Robust Key Establishment in Sensor Networks

Yongge Wang
Dept. of SIS, UNC Charlotte, USA, yonwang@uncc.edu

Abstract

Secure communication guaranteeing reliability, authenticity, and privacy in sensor networks with active adversaries is a challenging research problem since asymmetric key cryptosystems are not suitable for sensor nodes with limited computation and communication capabilities. In most proposed secure communication protocols, sensor nodes need to contact the base station to get a session key first if two sensor nodes want to establish a secure communication channel (e.g., SPINS). In several environments, this may be impractical. In this paper, we study key agreement protocols for which two sensor nodes (who do not necessarily have a shared key from the key predistribution phase) could establish a secure communication channel against active adversaries (e.g., denial of service attacks) without the involvement of the base station.

1 Introduction

A typical sensor network (see, e.g., [7]) consists of nodes, small battery powered devices, that communicate with each other and with more powerful base stations, which in turn is connected to an outside network. The number of base stations and sensors vary according to the purpose of sensor networks. For example, for a national natural event monitoring systems, there could be hundreds of base stations and thousands of or millions of sensors. On the other hand, for a small system such as the SmartDust sensor network [9], there is one or a few base stations and less than one hundred sensor nodes. Generally sensor nodes communicate over a wireless network and broadcast is the fundamental communication primitive. In addition to base stations and normal sensor nodes, some sensor networks could contain special sensor nodes with relatively stronger capabilities. These special sensor nodes could be used for special purposes such as routing or data aggregation and forwarding. These special sensor nodes could even have replaceable batteries. In particular, these special sensor nodes could be built significantly cheap (compared to base stations), thus deployed in a relatively large scale.

Though there could be different sensor nodes, most sensor nodes have limited capabilities. For example, UC Berkeley has designed the SmartDust sensor network [9, 8] with nodes that have a (32 8-bit registers, 4MHz)-CPU, 8KB flash memory, 512 bytes RAM, 512 bytes EEPROM, and 10 Kbps communication bandwidth.

In a sensor network, sensor node could be geographically distributed in a large scale of unattended area. Base stations may not be able to communicate with sensor nodes directly and sensor nodes may not be able to communicate with base station directly. One reason why a sensor node may not be able to communicate with the base station or another sensor node is that substantially less energy is spent to communicate over smaller distances (power consumption is proportional to the square of the distance [5]).

Energy consumption of public key cryptography operations (e.g., RSA encryption and signature schemes or DSA signature scheme) is significantly higher than that for symmetric encryption schemes (e.g., AES) and hash functions (e.g., SHA-1 or SHA-2). Thus symmetric key based encryption schemes and hash functions are often used for establishing secure channels between sensor nodes or between sensor nodes and base stations (see, e.g., [8]).

In order for two nodes to establish a secure channel, a shared secret is required. Classically there are two kinds of schemes for this purpose. The first kind of key distribution protocols is based on infrastructures using trusted third parties (e.g., the Kerberos-style schemes). Modified versions of these schemes have been proposed to distributed sensor networks (e.g., SPINS [8]). For these schemes, each sensor node has a built-in master key, this key is shared between the sensor node and the base station. In order for two sensor nodes to communicate securely, at least one node needs to contact a base station first to get a session key for them to communicate. As pointed out in [5], these schemes could be impractical for large scale distributed sensor networks because of the unknown network topology prior to deployment, communication range limitations, intermittent sensor-node operation, and network dynamics.

The second kind of schemes is based on key predistributions. A simple solution is to install the single master key to each deployed sensor node and to base stations. This single master key solution is not ideal since the capture of any single sensor node will compromise the security of the entire sensor network. A better solution requires to preinstall $n-1$ keys to each sensor node if there are n sensors nodes in the entire network. Then each pair of sensor nodes

could share a unique secret key. This will become impractical when the size of the network increases. In particular, it will be practically impossible to add new nodes to an existing network.

Recently, new key predistribution schemes have been proposed for sensor networks [5, 1]. In the Eschenauer and Gligor [5] random key predistribution scheme, m randomly chosen keys from the key space S is pre-installed into each sensor node before deployment (the exact values of m and S depend on the size and capabilities of the sensor network). After the deployment of the sensor network, the network will perform a key discovery process so that each sensor node could find out other sensor nodes in its wireless communication range for which it shares a secret. When the parameters S and m are chosen properly, they show that the sensor network is sufficiently connected according to the shared key paths. Because of the random choice of keys for each sensor node, a shared key may not exist between some pairs of nodes, in particular, between some pairs of nodes within wireless communication range. If this happens, the two nodes may try to find a path of nodes sharing keys pair-wise and then exchange a key so that they will have a direct link. It should be noted that this key is **known** to each sensor node on the path that is used for exchanging the key. In particular, if any node on the path is controlled by an adversary, then the communication between the two nodes are compromised.

Chan, Perrig, and Song [1] introduced multipath key reinforcement mechanisms to enhance the security of random key predistribution schemes. The multipath key reinforcement scheme in [1] could be used to defeat eavesdropping adversaries. However, an active adversary, who may control messages transmitted from a sensor node, could modify the message sent from u to v on some paths. Thus v will recover an incorrect session key. In another word, the multipath key reinforcement scheme in [1] is not secure (not reliable) against denial of service attacks by an active adversary. Indeed, a sensor node may corrupt without an attacker and send a random message out and v could recover an incorrect session key.

In this paper, we will study protocols to establish shared secret keys between sensor nodes. We are interested in protocols that could be used in the following two scenarios:

1. To establish shared secrets between nodes who do not have a key from the key predistribution phase.

2. In the random key predistribution scheme, the key shared between u and v may reside in the key ring memory of some other nodes in the network. If any of these nodes are captured by the adversary, then the security between u and v is compromised. Our protocol allows nodes to establish a different communication key that the adversary may not be able to recover even if it recovers the secret between u and v.

The protocols in [1] could also be used for the above tasks. The difference is that our protocol is secure against denial of services attacks by active adversaries and is robust against sensor node random crashes, though the protocol in [1] is neither secure against DoS attacks nor robust againt random crashes.

2 Overview and model

In addition to the random key predistribution schemes in [5, 1], we can also use other efficient methods to predistribute keys for sensor nodes. For example, for each area $\mathcal{L}$, a random key space $K_{\mathcal{L}}$ is chosen. When a sensor node is deployed to the area $\mathcal{L}$, m random keys from $K_{\mathcal{L}}$ will be preinstalled to the sensor node. Since flash memory (or ROM) is becoming cheap, sensor nodes may be designed to have larger flash memory and relatively more keys could be predistributed to each sensor.

A distributed sensor network could be modeled by a directed graph. After the sensor network is deployed with key predistribution, key-setup phase could be performed to construct a directed graph $G(V, E)$ (note that a directed graph is a graph $G(V, E)$ where all edges have directions). The vertex set V consists of all sensor nodes and base stations. There are several ways to construct the edge set E. In the following, we list two potential methods.

1. When nodes u and v share a secret key K_{uv} and u can send a message to v directly (i.e., v lies in the wireless transmission range of u), then there is a directed edge from u to v.

2. When nodes u and v share a secret key K_{uv} and u can send a message to v, either directly or via other intermediate nodes (may be bounded by hops number), then there is a directed edge from u to v.

Note that, when there is a directed edge from u to v, there is not necessarily a directed edge from v to u. The sensor network has the choice to choose a specific way to construct the directed graph with the property that if there is a directed edge from node u to node v, then there is a private and authenticated channel from u to v. In another word, secure communication from u to v is guaranteed.

In a directed graph for a sensor network, many nodes are only indirectly connected, as elements of an incomplete network of private and authenticated channels. In other words they need to use intermediate or internal nodes. In particular, a node v may lie in the wireless transmission range of node u but there is no direct edge from u to v and secure communication from u to v is impossible. Thus efficient and secure protocols need to be designed to enable

u and v to communicate securely even if there is no private and authenticated channel from u to v. In [5], the authors suggested the following solutions. When there is no directed edge between nodes u and v, u could find a path from u and v, choose a random key, and send it to v via this path. Obviously, all nodes on this path will learn the value of this key. If the adversary controls any node on this path by chance, then the communication channel from u to v is compromised. A more secure protocol was proposed in [1]: u and v find t node disjoint paths $p_1, \ldots, p_t$ from u to v. u chooses t random values $k_1, \ldots, k_t$ and sends k_i to v via path p_i. Then u and v can use the shared secret $k_1 \oplus \ldots k_t$. The advantage of this protocol is that if the adversary could manage to decrypt messages communicated over $t-1$ paths, it still cannot recover the shared secret between u and v. However, this protocol is not reliable if the adversary could control one path (e.g., it controls the path p_1), and could modify the random value transmitted over that path (e.g., the value of k_1). The goal of this paper is to design efficient protocols for sensor networks so that u and v will establish a secure channel against an active adversary controlling a fixed number of paths between u and v.

Achieving participants cooperation in the presence of faults is a major problem in classical distributed networks. Dolev, Dwork, Waarts, and Yung [4] first designed protocols that achieve private and reliable communication without the need for the parties to start with secret keys. The interplay of network connectivity and secure communication has been studied extensively (see, e.g., [4]). For example, Dolev et al. [4] showed that, in the case of t Byzantine faults, reliable communication is achievable only if the system's network is $2t+1$ connected. They also showed that if all the paths are one way, then $3t+1$ connectivity is necessary and sufficient for reliable and private communications. Desmedt and Wang [3] have initiated the study of secure communications in general networks when there are certain number of directed paths in one direction and another number of directed paths in the other direction. A directed graph constructed from a general sensor network corresponds to the scenarios that have been considered in Desmedt and Wang [3].

In the following, we first introduce some notations. For a directed graph $G(V, E)$ and two nodes $u, v \in V$, throughout this paper, n denotes the number of vertex disjoint paths between the two nodes and t denotes the number of faults under the control of the adversary. We write $|S|$ to denote the number of elements in the set S. We write $x \in_R S$ to indicate that x is chosen with respect to the uniform distribution on S. Let $\mathbf{F}$ be a finite field, and let $a, b, c, s \in \mathbf{F}$. We define $\text{auth}(s; a, b) := as + b$ (see, e.g., [6]). Note that each authentication key $key = (a, b)$ can be used to authenticate one message s without revealing any information about any component of the authentication key.

In a message transmission protocol, the sender u starts with a message s^u drawn from a message space $\mathcal{M}$ with respect to a certain probability distribution. At the end of the protocol, the receiver v outputs a message s^v. We consider a synchronous system in which messages are sent via multicast in rounds. During each round of the protocol, each node receives any messages that were multicast for it at the end of the previous round, flips coins and perform local computations, and then possibly multicasts a message. We will also assume that the message space $\mathcal{M}$ is a subset of a finite field $\mathbf{F}$.

We consider two kinds of adversaries. A passive adversary (or gossiper adversary) is an adversary who can only observe the traffic through t internal sensor nodes (except the message sender and the receiver). An active adversary (or Byzantine adversary) is an adversary with unlimited computational power who can control t internal nodes. That is, an active adversary will not only listen to the traffics through the controlled nodes, but also control the message sent by those controlled nodes. Both kinds of adversaries are assumed to know the complete protocol specification, message space, and the complete structure of the graph.

For any execution of the protocol, let adv be the adversary's view of the entire protocol. We write $adv(s, r)$ to denote the adversary's view when $s^u = s$ and when the sequence of coin flips used by the adversary is r.

Definition 2.1 *(Franklin and Wright [6])*

1. *Let $\delta < \frac{1}{2}$. A message transmission protocol is* δ-reliable *if, with probability at least $1-\delta$, v terminates with $s^v = s^u$. The probability is over the choices of s^u and the coin flips of all nodes. A message transmission protocol is* reliable *if it is* 0-reliable.

2. *A message transmission protocol is* ε-private *if, for every two messages s_0, s_1 and for every r,*

$$\sum_c |\Pr[adv(s_0, r) = c] - \Pr[adv(s_1, r) = c]| \leq 2\varepsilon.$$

   The probabilities are taken over the coin flips of the honest parties, and the sum is over all possible values of the adversary's view. A message transmission protocol is perfectly private *if it is* 0-private.

3. *A message transmission protocol is* (ε, δ)-secure *if it is ε-private and δ-reliable.*

For two nodes u and v in a directed graph such that there are $2t+1$ node disjoint paths from u to v, there is a straightforward reliable message transmission from u to v against

a t-active adversary: u sends the message s to v via all the $2t+1$ paths, and v recovers the message s by a majority vote.

3 Secure message transmission in sensor networks

For classical distributed networks, Dolev, Dwork, Waarts, and Yung [4] showed that if all channels from node u to node v are one-way, then $(3t+1)$-connectivity is necessary and sufficient for (0,0)-secure message transmissions from u to v against a t-active adversary. They also showed that if all channels between u and v are two-way, then $(2t+1)$-connectivity is necessary and sufficient for (0,0)-secure message transmissions between u and v against a t-active adversary. Desmedt and Wang [3] showed that the necessary and sufficient condition for $(0,\delta)$-secure message transmission from node u to node v against a t-active adversary is that there are at least $t+1$ node disjoint paths from u to v and there are at least $2t+1$ node disjoint paths in total from u to v and from v to u. Desmedt and Wang [3] also showed that if there are l directed node disjoint paths from v to u, then a necessary and sufficient condition for $(0,0)$-secure message transmission from u to v against a t-active adversary is that there are $\max\{3t+1-2l, 2t+1\}$ directed node disjoint paths from u to v.

Before we describe our protocols, we first define a primitive: "u sends a value x to v via a path p". Assume that the path p consist of the edges $u \rightarrow u_1$, ..., $u_j \rightarrow v$. Then the meaning of the above primitive is that u sends x to u_1 via the secure channel $u \rightarrow u_1$ (i.e., encrypted and authenticated), u_1 forwards x to u_2 via the secure channel $u_1 \rightarrow u_2$, ..., and u_n forwards x to v via the secure channel $u_n \rightarrow v$.

Assume that it is sufficient to $(0,\delta)$-securely establish a secure channel from u to v and at most one sensor node between u and v is controlled by the active adversary. Let p_1 and p_2 be two node disjoint paths from u to v, and q be the node disjoint path from v to u. In the following protocol from [3], u securely sends a random key $s^u \in_R \mathbf{F}$ to v. The protocol is $(0,\delta)$-secure against an active adversary that controls at most one path. The goal of the protocol is that no matter which node the adversary controls, she learns nothing about this secret and v gets this secret with high probability.

PROTOCOL I

Step 1 u chooses $s_0^u \in_R \mathbf{F}$, $(a_0^u, b_0^u), (a_1^u, b_1^u) \in_R \mathbf{F}^2$, and let $s_1^u = s^u - s_0^u$. For each $i \in \{0,1\}$, u sends $(s_i^u, (a_i^u, b_i^u), \text{auth}(s_i^u; a_{1-i}^u, b_{1-i}^u))$ to v via path p_i.

Step 2 Assumes that v receives $(s_i^v, (a_i^v, b_i^v), c_i^v)$ via path p_i. v checks whether $c_i^v = \text{auth}(s_i^v; a_{1-i}^v, b_{1-i}^v)$ for $i = 0$ and $i = 1$. If both equations hold, then v knows that with high probability the adversary was either passive or not on the paths from u to v. v can recover the secret $s^v = s_0^v + s_1^v$, sends "OK" to u via the path q, and terminates the protocol. Otherwise, one of the equations does not hold and v knows that the adversary was on one of the paths from u to v. In this case, v chooses $(a^v, b^v) \in_R \mathbf{F}^2$, and sends $((a^v, b^v), (s_0^v, (a_0^v, b_0^v), c_0^v), (s_1^v, (a_1^v, b_1^v), c_1^v))$ to u via the path q.

Step 3 If u receives "OK", then u terminates the protocol. Otherwise, from the information u received via path q, u decides which path from u to v is corrupted and recover v's authentication key (a^u, b^u). u sends $(s^u, \text{auth}(s^u; a^u, b^u))$ to v via the non-corrupted path from u to v.

Step 4 v recovers the secret s^v and checks that the authenticator is correct.

Desmedt and Wang [3] show that the above protocol is $(0,\delta)$-secure against an adversary that controls at most one sensor node between u and v.

For certain environments, it is necessary to design protocols which are $(0,\delta)$-secure against an active adversary who may control more than one node between u and v. According to the results in [3], if there are at least $t+1$ node disjoint paths from u to v and there are at least $2t+1$ node disjoint paths in total from u to v and from v to u then there is an efficient $(0,\delta)$-secure message transmission from node u to node v against a t-active adversary.

In some environments, $(0,0)$-security may be desired. Indeed, if u and v are sufficiently connected, then a $(0,0)$-secure protocol could be designed for u to send a secret key to v with perfect privacy and with perfect reliability. According to the results in Desmedt and Wang [3], we have the following facts. If there are $n = \max\{3t+1-2l, 2t+1\}$ directed node disjoint paths $p_1, \ldots, p_n$ from u to v and l directed path $q_1, \ldots, q_l$ from v to u ($q_1, \ldots, q_l$ are node disjoint from $p_1, \ldots, p_n$) then there is an efficient $(0,0)$-secure message transmission protocol from u to v against a t-active adversary. Thus if we want to tolerate more powerful adversaries, we need to find more paths between u and v to design efficient and robust protocols for establishing a secure channel from u to v.

4 Sensor networks and radio networks

In Section 3, we used directed graphs to model sensor networks with key predistribution. Though the directed graph model is sufficient for most applications, some special cases are not included in this model. For example, in

the random key predistribution scheme [5, 1], it is possible that two edges $u_1 \rightarrow v_1$ and $u_2 \rightarrow v_2$ correspond to the same key. In another word, the key shared by sensor nodes u_1 and v_1 is the same as the key shared by sensor nodes u_2 and v_2. If this happens, the key agreement protocol in Section 3 using node disjoint paths may not provide secure solutions. For example, assume that p_1 and p_2 are node disjoint directed paths from u to v in the **PROTOCOL I** of Section 3, and there are nodes w_1, w_2, w_3, w_4 such that

1. w_1 and w_2 are on p_1, w_3 and w_4 are on p_2;
2. There is a directed edge $w_1 \rightarrow w_2$ corresponding to a key k_0;
3. There is a directed edge $w_3 \rightarrow w_4$ corresponding to a key k_0;
4. w_4 is within the range of w_1 or w_2 is within the range of w_3.

Then all messages from u to v could be decrypted by the node w_2 or w_4. Thus the **PROTOCOL I** in Section 3 is not $(0, \delta)$-secure against a 1-active adversary.

As mentioned in [2], radio networks could be used to model this kind of scenarios. A *radio network* is a *directed colored-edge multigraph* $R(V, E, F, c)$, where V is the node set (in our case, corresponding to sensor nodes), E is the directed edge set (there might be more than one directed edge from one node to another one), F is the color set, and c is a map from E to F (the map c assigns a color to each edge).

After the sensor network is deployed with key predistribution (e.g., random key predistribution [5, 1]), key-setup phase could be performed to construct a radio network $R(V, E, F, c)$. The vertex set V consists of all sensor nodes and base stations, and the color set F consists of key identifications. The edge set E and the mapping c could be defined as follows.

- When nodes u and v share a secret key K_{uv} with key identification id_{key}, and u can send a message to v directly (i.e., v lies in the wireless transmission range of u), then draw a directed edge from u to v, and let $c(u \rightarrow v) = id_{key}$.

Necessary and sufficient conditions for secure communication in general radio networks have been studied in [2]. In this paper, we are mainly interested in the applications to sensor networks. In the following, we will analyze the conditions to be added to the protocol in Section 3 so that they remain secure.

For two paths p_1 and p_2 between u and v, we say that p_1 and p_2 share a sink if there exist nodes w_1, w_2, w_3, w_4 such that

1. w_1 and w_2 lie on p_1, w_3 and w_4 lies on p_2;
2. $c(w_1 \rightarrow w_2) = c(w_3 \rightarrow w_4)$.
3. w_4 is in the communication range of w_1 or w_2 is in the communication range of w_3 .

Let $R(V, E, F, c)$ be a radio network and $u, v \in V$. If there are two node disjoint directed paths p_1 and p_2 from u to v and one directed path q (node disjoint from p_1, p_2) from v to u such that no pair from the paths p_1, p_2, q shares a common sink, then the message transmission protocol **PROTOCOL I** in Section 3 is $(0, \delta)$-secure against a 1-active adversary. However it is straightforward to show that this condition is not necessary. For example, if only p_1 and p_2 share a common sink and the sink is not located on either p_1 or p_2, then the following protocol is a $(0, \delta)$-secure (against a 1-active adversary) key agreement protocol between u and v.

PROTOCOL II

Step 1 In this step, u sends $s^u \in_R \mathbf{F}$ and $\bar{s}^u \in_R \mathbf{F}$ to v:

1. u chooses $s_0^u \in_R \mathbf{F}$, $(a_0^u, b_0^u), (a_1^u, b_1^u) \in_R \mathbf{F}^2$, and let $s_1^u = s^u - s_0^u$. For each $i \in \{0, 1\}$, u sends $(s_i^u, (a_i^u, b_i^u), \text{auth}(s_i^u; a_{1-i}^u, b_{1-i}^u))$ to v via path p_i.
2. u chooses $\bar{s}_0^u \in_R \mathbf{F}$, $(\bar{a}_0^u, \bar{b}_0^u), (\bar{a}_1^u, \bar{b}_1^u) \in_R \mathbf{F}^2$, and let $\bar{s}_1^u = \bar{s}^u - \bar{s}_0^u$. For each $i \in \{0, 1\}$, u sends $(\bar{s}_i^u, (\bar{a}_i^u, \bar{b}_i^u), \text{auth}(\bar{s}_i^u; \bar{a}_{1-i}^u, \bar{b}_{1-i}^u))$ to v via path p_i.

Step 2 Assumes that v receives $(s_i^v, (a_i^v, b_i^v), c_i^v)$ and $(\bar{s}_i^v, (\bar{a}_i^v, \bar{b}_i^v), \bar{c}_i^v)$ via path p_i. v checks whether $c_i^v = \text{auth}(s_i^v; a_{1-i}^v, b_{1-i}^v)$ and $\bar{c}_i^v = \text{auth}(\bar{s}_i^v; \bar{a}_{1-i}^v, \bar{b}_{1-i}^v)$. v distinguishes the following two cases:

1. All four equations hold. v knows that with high probability the adversary was either passive or not on the paths from u to v. v recovers both s^v and $\bar{s}^v$. Note that if the adversary controls the common sink of p_1 and p_2, then the adversary may recover these two values also. v chooses $r^v \in_R \mathbf{F}$, and sends ("OK", r^v, auth$(r^v; s^v, \bar{s}^v)$) to u via the path q.
2. At least one of the equations does not hold and v knows that the adversary was on one of the paths from u to v. In this case, v chooses $r^v \in_R \mathbf{F}^2$, and sends ("KEY", r^v, $(s_0^v, (a_0^v, b_0^v), c_0^v), (s_1^v, (a_1^v, b_1^v), c_1^v)$,

$(\bar{s}_0^v, (\bar{a}_0^v, \bar{b}_0^v), \bar{c}_0^v), (\bar{s}_1^v, (\bar{a}_1^v, \bar{b}_1^v), \bar{c}_1^v))$ to u via the path q. v sets the agreed key as r^v and terminates the protocol.

Step 3 u distinguishes the following three cases:

1. u receives ("OK", r^u, d^u) from v via the path q. u checks whether $d^u = \text{auth}(r^u; s^u, \bar{s}^u)$. If the equation holds, u knows that with high probability, v has recovered the correct values of s^u and s^v, and r^u was the value that v sent via the path q. u sends "KEY is SUM" via p_1 and p_2 to v. u sets the agreed key as $s^u + \bar{s}^u + r^u$ and terminates the protocol. Otherwise, the equation does not hold and u knows that the path q is corrupted. u sends "OK" to v via both paths p_1 and p_2. u sets the agreed key as $s^u + \bar{s}^u$ and terminates the protocol.
2. u receives ("KEY", r^u, $(\hat{s}_0^u, (\hat{a}_0^u, \hat{b}_0^u), \hat{c}_0^u), (\hat{s}_1^u, (\hat{a}_1^u, \hat{b}_1^u), \hat{c}_1^u)$, $(\hat{\bar{s}}_0^u, (\hat{\bar{a}}_0^u, \hat{\bar{b}}_0^u), \hat{\bar{c}}_0^u), (\hat{\bar{s}}_1^u, (\hat{\bar{a}}_1^u, \hat{\bar{b}}_1^u), \hat{\bar{c}}_1^u))$ from v via the path q. u further distinguishes the following two cases:
   - 2.a) For at least one of the $i = 0, 1$, $(\hat{s}_i^u, (\hat{a}_i^u, \hat{b}_i^u), \hat{c}_i^u) = (s_i^u, (a_i^u, b_i^u), \text{auth}(s_i^u; a_{1-i}^u, b_{1-i}^u))$ and $(\hat{\bar{s}}_i^u, (\hat{\bar{a}}_i^u, \hat{\bar{b}}_i^u), \hat{\bar{c}}_i^u) = (\bar{s}_i^u, (\bar{a}_i^u, \bar{b}_i^u), \text{auth}(\bar{s}_i^u; \bar{a}_{1-i}^u, \bar{b}_{1-i}^u))$. In this case, u determines that with high probability, the path p_{1-i} is corrupted. u sets the agreed key as r^u and terminates the protocol.
   - 2.b) For both $i = 0$ and $i = 1$, the equations do not hold. In this case, u determines that the path q is corrupted. u sends "OK" to v via both paths p_1 and p_2. u sets the agreed key as $s^u + \bar{s}^u$ and terminates the protocol.
3. u receives anything else. u sends "OK" to v via both paths p_1 and p_2. u sets the agreed key as $s^u + \bar{s}^u$ and terminates the protocol. Note that in this case, u discovers that the path q is corrupted.

Step 4 v distinguishes the following two cases (note that v continues to this step only if v recovers the correct values of s^v and $\bar{s}^v$ in Step 2):

1. v receives "KEY is SUM" on either path p_1 or p_2. v sets the agreed key as $s^v + \bar{s}^v + r^v$ and terminates the protocol. Note that in this case, u recovered the correct value of r^u.
2. v receives "OK" from both paths p_1 and p_2. v sets the agreed key as $s^v + \bar{s}^v$ and terminates the protocol. Note that in this case, u discovers that the path q is corrupted.

It is straightforward to show that the above protocol is $(0, \delta)$-secure against a 1-active adversary. In another word, at the end of the protocol, u and v agree on a secret key with high probability and the adversary gains no information about the secret key.

References

[1] H. Chan, A. Perrig, and D. Song. Random key predistribution for sensor networks. In: *Proceedings of IEEE Conference on Security and Privacy*. 2003.

[2] Y. Desmedt, R. Safavi-Naini, H. Wang, and Y. Wang. Radio networks with reliable communications. To appear.

[3] Y. Desmedt and Y. Wang. Perfectly secure message transmission revisited. In: *Proc. Eurocrypt '02*, pages 502–517,LNCS 2332, Springer-Verlag, 2002.

[4] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *J. of the ACM*, **40**(1):17–47, 1993.

[5] 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, 2002.

[6] M. Franklin and R. Wright. Secure communication in minimal connectivity models. *Journal of Cryptology*, **13**(1):9–30, 2000.

[7] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. To appear in: *Proc. 1st Int. Workshop on Sensor Network Protocols and Applications*, 2003.

[8] A. Perrig, R. Szewczyk, J. Tygar, V. Wen, and D. Culler. SPINS: Security protocols for sensor networks. *Wireless Networks* **8**:521–534, 2002.

[9] K. Pister, J. Kahn, and B. Boser. SmartDust: wireless networks of millimeter-scale sensor nodes, 1999.

[10] Y. Wang and Y. Desmedt. Secure communication in multicast channels: the answer to Franklin and Wright's question. *J. of Cryptology*, **14**(2):121–135, 2001.