M. Matsui. A practical implementation of elliptic curve cryptosystems over gf(p) on a 16bit microcomputer. In *Proceedings of the First Internation Workshop on Practice and Theory in Public Key Crpytography*, pages 182–194, 1998.

[9] A. Hodjat and I. Verbauwhede. The energy cost of secrets in ad-hoc networks. In *Proceedings of the IEEE Circuits and Systems Workshop on Wireless Communications and Networking*, 2002.

[10] S.I. Huang. Adaptive random key distribution schemes for wireless sensor networks. In *Proceedings of the International Workshop on Advanced Developments in Software and Systems Security*, 2003.

[11] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. In *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications*, pages 113–127, 2003.

[12] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Proceedings of the Sixth International Conference on Mobile Computing and Networking (MobiCom 2000)*, pages 243–254, August 2000.

[13] T. Leighton and S. Micali. Secret-key agreement without public-key cryptography. In *Advances in Cryptology - Crypto '93*, pages 456–479, 1993.

[14] J. Li, C. Blake, D. De Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *Proceedings of the 7th ACM International Conference on Mobile Computing and Networking*, 2001.

[15] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *Proceedings of the Tenth ACM Conference on Computer and Communications Security (CCS 2003)*, pages 52–61, October 2003.

[16] J. Newsome and D. Song. GEM: Graph embedding for routing and data-centric storage in sensor networks without geographic information. In *Proceedings of the First International Conference on Embedded Networked Sensor Systems*, pages 76–88, 2003.

[17] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. In *Seventh Annual International Conference on Mobile Computing and Networks (MobiCom 2001)*, pages 189–199, July 2001.

[18] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*, pages 96–108, 2003.

[19] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: a geographic hash table for data-centric storage. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, September 2002.

[20] J. Steiner, C. Neuman, and J. Schiller. Kerberos: An authentication service for open network systems. In *Usenix Winter Conference*, pages 191–202, January 1988.