Secure Random Key Pre-Distribution Against Semi-Honest Adversaries

Mike Burmester[1], Reihaneh Safavi-Naini[2], and Gelareh Taban[3]

[1] Florida State University, Tallahassee FL 32306, USA
{burmester}@cs.fsu.edu
[2] University of Calgary, Canada
{rei}@ucalgary.ca
[3] University of Maryland, College Park
{gelareh}@umd.edu

Abstract. Recently, Eschenauer and Gligor [EG02] proposed a model (the EG-model) for random key pre-distribution in distributed sensor networks (DSN) that allows sensors to establish private shared keys. In this model, each sensor is randomly assigned a set of keys, called a *key-ring*, from a secret key-pool. Two nodes can communicate securely by using a shared key (direct key) or via a chain of shared keys (key-path). The authors show how the key-ring size can be chosen so that nodes are *guaranteed* to be linked either by direct keys or by key-paths. Security of this system is proven for an *eavesdropping (passive) adversary.*

In this paper we assume the same key pre-distribution set-up but consider a *semi-honest* adversary. Semi-honest adversaries are privacy adversaries that have access to a fraction of the keys in the key pool—the *compromised keys*, but are otherwise passive, in the sense that they do not cause nodes to deviate from protocol executions (to remain undetectable). Since they can decrypt messages secured by key-paths with compromised keys, the security guarantees of the EG model break down.

We revisit the security of key establishment in the presence of such adversaries and make a number of contributions. First, we show that it is possible to choose the size of the key-rings so that any two nodes can exchange a private key securely in the presence of a semi-honest adversary. Second, we give a protocol that achieves this guarantee and prove its security. Third, we introduce a new efficiency parameter for the EG-model that allows the protocol designer to trade-off the communication required for key establishment with the key-ring size. Finally, we propose a concrete key establishment protocol (based on the DSR protocol) that guarantees security in the presence of a semi-honest adversary.

1 Introduction

A distributed sensor network (DSN) consists of large numbers of low powered wireless sensors that communicate with each other to provide information to a base station. Distributed sensor networks have found wide application such as sensing environmental data or monitoring infrastructures and areas that need constant surveillance [DS05,MCP+02]. In many applications, sensors are deployed in hostile environments where sensor communication must be protected. Once deployed, the nodes must operate on battery power and so traditional public-key cryptographic methods that require high computation and resource consumption, are not applicable. For symmetric key cryptography nodes must be able to efficiently establish private shared keys. Eschenauer and Gligor [EG02] pioneered an approach to solve this problem called *random key pre-distribution*, that assigns to sensors sets of keys such that pairs of sensors can efficiently establish a shared key, that is guaranteed to exist if the system parameters are chosen appropriately. In this approach each sensor node is assigned a set of keys, called a *key-ring*, that are randomly selected from a fixed pool of secret keys. Two nodes use a simple *shared key discovery* protocol to discover the keys they have in common. If they share a key, then they can use this to communicate securely. The shared keys

establish a *trust-graph* that supports private communication. In particular, if two nodes S, T do not have a common key, this graph can be used to exchange a private key through a *multi-hop trust-path* where a secret key chosen by S will be sent to T by *link-encryption* (in which the key is encrypted and decrypted at each trust-hop until it reaches T—see Section 2.3 for a discussion on link-encryptions in DSNs).

Let $\mathcal{N}$ be the set of sensor nodes and $\mathcal{N}^t$ the trust-graph. Eschenauer and Gligor modeled $\mathcal{N}^t$ as a random graph and used the random graph theory of Erdös and Rényi [ER60] to determine the size of the key-rings (for a given key-pool size) such that the trust-graph is guaranteed to be connected (that is, there is a trust-path connecting any two nodes). The security of this system is *guaranteed against an eavesdropping (passive) adversary*, i.e., an outsider adversary who can only observe communicated messages but does not have access to the keys of the key-rings.

In this paper we consider semi-honest adversaries. A *semi-honest* adversary is an *insider* adversary who has access to a fraction of the keys deployed in the network, for example by compromising a number of sensor nodes and learning their key-rings. The adversary however adheres to the prescribed protocol(s) (to remain undetectable while decrypting protected communication). Eschenauer and Gligor considered the effect of such adversaries and showed that the compromise of a single node would result in each trust-edge being compromised with probability k/N, where k and N are the sizes of the key-ring and the key-pool, respectively.[4] As the number of compromised nodes increases, the number of compromised keys increases and a large portion of the trust-edges, and hence the communication network, will become compromised. We note that a trust-path that has even one compromised edge will be compromised, as the adversary can decrypt the (link-encrypted) communication over that path. This means that a path may be compromised even if all the nodes on it are not compromised.

A number of authors [CPS03,DDHV03,LN03,ZXSJ03] have considered semi-honest adversaries and proposed ways to strengthen the security of the DSN system in such settings. In all these cases, the original random graph modeling and analysis remain the same and the modifications involve either strengthening the requirement for establishing trust-edges, or exploiting multiple paths to strengthen the security of trust-paths. The main shortcoming of all these approaches is that the security guarantee of the original EG model is lost and although security against semi-honest adversaries may be improved, there is *no guaranteed security* (in the sense of the EG model). Moreover, the improvements rely on assumptions such as *uniform deployment with a certain node density* that may not hold in many applications. A more complete review of these works is presented in Section 1.1.

Problem statement. We consider DSN deployments in which the adversary has access to a fraction of the keys assigned to sensors. The adversary is semi-honest, that is, adheres to the protocol specification but will use the compromised keys to undermine the privacy of DSN traffic. Our goal is to extend the EG-model of random key pre-distribution to provide guaranteed privacy in the presence of semi-honest adversaries.

Our contribution. We consider a *semi-honest α-adversary*: that is, a privacy adversary that has compromised a fraction of up to α of the edges of the trust-graph; equivalently, an adversary that has access to a fraction up to α of the keys of the key-pool—for instance, through the compromise of a subset of nodes. We address the following challenges:

1. *The existence of non-compromised trust-paths in the presence of an α-adversary.* Since the adversary is semi-honest, one cannot distinguish keys that are compromised from keys that are not. We show that to protect an EG system against α-adversaries, it is sufficient to scale the edge probability of the trust-graph by $(1 - \alpha)$, and use the connectivity of the resulting random graph to determine the size of the key-rings. We give an upper-bound approximation for the new size m_α of the key-rings, in terms of the

[4] More precisely, a trust-edge is compromised with probability k/N^*, where N^* is the size of the deployed key-pool.

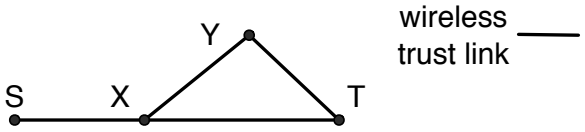
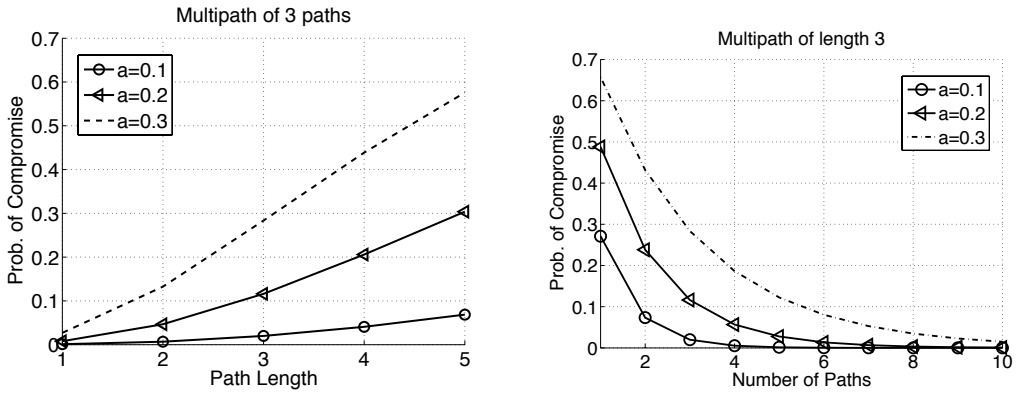


Fig. 1. The probability of successful multipath key establishment increases with the number of trust-paths used: if in this graph links are compromised with probability 0.5, then the probability of successful key establishment increases from 0.25, when using disjoint wireless paths, to 0.375 when using all (two) paths.

original key-ring size m and α, the power of the adversary. A more precise value for m_α can be obtained from Appendix A in [EG02] in terms of the key-pool size and the desired link probability. We show (Section 3.2) that this choice of key-rings ensures that, for any α-adversary: any two non-compromised sensors are linked by a key-path that is not compromised.

2. *Establish a private shared key between nodes.* The existence of non-compromised trust-paths in an EG system can be guaranteed by appropriate scaling of the key-ring size. However, in general, there are many trust-paths connecting nodes. Since we cannot distinguish compromised trust-paths from non-compromised paths when the adversary is semi-honest, it may be hard to find non-compromised paths. Note that in the original EG model (with passive adversaries), the existence of a trust-path is sufficient to guarantee private communication (by link-encryption)—more details in Section 2.2.

We propose (Section 3.2) a key establishment protocol that is secure (private) in the presence of semi-honest α-adversaries. The protocol has three steps. In the first, the source S finds *all* the trust-paths that link it to the target T. In the second S sends privately (link-encrypted) through each such path to T the share of a secret key, using a (ν, ν) secret sharing scheme, where ν is the number of paths. In the last step, T computes the key by combinining the shares it has received. If the size of the key-rings is determined as in (1.), then at least one of the shares will be private and thus the established key will be private.



(a) (b)

Fig. 2. The probability that the privacy of multipath key establishment is compromised increases with the length of trust-paths and decreases with the number of trust-paths—α is the fraction of compromised trust-edges.

We emphasize that using *all paths* together with key-rings of appropriate size is essential to our argument for providing privacy guarantees, and that no multipath protocol that uses the original key-ring size and a subset of paths (e.g., wireless-disjoint or node-disjoint paths) can provide privacy guarantees against an α-adversary. Figure 1 shows that the security of multipath key establishment improves as more paths are used to transmit secret shares. Assume that each link has probability $\alpha = 0.5$ to be compromised. Nodes S, T are connected via two trust-paths $SXYT$ and SXT. These paths have probability 0.5^3 and 0.5^2, respectively, of remaining uncompromised. If a single path is used to transport a private key, then this is secure with probability no more than 0.25. However if both paths are used to transport shares of the key, the probability increases to 0.375. Consequently the probability of successful key establishment increases when more paths are used (even if these are not disjoint). This is reflected in Figure 2(b) which illustrates that the probability that a multipath key establishment protocol is compromised decreases exponentially as more trust-paths are used to transmit key shares.

3. *The efficiency of private communication.* The protocol in (2.) establishes the existence of a secure trust-path in the presence of a semi-honest adversary. In practice the total number of paths between two nodes grows exponentially with the path length and so the communication cost of the protocol can become unwieldy. Note that the communication cost of key establishment between two nodes in the original EG model is linear in the length of the trust-path linking the nodes. However in the case of semi-honest adversaries, the communication cost grows exponentially with the path length (see the protocol in Section 4 and subsequent Remarks), and it is crucial to consider ways of reducing this cost. This is especially important in resource-constrained sensor networks where it has been shown that radio energy dominates total energy expenditure on a sensor [RSPS02].

We introduce bounds on the diameter of the trust-graph as a means of controlling the length of trust-paths and ensuring that every two nodes are connected by at least one *short* trust-path: e.g., a path that is shorter than a pre-specified number. We show how to choose the size of the key-rings such that the diameter of the trust-graph is bounded by D (Section 4) and so effectively reduce the communication cost of key establishment at the expense of increasing the size of key-rings. The protocol in (2.) above can be easily modified to incorporate this restriction. The trade-off between communication cost and key-ring size can be employed to design the parameters of the protocol for particular application scenarios. Note that constraining the length of trust-paths also improves the security of the multipath key exchange—see Figure 2(a).

4. *Design a private key establishment protocol with flexible efficiency.* We propose a concrete two-pass key establishment protocol, based on the Dynamic Source Routing (DSR) [JMH] protocol that is secure (private) in the presence of a semi-honest α-adversary (Section 5). The protocol has efficiency parameters that can be tuned for particular applications. In the first pass, the source S uses depth restricted flooding to allow the target T to discover all trust-paths of length at most D between S and T. In the second pass, T selects a random key and sends a share of it (using a (ν, ν) secret sharing scheme, ν the number of trust-paths) link-encrypted through each discovered path to S. S can then compute the key by combining these shares.

The protocol has two efficiency parameters: the key-ring size m and the diameter D of the trust-graph. A higher value for D requires deeper flooding in the network and so increases the communication cost, while a low value D, for example $D = 2, 3$, requires small flood depth and less communication. We show how to choose m for a given D and key-pool such that there is at least one non-compromised path of length at most D between S and T. This path will be discovered in the flooding stage in the first pass and thus the established key will be private.

An important contribution of our work is the clear separation of the deployment topology (e.g., the wireless distribution of nodes) from the trust-graph properties in providing protection against node compromising adversaries. We use the properties of the trust-graph to model semi-honest adversaries and

provide guaranteed security. This security will hold as long as the wireless (communication) graph and the trust-graph are connected. This is the minimum assumption for a deployed DSN and is the same as the assumption of the original EG model (see Section 2 for more details).

1.1 Related work

A number of papers in the literature have proposed improvements of the EG model to allow for semi-honest adversaries. However these assume that the deployed network has a particular structure (e.g., the nodes are uniformly distributed), and hence will not apply for some deployments. The distinctive aspect of our approach is that we make no such assumption. To allow for semi-honest adversaries we modify the parameters of the original random trust-graph by increasing the key ring-size. Our only assumption for the deployed wireless network is that it is (wirelessly) connected.

Eschenauer and Gligor observed that a semi-honest adversary who compromises a single node has probability k/N of learning the key of a trust-edge. Subsequent work improved the resiliency of the original EG system against semi-honest adversaries by: (i) strengthening link security [CPS03], and (ii) using multipath key establishment [CPS03,ZXSJ03,LZ05].

To strengthen the link-keys in the original EG model, q-composite schemes were proposed in [CPS03] for which a trust-link is established if at least q keys are shared between two nodes. To ensure graph connectivity, the key-ring size is increased. In the resulting scheme, a threshold is set such that, if the adversary compromises fewer nodes than the threshold, the q-composite scheme will be more resilient than the original EG system; otherwise it is less resilient. The scheme however, does not provide any guarantees for the security of the key establishment for semi-honest adversaries.

We next discuss improvements based on multipath key establishment. Dispersing secret information for improving security was first proposed by Rabin [Rab89] and later applied by Tsirigos and Haas [TH04] to alleviate path failure (e.g., packet dropping) in routing. Chan et al. [CPS03] proposed this approach to improve the security of the EG scheme against semi-honest adversary. This scheme, requires that pairs of nodes find all 2-hop *disjoint wireless paths* between them, and send a share of secret key over each path. They analyzed their scheme using a geometric approach, *assuming a uniformly deployed network*, and computed the average number of 2-hop disjoint secure wireless paths between two nodes. This number was used to determine the number of shares of the key. Du et al. [DDHV03] extended this analysis to 3-hop disjoint paths. Although both schemes adjust the deployed node density and key-ring size to ensure that a small average number of paths can be established between two nodes, they cannot offer any guarantee in arbitrary deployments.

An important limitation of the geometric analysis is that for longer paths, the complexity of the computation for finding the probability of the existence of disjoint paths becomes unmanageable. Huang et al. [HMvdLM07] address this problem in *non-adversarial settings* and for *uniformly deployed networks*. They computed the probability that a node is connected with its h-hop neighbors.

Note that if disjoint wireless paths are used (as in [CPS03]) then only a subset of all available trust-paths will be used to transmit (link-encrypted) secret key shares. This means that even with proper scaling of the key-ring size, there may be no non-compromised paths between some pairs of nodes and therefore any protocol that uses subsets of possible paths will fail for such pairs.

Zhu et al. [ZXSJ03] also propose a multipath solution for improving security against a semi-honest adversary. The scheme transports secret key shares over w 2-hop disjoint trust-paths between nodes, where w is a security parameter. The scheme however, does not provide any guarantees that w paths can always be found. When w paths are not found, the protocol is either aborted and delayed until network conditions change (due to node movement), or a lower security level link is established. The first case works only in dynamic networks and the second case reduces the security level of the *whole* system as a system is only as secure as its weakest link.

Li et al. [LZ05] present a node-disjoint multipath routing protocol that relies on [LC04] to first find the node-disjoint paths and then exchanging shares of a secret key. The authors do not offer any guarantees of success. Additionally by assuming node-disjoint paths, only a subset of all trust-paths is found, which reduces the probability of success (refer to Figure 1).[5]

Finally the resilience of key pre-distribution systems can be improved by enforcing structure on the key-pool [DDHV03,LN03,LS05]. Although schemes that use this approach have some interesting properties, they generally have the limitation that the security of the system *completely* breaks down if the number of compromised nodes is higher than a certain threshold, a design parameter of the system. Also the security guarantees in these systems are obtained at the expense of reduced flexibility in the system parameters [LS05,cY07], lack of scalability [DDHV03] or extensibility [cY07] as well as pre-deployment knowledge which is not always practical [LN03].

2 Preliminaries

2.1 Random Key Pre-distribution

A random key pre-distribution scheme has the following parameters:

- $\mathcal{S} = \{S_1, \ldots, S_n\}$ be a set of n sensors
- $\mathcal{K} = \{K_1, \ldots, K_N\}$ be a set of N keys, the *key-pool*
- $\{\mathcal{K}_i\}_{i \in \mathcal{S}}$ be a set of *key-rings*, where $\mathcal{K}_i \subset \mathcal{K}$ and $|\mathcal{K}_i| = m$ is fixed.

Each sensor S_i is given a key-ring $\mathcal{K}_i$ which is a randomly selected subset of a key-pool $\mathcal{K}$. Two sensors that share at least one key in their key-rings have a *trust-link*. This defines a graph $\mathcal{N}^t$, called the *trust-graph* or *key-graph*, whose node-set is $\mathcal{S}$ and edge-set is the set of trust-links. Since the key-rings are randomly selected, the probability p of two nodes having at least one shared key (and so being connected) is independent of the two nodes. This means that the trust-graph can be modeled as a random graph $\mathcal{G}(n, p)$ with n nodes and edge probability p.

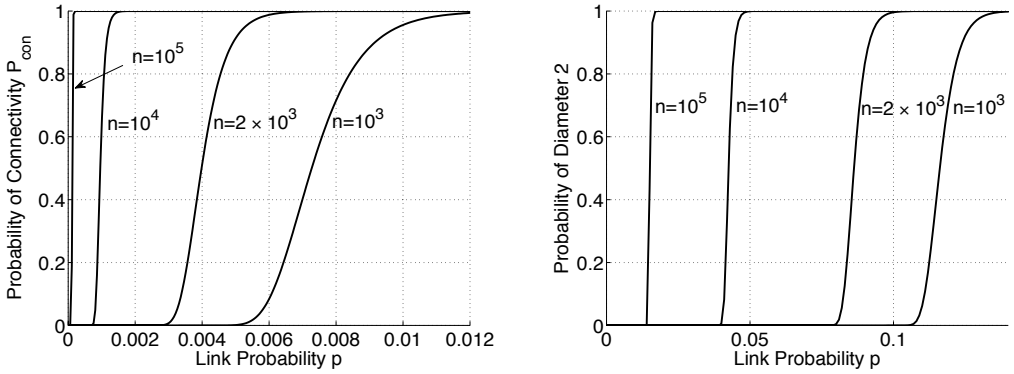
Random graphs Erdös and Rényi [ER60,Bol01] studied random graphs $G(n, p)$ on n nodes in which each edge occurs independently with probability p. They showed that there are certain classes of (monotone increasing) properties for which there are sharp thresholds for the probability p, which if crossed, will cause the graphs to 'almost surely' acquire these properties as $n \to \infty$. As noted by Eschenauer and Gligor, graph connectivity is one such property. The connectivity threshold is given by:

Theorem 1. [Bol01] *Let $G \in \mathcal{G}(n, p)$ with $p = \frac{1}{n}(\log n + c + o(1))$, c is a fixed real number. Then the probability P_{con} that G is connected converges to $e^{-e^{-c}}$ as $n \to \infty$.*

In other words to ensure connectivity with probability P_{con} the required edge-probability p for a network of size n is bounded by, $p = \ln n/n + c/n$, as $n \to \infty$. Here c determines the probability P_{con} that the graph is connected. This theorem is used to determine p, for a required P_{con} and n, and hence the key-ring size for a given key-pool.

In this paper we consider a second monotone property, the *diameter of a graph*. (The diameter of a graph is the greatest distance between any two of its nodes.) The threshold for the graph diameter is given by:

[5] In fact, the number of disjoint paths between two nodes, is less than the number of wireless disjoint paths (as used by Chan et al. [CPS03]), which is smaller than the total number of trust-paths.



(a) The probability that $\mathcal{G}(n,p)$ is connected for different values of p, n.

(b) The probability that the diameter of $\mathcal{G}(n,p)$ is $D = 2$ for different values of p, n.

Fig. 3.

Theorem 2. [Bol01] *Let $c > 0$ be a constant, $D \geq 2$ a natural number, and p satisfy: $p^D n^{D-1} = \log(n^2/c)$. Suppose that $pn/(\log n)^3 \to \infty$. Then $G \in \mathcal{G}(n,p)$ has diameter D with probability $P_D = e^{-c/2}$ and diameter $D+1$ with probability $1 - e^{-c/2}$ as $n \to \infty$.*

For a graph of size n, for large enough n, Theorem 2 gives the threshold probability p implicitly by $p^D n^{D-1} = \log(n^2/c)$, for which the value of the diameter is D with probability P_D, where as before P_D is defined as a function of c.

Figures 3(a) and 3(b) respectively plot the points (p, P_{con}) and (p, P_D) for four values of n: $10^3, 2 \times 10^3, 10^4, 10^5$, taking $o(1) = 0$ and $\log = \ln$. The figures show the rapid phase transition of the random graphs with respect to graph connectivity and diameter $D = 2$.

2.2 Network model

A deployed key-ring DSN is associated with a wireless network $\mathcal{N}^w$. The union of the trust-graph $\mathcal{N}^t$ and $\mathcal{N}^w$ is a network $\mathcal{N}^{w+t}$ with edge-set the union of the edge-sets of $\mathcal{N}^w$ and $\mathcal{N}^t$. The trust-graph $\mathcal{N}^t$ when projected onto $\mathcal{N}^w$ induces a trust infrastructure $\mathcal{N}^{wt}$ on the deployed network whose edge-set is the intersection of the edge-sets of $\mathcal{N}^t$ and $\mathcal{N}^w$. Figure 4 shows the relationship between these graphs.

We shall assume that each node X_i in the network has a unique identifier $id(X_i)$ and that all keys have labels recognized by the nodes in the network. Once nodes are deployed, a neighbor discovery algorithm is executed. We assume that the neighborhood relation is symmetric (bi-directional), that is, a node discovers those and only those nodes that can discover it, and that neighborhood discovery is asynchronous and distributed.

Discovering neighbors in $\mathcal{N}^w$ and $\mathcal{N}^t$. Several wireless neighbor discovery algorithms have been proposed in the literature (see e.g., [BE81,NNS98,ZHS03]). These algorithms are employed at the network layer. Discovering the trust-neighbors of a node X_i requires identifying all nodes in the network that share at least one key with X_i. A straightforward protocol is to have each node broadcast (flood throughout the DSN) in clear-text its identifier and a list of the labels of all the keys that are in its key-ring. Using this information a node can locally construct $\mathcal{N}^t$ and find a path that links it between to non-neighbor nodes in the trust-graph. Protocols discovering neighbors in N^t are discussed in [EG02]. Note that these algorithms can also be used to determine the wireless distance of trust-neighbors.

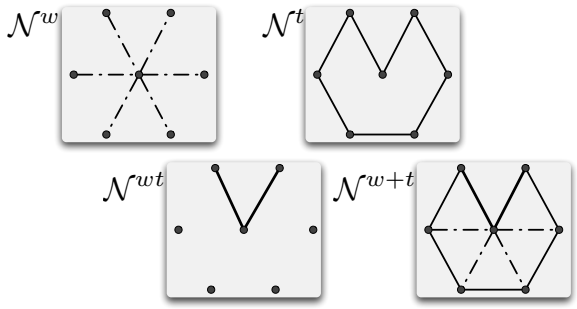


Fig. 4. A deployment in which the wireless network $\mathcal{N}^w$, the trust network $\mathcal{N}^t$ and the union network $\mathcal{N}^{w+t}$ are connected, but the deployed trust network $\mathcal{N}^{wt}$ is not.

2.3 Key establishment in non-uniform deployments

Let S be the source and T the target. We distinguish two ways in which S and T can exchange (establish) a private key. Let κ be a random key selected by S.

Link-encryption in $\mathcal{N}^{wt}$. This method is used when the source S and the target T are connected by a path $\pi^{wt}(S,T) = (S, S_1, \ldots, S_u, T)$ in $\mathcal{N}^{wt}$. In this case each wireless edge of the path is assigned a key from the key-pool. To send the key κ to T privately, S encrypts κ with the key it shares with S_1 and sends the encryption to S_1; S_1 decrypts it to get κ, and then re-encrypts it with the key it shares with S_2, and so on, until κ reaches T encrypted with a key it shares with S_u.

In practice the topology of the deployed wireless network $\mathcal{N}^w$ is not known in advance (i.e., when the key-rings are assigned to sensors) and any assumption regarding its uniformity (e.g., the degree of nodes) will not hold in some deployments. Figure 4 gives an example of a deployment with 7 nodes. The wireless network N^w is a star topology and highly nonuniform, while the trust network $\mathcal{N}^t$ is uniform (degree 2). Both graphs are individually connected (as is their union $\mathcal{N}^{w+t}$), but the intersection graph, $\mathcal{N}^{wt}$, is highly disconnected. However, the union network $\mathcal{N}^{w+t}$ is connected.

Eschenauer and Gligor showed in [EG02] how to adjust the key-ring size so as to guarantee connectivity in $\mathcal{N}^{wt}$ for uniform deployments. However, for arbitrary deployments such as those above, the network $\mathcal{N}^{wt}$ is not connected. The minimum connectivity requirement however is that $\mathcal{N}^t$ and $\mathcal{N}^w$ are individually connected. In such cases secure key establishment can be achieved with link-encryption in $\mathcal{N}^{w+t}$.

Link-encryption in $\mathcal{N}^{w+t}$. This method is used when the source S and the target T are not connected in $\mathcal{N}^{wt}$ but are connected in both $\mathcal{N}^t$ and $\mathcal{N}^w$. Suppose that S, T are connected by the trust-path $\pi^t(S,T) = (S, S_1, \ldots, S_u, T)$ in $\mathcal{N}^t$. To send κ to T privately, S encrypts it with the key it shares with S_1 and sends this to S_1 over a wireless path that is guaranteed to exist in $\mathcal{N}^w$. When S_1 gets the encryption, it decrypts it and then re-encrypts it with the key it shares with S_2 and forwards it through the wireless network to the next node on the trust-path, until eventually the key reaches T.

We make the following observations regarding link-encryptions.

1. For both types of encryption the session key κ is private if and only if the keys assigned to the edges of $\pi^t(S,T)$ are not compromised. If the key assigned to a trust-edge is compromised, then the path and hence the session key and all future encrypted communication using this key, will be compromised.

2. Link-encryption in $\mathcal{N}^{wt}$ has lower communication cost than link-encryption in $\mathcal{N}^{w+t}$. This is because sending a message over a trust-edge requires traversing a single edge in $\mathcal{N}^{w}$ while in $\mathcal{N}^{w+t}$, sending a message over a trust-edge requires traversing multiple edges of the wireless graph.

From the above discussion we see that:

Minimum connectivity requirement. To establish a private session key in the presence of a passive adversary (an eavesdropper) it is necessary and sufficient that the trust-graph $\mathcal{N}^t$ and the wireless network $\mathcal{N}^w$ be connected.

3 Securing Communication Against Semi-Honest Adversaries

3.1 Adversary model

We shall assume that the random key pre-distribution process is trusted and that key-rings are assigned to sensors prior to deployment in a secure environment. This implies that key pre-distribution and key compromise are independent events.

We consider a privacy adversary that is semi-honest. The adversary has access to a fraction α of the deployed keys in the key-pool, and is passive. The goal of the adversary is to undermine the privacy of communication without being detected. The probability that a key associated with a trust-edge in $\mathcal{N}^t$ is not compromised is $(1-\alpha)$. We can therefore model the non-compromised trust-graph by a random graph $G(n, p(1-\alpha))$. This is a subgraph of $\mathcal{N}^t$ which we denote by $\mathcal{N}^t_\alpha$.

Security against α-adversaries. Assume that the key-ring size m is designed for edge-probability p that guarantees trust-connectivity with probability $P_{con} = A$, against a passive adversary (the original EG model). Here A can be seen as the level of security guarantee offered by the system.

To guarantee security in the presence of an α-adversary with probability A, we need to choose key-rings of size m_α, and hence the associated trust-graph $G(n, p_\alpha)$ must be such that the subgraph $G(n, p_\alpha(1-\alpha))$ is connected with probability $P_{con} = A$. From Theorem 1 the edge-probability of random graphs with fixed probability of connectivity $P_{con} = A$, is a function of the number of nodes n and c. To obtain $P_{con} = A$ in $G(n, p_\alpha(1-\alpha))$, we therefore need to have $p_\alpha(1-\alpha) = p$: that is, choose key-rings such that probability that two arbitrary key-rings have at least one common key is $p_\alpha = p/(1-\alpha)$.

Scaling the key-ring size. Knowing the edge probability p of trust-graph allows us to precisely find the key-ring size m_α that guarantees privacy for α-adversaries. This is done using the methodology described in Appendix A of [EG02]. Table 1 shows how the key-ring size m_α increases with the value of α.

α	0	0.1	0.2	0.3	0.4	0.5
m_α	43	45	49	51	56	62
m^*_α	43	64	67	72	78	86

Table 1. The relation between the exact key-ring size m_α and the upper bound of the key-ring size m^*_α for α-adversaries to guaranteed privacy, in a network with 10^3 nodes and key-pool size 10^5. Note that $\alpha = 0$ is equivalent to the case when we only consider a passive adversary.

We also provide (in Appendix B) an upper bound m^*_α for the key-ring size m_α that will guarantee privacy in the presence of a semi-honest α-adversary in terms of the key-ring size m that will guarantee privacy for passive adversaries. We show that assuming $N > m(2m-1)$ (e.g., key ring size up to 70 for key-pool size 10000), we have $m^*_\alpha \leq 2m/(1-\alpha)$. That is, *security against an α-adversary requires that key-rings be scaled up by a factor of no more than* $2/(1-\alpha)$. Table 1 shows the actual relationship between the exact key-ring size m_α and the upper bound m^*_α.

3.2 Protocol description

In Section 2.3 we showed how privacy for a passive adversary (an eavesdropper) in a DSN can be established by using link-encryption when the networks N^t and N^w are connected. However, with semi-honest adversaries, link-encryption becomes insecure and cannot be used to transport private keys. We now propose a key establishment protocol that guarantees privacy in the presence of an α-adversary, provided the size of the key-rings is chosen appropriately. Let S be the source and T the target. The protocol has four steps.

1. S finds *all paths* $\pi^t_i(S,T)$ in $\mathcal{N}^t$ that connect it to T. Let $\nu_{ST} = |\{\pi^t_i(S,T)\}|$.
2. S chooses ν_{ST} random numbers $r_1, \ldots, r_{\nu_{ST}} \in Z_q$, q large, and computes the key $\kappa_{ST} = \sum_{i=1}^{\nu_{ST}} r_i \bmod q$.
3. For each $i = 1, \ldots, \nu_{ST}$, S sends to the target T the share r_i link-encrypted over the path $\pi^t_i(S,T)$.
4. T receives link-encrypted shares $r_1, \ldots, r_{\nu_{ST}}$ and computes $\kappa_{ST} = \sum_{i=1}^{\nu_{ST}} r_i \bmod q$.

Theorem 3. *The key-establishment protocol described above guarantees privacy against semi-honest α-adversaries in an EG key pre-distribution DSN, if we increase the size of the key-rings by a factor of no more than* $2/(1-\alpha)$.

The proof is based on the fact that there is at least one uncompromised path between any two nodes and so at least one of the shares of the key κ_{ST} is not known to the adversary. The privacy of the key follows from the fact that in the sum $\sum_{i=1}^{\nu_{S,T}} r_i \bmod q$, one unknown share will result in the key itself being unknown.

Remarks

1. *Efficiency*: The protocol requires that *all* trust-paths between the two nodes, are known. This information can be obtained by using the key discovery protocol described in [EG02]. However, this could be computationally expensive for low powered sensors. In the next section we show how cost can be reduced by using larger key-rings.
2. *Multipath protocols*: Protocols that use *disjoint multipaths* to assure privacy with semi-honest adversaries will fail in the general case. The reason is that these protocols require that there are sufficiently many disjoint paths so that at least one of them is not compromised. In our setting the (expected) number of compromised edges ($\alpha pn(n-1)/2 = \mathcal{O}(n^2)$) can be much larger then the number of disjoint paths (bounded by the expected degree of nodes $p(n-1) = \mathcal{O}(n)$).
3. Because the adversary is semi-honest, it is not possible to distinguish between compromised and non-compromised paths. If the key-ring size is chosen at its minimum required value, then lower values will fail to guarantee the existence of non-compromised paths. The protocol presented in this section is minimal for the required security guarantee.

4 Improving Efficiency

The protocol in Section 3.2 requires that a share of a private key be sent (link-encrypted) over each one of the trust-paths that link the nodes S, T. Since the communication overhead over long paths can be excessive for DSNs with constrained resources this protocol is not practical.

In this section we show how to use a result from random graph theory to increase the communication efficiency at the cost of higher node storage (larger key-ring size). The basic idea is to ensure that the value D of the diameter of the non-compromised trust-graph is small. (Graphs that are not connected have diameter $D = \infty$.)

Restricting the diameter of $\mathcal{N}^t$. Using Theorem 2, the key-ring size can be chosen so that for any two nodes there is at least one trust-path of length at most D that links them with probability P_D. Suppose the adversary is α-semi-honest. Then the non-compromised trust-graph $\mathcal{N}_\alpha^t$ is a random graph $\mathcal{G}(n, p(1-\alpha))$, and we can take the size of the key-rings in Theorem 2 to be such that the value of the diameter of $\mathcal{G}$ is small, say $D = 3$ or $D = 4$, with probability P_D. This will reduce the communication cost of the key establishment protocol significantly. We use the same protocol as in Section 3.2, except for the first step which is replaced by:

1. (*Revised*) The source S finds all paths $\pi_{i,D}^t(S,T)$ of length at most D in $\mathcal{N}^t$ that link it to the target T. Let $\nu_{ST} = |\{\pi_{i,D}^t(S,T)|$.

4.1 Analysis

The protocol has four parameters, (n, α, P_D, len^*).

n: is the minimum size of the network for the provided security guarantee.

α: is the fraction of the compromised keys tolerated by the protocol.

P_D: is the success probability of the protocol.

len^*: the efficiency parameter of the protocol.

Choosing the algorithm parameters. We determine the edge probability p_α of the non-compromised trust-graph $\mathcal{N}_\alpha^t$ as follows—refer to Theorem 2.

1. From $P_D = e^{-c/2}$ we get: $c = -2 \cdot \log P_D$.
2. From $p^{len^*} n^{len^*-1} = \log(n^2/c)$ we get: $p = (\log(n^2/c)/n^{len^*-1})^{1/len^*}$.
3. From $(1-\alpha)p_\alpha = p$ we get $p_\alpha = p/(1-\alpha)$.

The probability p_α can then be used to determine the size m_α of the key-ring as in [EG02]. Observe that the security guarantee will hold for larger n because the diameter D is a monotone property for random graphs.

4.2 An example

To understand our scheme, we present a simple example. We consider a sensor network with 10^4 nodes, where the key-pool size is 10^5 and the adversary has compromised $\alpha = 25\%$ of the deployed keys in the network. In this example, we first compute the key-ring size that would secure the sensor network against a semi-honest 0.25-adversary *without* efficiency considerations and then show how the key-ring size must be increased when efficiency is a requirement. In particular, we show that by constraining the diameter of the trust-graph to $D = 3$ with probability $P_D = 0.999$ (and $D = 4$ with probability 0.001), in the presence of a semi-honest 0.25-adversary, the key-ring size increases from 13 keys to 29 keys.

To compute the key-ring size for a non-constrained trust-graph (i.e., when no graph diameter constraint is imposed), we simply use the method presented in the original EG scheme. Given a graph connectivity probability of $P_c = 0.999$ in a network of $P = 10^4$ nodes and a key-pool size $N = 10^5$, we first find $c = 6.91$ so that two nodes can establish a secure connection with probability $p = 0.0016$. If 25% of the

edges of the trust-graph are compromised, then the new edge-probability is $p_\alpha = p/(1-0.25) = 0.0021$. Therefore, to secure the system against a semi-honest adversary without efficiency constraints, the size of the key-rings must be at least 13 keys.

Next, we constrain the diameter of the trust-graph so that with probability $P_D = 0.999$ the diameter $D = 3$. We use the algorithm described in Section 3.2. From Step 1, we get $c = 0.002$. Using this value and $len^* = 3$, Step 2 gives us the trust-edge probability $p = 0.00627$. Therefore, the random graph $\mathcal{G}(10^4, 0.00627)$ has diameter $D = 3$ with probability $P_D = 0.999$ as $n \to \infty$, see Figure 3(b). If 25% of the edges of the trust-graph are compromised, then the new edge-probability is $p_\alpha = p/(1-0.25) = 0.00836$. For this configuration, assuming a key-pool size $N = 10^5$, we see that the key-ring size should be at least 29 keys, by using the approach in [EG02].

Discussion 1. Our analysis is for the trust-graph $\mathcal{N}^t$ and its non-compromised subgraph $\mathcal{N}^t_\alpha$. The key-ring sizes can be determined for deployments with uniform neighborhood distribution as in [EG02]. That is, assuming a uniform node density (degree), the probability p can be adjusted to guarantee connectivity or diameter D, in $\mathcal{N}^{wt}$.

2. If all the trust-paths linking S and T are compromised then S, T cannot communicate privately and it will not be possible to establish a private key between S and T. A trust-graph designed to have diameter D puts an upper-bound on the cost of establishing a secure key between two arbitrary points. For a deployed network with fixed key-ring size, one can determine len^* for the first pass of the protocol to guarantee security at a desired level.

3. Our key establishment protocol uses *all paths of up to certain trust-length*. The two main advantages of this approach are: (i) security guarantees for correctly chosen key-ring sizes and (ii) efficiency, as we do not require any processing of the paths to verify properties such as node or edge disjointness.

5 A Practical Protocol

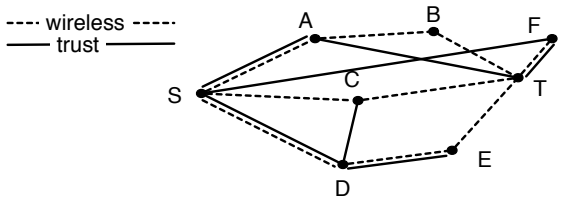
We next propose a practical key establishment protocol that guarantees privacy in the presence of semi-honest adversaries. To improve the efficiency of the protocol, we select the parameters of the system so that the diameter of the non-compromised trust-graph is reduced to len^*.

The protocol is based on the widely used DSR protocol [JMH] and has two passes. The first pass discovers all the trust-paths in the trust network $\mathcal{N}^t$ between the source S and target T that have length at most len^*. In the second, T sends shares of a private key link-encrypted over each discovered path to S.

Note that every trust-path $\pi^t(S,T)$ between S and T can be extended to a wireless route $\pi^w(S,T)$ between S and T. Let π^t be a discovered partial trust-path, starting at S, and π^w one of its wireless extensions. We use the following convention: a node $X \in \pi^w$ that belongs to π^t is denoted as X^* on π^w; in particular the source is denoted by S^*. If $X \notin \pi^t_i$ it is simply denoted as X. This notation will make it possible to use the route π^w for link-encryption: starred nodes will decrypt/encrypt data and forward it whereas non-starred nodes will simply forward data. Finally, the subpath of π^w consisting of its last starred node and all following (non-starred) nodes is denoted by $trunc(\pi^w)$.

We shall assume that each node X knows all its wireless neighbors $Y \in n^w(X)$ as well as all its trust-neighbors $Z \in n^t(X)$ and their wireless distance $d^w = d^w(X,Z)$ (the minimum number of wireless hops that separate them).

We now present a high-level description of the protocol. The detailed algorithm is provided in Appendix A. Figure 5 illustrates how all the trust-paths of length at most 3 are discovered between nodes S and T for a particular network.



Pass 1: Discover all trust paths of length at most 3

Rnd 1 $S \rightarrow :\quad \langle (S^*), A, 1, 3\rangle,\ \langle (S^*), F, 3, 3\rangle,\ \langle (S^*), D, 1, 3\rangle$

Rnd 2 $A \rightarrow :\quad \langle (S^*, A^*), T, 2, 2\rangle,\ \langle (S^*, A), F, 2, 3\rangle$
$C \rightarrow :\quad \langle (S^*, C), F, 2, 3\rangle$
$D \rightarrow :\quad \langle (S^*, D), F, 2, 3\rangle,\ \langle (S^*, D^*), C, 1, 2\rangle\ ,\ \langle (S^*, D^*), E, 1, 2\rangle$

Rnd 3 $B \rightarrow :\quad \langle (S^*, A^*, B), T, 1, 2\rangle,\ \langle (S^*, A, B), F, 1, 3\rangle$
$T \rightarrow :\quad \langle (S^*, C, T), F, 1, 3\rangle$
$D \rightarrow :\quad \langle (S^*, C, D), F, 1, 3\rangle$
$C \rightarrow :\quad \langle (S^*, D, C), F, 1, 3\rangle$
$E \rightarrow :\quad \langle (S^*, D, E), F, 1, 3\rangle$

Rnd 4 T stops : $\quad \langle (S^*, A^*, B, T^*)\rangle.$
$F \rightarrow :\quad \langle (S^*, C, T, F^*), T, 1, 2\rangle$

Rnd 5 T stops : $\quad \langle (S^*, C, T, F^*, T^*)\rangle.$

Fig. 5. An example of the execution of Pass 1, in which all trust-paths between S and T are discovered. Note: $\langle path, X, d, len^*\rangle$ is a route request bounded by the wireless-distance d to X and the trust-distance len^* to T. Initially $len^* \leftarrow 3$.

Pass 1. Discovering trust-paths (downstream)
The source S broadcasts a route request

$$m_{rreq} = \langle sn, \pi^w, d^w, X, T, len^*\rangle$$

for each one of its trust-neighbors $X \in n^t(S)$: sn is a sequence number, $\pi^w = (S^*)$, d^w is the wireless distance between S and T, and len^* is an upper bound on the trust-length.

Suppose that an intermediate node X_i receives the route request m_{rreq}. We distinguish two cases: $X_i \neq X$ and $X_i = X$. In the first case, if $d^w > 1$ and $X_i \notin trunc(\pi^w)$ (to prevent wireless cycles), X_i broadcasts m_{rreq} with: $\pi^w \leftarrow \pi^w || X_i$ and $d^w \leftarrow d^w - 1$. If $d^w = 1$ and $X \in n^w(X_i)$ then X_i unicasts m_{rreq} with: $\pi^w \leftarrow \pi^w || X_i$ and $d^w \leftarrow 0$.

Next let $X_i = X$. If $len^* = 1$ and $T \in n^t(X_i)$ then X_i broadcasts m_{rreq} with: $\pi^w \leftarrow \pi^w || X_i^*$, $X \leftarrow T$, $d^w \leftarrow d^w(X_i, T)$ and $len^* = 0$. If $len^* \geq 2$ then X_i broadcasts to each one of its trust-neighbors $Z \in n^t(X_i)$, $Z \notin \pi^t$ (to prevent trust-path cycles) m_{rreq} with: $\pi^w \leftarrow \pi^w || X_i^*$, $X \leftarrow Z$, $d^w \leftarrow d^w(X_i, Z)$ and $len^* = len^* - 1$.

Pass 2. Key establishment (upstream)
When target T receives m_{rreq} with wireless path π^w and sequence number sn, it selects a random key share $r_i \in Z_q$ and unicasts a route reply,

$$m_{rrep} = \langle sn, \pi^w, E\rangle\,,$$

where $E = E_K(r_i)$ is the encryption of r_i with key K shared with the last starred node on π^w. The route reply is sent upstream link-encrypted to S: starred nodes will decrypt/encrypt E and unicast the

new encryption, while unstarred nodes will simly unicast it. The source S will decrypt all encryptions E with sequence number sn, and compute the sum $k_{ST} = \sum_i r_i \bmod q$.

In this protocol all nodes have timers. When they first receive m_{rreq} with sequence number sn they set an sn-timer. The timers are used to control their actions for the flows sn. On timeout, the node actions are terminated. In particular, the source node S will compute the key k_{ST} for session sn, which is shared with T.

The algorithm will discover all trust-paths of length at most len^* and so the established key will be private if there is at least one non-compromised trust-path of at most this length linking S and T.

Analysis. For any α-adversary we can choose the parameters of the key-ring DSN so that the key establishment algorithm will succeed with a pre-determined probability P_D. Indeed, if the edge probability of trust-graph $\mathcal{N}^t$ is p then the probability that a link is not compromised is $(1-\alpha)p$. The algorithm will succeed with probability P_D if p is chosen such that $(1-\alpha)p$ is sufficiently large to guarantee with probability P_D that the diameter of this graph (the non-compromised graph) is len^*—for example, $D = 2$ or $D = 3$ (this implies that this graph is also connected).

6 Concluding Remarks

Eschenauer and Gligor launched a new direction of research with their seminal paper on random key predistribution in distributed sensor networks. They used a random graph to model the trust infrastructure of their system, with the sensors corresponding to nodes and the edges corresponding to trust-links. This system is proven secure for passive (eavesdropping) adversaries. Although there have been numerous subsequent works where stronger adversary scenarios are considered, none extended the security guarantees of the key establishment against stronger adversaries *in the random graph model*. This is important as it allows security guarantees to be made *independent* of the deployment model.

In this work, we consider semi-honest adversaries that have access to a fraction of the deployed keys of the network, but do not cause nodes to deviate from the specified protocol. We show how the original key pre-distribution system can be strengthened to account for semi-honest adversaries in the random graph model. We also introduce an efficiency parameter that constrains the trust-path length between two nodes. Finally, we present and analyze a practical key exchange protocol that guarantees privacy between all nodes in the presence of a semi-honest adversary.

Our contribution is a first step in extending the random graph model of Eschenauer and Gligor. In the future we plan to consider more powerful Byzantine adversaries that can cause nodes to behave arbitrarily. Additionally, we plan to use simulations and example deployment scenarios that allow us to quantify our efficiency parameter.

References

[BE81] D. Baker and Anthony Ephremides. The architectural organization of a mobile radio network via a distributed algorithm. *IEEE Transactions on Communications*, 29(11):1694–1701, 1981.

[Bol01] B. Bollobás. *Random Graphs (2nd edition)*. Cambridge University Press, 2001.

[CPS03] Howan Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, 2003.

[cY07] Seyit A. Çamtepe and Bülent Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 15(2):346–358, 2007.

[DDHV03] Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS)*, pages 42–51, 2003.

[DS05] D. M. Doolin and N. Sitar. Wireless sensors for wildfire monitoring. In *Proceedings of SPIE Symposium on Smart Structures & Materials / NDE*, 2005.

[EG02] L. Eschenauer and V. Gligor. A key management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 41–47, New York, NY, USA, 2002.

[ER60] P. Erdös and A. Rényi. The evolution of random graphs. *Magyar Tud. Akad. Mat. Kutató Int. Közl*, 5:17–61, 1960.

[HMvdLM07] Dijiang Huang, M. Mehta, A. van de Liefvoort, and D. Medhi. Modeling pairwise key establishment for random key predistribution in large-scale sensor networks. *IEEE/ACM Transactions on Networking*, 15(5):1204–1215, Oct 2007.

[JMH] David B. Johnson, David A. Maltz, and Yih-Chun Hu. The dynamic source routing protocol for mobile ad hoc networks - ietf manet working group.

[LC04] X. Li and L. Cuthbert. Node-disjointness-based multipath routig for mobile ad hoc networks. *Proceedings of the 1st ACM International Workshop on PE-WASUN*, pages 23–29, 2004.

[LN03] Donggang Liu and Peng Ning. Establishing pairwise keys in distributed sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, pages 52–61, 2003.

[LS05] Jooyoung Lee and Douglas R. Stinson. Deterministic key predistribution schemes for distributed sensor networks. In *Selected Areas in Cryptography*, pages 294–307, 2005.

[LZ05] Hui Ling and Taieb Znati. End-to-end pairwise key establishment using multi-path in wireless sensor network. In *IEEE Global Telecommunications Conference*, 2005.

[MCP+02] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97, 2002.

[NNS98] T. Narten, E. Nordmark, and W. Simpson. Neighbor discovery for IP version 6. In *RFC 2461, Internet Engineering Task Force*, December 1998.

[Rab89] Michael O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *Journal of the ACM (JACM)*, 36(2):335–348, April 1989.

[RSPS02] V. Raghunathan, C. Schurgers, Sung Park, and M.B. Srivastava. Energy-aware wireless microsensor networks. *IEEE Signal Processing Magazine*, 19(2):40–50, Mar 2002.

[TH04] A. Tsirigos and Z.J. Haas. Analysis of multipath routing–part i: the effect on the packet delivery ratio. *IEEE Transactions on Wireless Communications*, 3(1):138–146, Jan. 2004.

[ZHS03] Rong Zheng, Jennifer C. Hou, and Lui Sha. Asynchronous wakeup for ad hoc networks. In *Proceedings of 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 35–45, 2003.

[ZXSJ03] Sencun Zhu, Shouhuai Xu, Sanjeev Setia, and Sushil Jajodia. Establishing pair-wise keys for secure communication in ad hoc networks: A probabilistic approach. In *11th IEEE International Conference on Network Protocols*, 2003.

A Protocol Description

We assume that each node X has: a counter ctr to generate sequence numbers sn, timers $timer_X$, and a cache $cache(X)$. Initially $route \leftarrow \perp$ and $cache(X) \leftarrow \perp$. Each node has a time-to-live ttl^{sn} timer associated with each sequence number it stores in its cache. When ttl^{sn} timeouts, the sequence number sn in the cache is discarded. We use the syntax defined in Section 5 for route requests, $m_{rreq} = \langle sn, route, d^w, X, T, len^* \rangle$, and $m_{rrep} = \langle sn, route, E \rangle$, for route replies. The value of $route$ is a list of nodes that specifies a wireless route (in $\mathcal{N}^w$). Some of the nodes are starred: these define a trust-graph (in $\mathcal{N}^t$). As in Section 5, the subroute of $route$ consisting of its starred nodes is denoted by $route^t$, and the subroute consisting of its last starred node and all following nodes is denoted by $trunc(route)$.

A key establishment algorithm

Pass 1 (downstream)

Source node S

Set: $sn \leftarrow ctr$, $ttl_S^{sn} \leftarrow timer_S$, $route \leftarrow (S^*)$, $k_{ST} \leftarrow 0$.

For each $Z \in n^t(S)$ set: $X \leftarrow Z$, $d^w \leftarrow d^w(S, Z)$

Broadcast: m_{rreq}

Intermediate node X_i

If m_{rreq} is received

If $X_i \neq X$, $X_i \notin trunc(route)$, $d^w \geq 1$

Then set: $route \leftarrow route||X_i$, $d^w \leftarrow d^w - 1$

If $d^w > 1$ then broadcast m_{rreq}; else if $X \in n^w(X_i)$ unicast m_{rreq}

If $X_i = X$, $len^* = 1$, $T \in n^t(S_i)$, then set: $route \leftarrow route||X_i^*$, $X \leftarrow T$, $d^w \leftarrow d^w(X_i, T)$, $len^* = 0$

Broadcast m_{rreq}

If $X_i = X$, $len^* \geq 2$, then for each $Z \in n^t(X_i)$, $Z \notin route^t$

Set $route \leftarrow route||X_i^*$, $X \leftarrow Z$, $d^w \leftarrow d^w(X_i, Z)$, $len^* = len^* - 1$

Broadcast m_{rreq}

Pass 2 (upstream)

Target node T

If m_{rreq} is received and $X \in n^t(T)$ then

If $sn \notin cache(T)$ then set $ttl_T^{sn} \leftarrow timer_T^{sn}$, store sn in $cache(T)$, select $r \in_R Z_q$

Compute the encryption $E_K(r)$ with the key K shared with with X

Set $E \leftarrow E_K(r)$, $route \leftarrow route||T^*$, $k_{ST} \leftarrow r$

Unicast m_{rrep}

If $sn \in cache(T)$ and $ttl_T^{sn} \neq 0$, then select $r \in_R Z_q$, compute $E_K(r)$ with the key K shared with X

Set $E \leftarrow E_K(r)$, $route \leftarrow route||T^*$, $k_{ST} \leftarrow k_{ST} + r$

Unicast m_{rrep}

If m_{rreq} is received and $X \notin n^t(T)$ then

If $sn \notin cache(T)$ then set $ttl_T^{sn} \leftarrow timer_T^{sn}$, store sn in $cache(T)$

If $d^w > 0$ then set $route \leftarrow route||T$

Unicast m_{rrep}.

If $sn \in cache(T)$, $ttl_T^{sn} \neq 0$, $d^w > 0$ then set $route \leftarrow route||T$

Unicast m_{rrep}.

Intermediate node X_i

If m_{rrep} is received then

If $sn \in cache(X_i)$, $ttl_{X_i}^{sn} \neq 0$, $X_i^* \in route$ then

Decrypt E with key K shared with the downstream starred neighbor of X_i on $route$ to get r

Re-encrypt r with key K' shared with the upstream starred neighbor of X_i on $route$ to get $E_{K'}(r)$

$E \leftarrow E_{K'}(r)$

Unicast m_{rrep}

Source node S

While $ttl_S^{sn} \neq 0$ do

If m_{rrep} is received with sequence number sn then

Decrypt E with key K shared the downstream starred neighbor of S to get r

$k_{ST} \leftarrow k_{ST} + r \bmod q$

When $ttl_S^{sn} = 0$ output k_{ST}

B An Approximation For the Key-ring Size

Let $\mathcal{G}(n, p_m)$ be the random graph of an EG system, m the key-ring size and $0 < \alpha < 1$.

Theorem 4. *A key-ring size $M = 2m/(1-\alpha)$ guarantees random graph probability $p_M > p_m/(1-\alpha)$, when $N > m(2m-1)$.*

Proof. Assume that node Y has been assigned a key-ring with m keys, and that we start drawing keys for the key-ring of node X from a key-pool of size N. We are interested in collisions when the drawn keys are not replaced. The probability that the first key drawn for Y is one of the m keys of Y is m/N. The probability of no collision is $1 - m/N$. The probability of no collision for the second drawn key is: $1 - m/(N-1)$; and so on. So the probability that X, Y have a shared key is:

$$p_m = 1 - (1 - \frac{m}{N})(1 - \frac{m}{N-1}) \cdots (1 - \frac{m}{N-m+1}).$$

This can be expended to get:

$$p_m \;=\; \sum_{0 \le i < m} \frac{m}{N-i} \;-\; \sum_{0 \le i < j < m} \frac{m^2}{(N-i)(N-j)} \;+\; \cdots . \tag{1}$$

The second term is bounded by,

$$\sum_{0 \le i < m} \frac{m}{(N-i)} \sum_{i < j < m} \frac{m}{N-j} < \sum_{0 \le i < m} \frac{m}{(N-i)} \frac{m(m-i-1)}{N-m+1} \;,$$

which if we assume that $N > m(2m-1)$. is less than

$$\frac{1}{2} \sum_{0 \le i < m} \frac{m}{N-i} \;.$$

The absolute values of the terms in (1) are decreasing. It follows that:

$$\frac{1}{2} \sum_{0 \le i < m} \frac{m}{N-i} \;<\; p_m \;<\; \sum_{0 \le i < m} \frac{m}{N-i} \,. \tag{2}$$

Take $M = 2m/(1-\alpha)$. Then the edge probability of the random graph $\mathcal{G}(n, p_M)$ is bounded by

$$\frac{1}{1-\alpha} \sum \frac{m}{N-i} \;=\; \frac{1}{2} \sum \frac{M}{N-i} \;<\; p_M,$$

from (2), and also

$$\frac{p_m}{1-\alpha} \;<\; \frac{1}{1-\alpha} \sum \frac{m}{N-i} \,.$$

It follows that: $p_M > p_m/(1-\alpha)$.