Sensors **2010**, *10*, 4410-4429; doi:10.3390/s100504410

OPEN ACCESS

sensors

ISSN 1424-8220

www.mdpi.com/journal/sensors

Article

Untraceable Mobile Node Authentication in WSN

Kyusuk Han [1], Kwangjo Kim [1] and Taeshik Shon [2,⋆]

[1] KAIST, 119 Munjiro Yuseonggu, Daejeon, Korea

[2] SAMSUNG Electronics CO., LTD, Suwon, Korea

⋆ Author to whom correspondence should be addressed; E-Mail: ts.shon@samsung.com; Tel.: +82-31-279-5402; Fax: +82-31-279-5255

Received: 28 February 2010; in revised form: 20 March 2010 / Accepted: 5 April 2010 / Published: 30 April 2010

Abstract: Mobility of sensor node in Wireless Sensor Networks (WSN) brings security issues such as re-authentication and tracing the node movement. However, current security researches on WSN are insufficient to support such environments since their designs only considered the static environments. In this paper, we propose the efficient node authentication and key exchange protocol that reduces the overhead in node re-authentication and also provides untraceability of mobile nodes. Compared with previous protocols, our protocol has only a third of communication and computational overhead. We expect our protocol to be the efficient solution that increases the lifetime of sensor network.

Keywords: wireless sensor networks; authentication; mobile node; untraceability; key distribution

1. Introduction

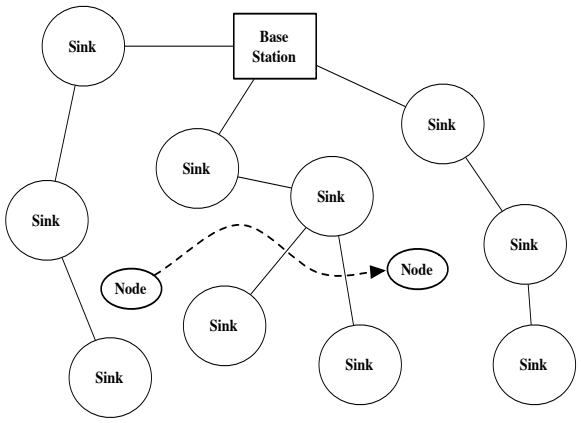
Wireless Sensor Network (WSN) is the network that consists of light-weight battery-powered devices with short-ranged wireless communication function. The devices have sensors that gather the environmental information. After sensing the information, the devices send the information to the networks. We define such devices as sensor node, and the core parts of the network as sinks and the base station (Figure 1).

Authenticated key distribution in WSN is one of the fundamental security problems. Employing the security protocols of other computer networks to WSN is insufficient because the light-weight devices

have limited resources. Thus, the most important issues in security researches on WSN are the design of resource-efficient security protocol. Several approaches such as key pre-distribution, pairwise key agreement, group key based key agreement and hierarchical key management schemes were introduced for the efficient authenticated key distribution.

Zigbee [1] specifies the key pre-distribution method that stores the master secret between two entities for commercial application that also requires the large key storage management in scalable network. The pairwise key agreement protocols based on the random key pre-distribution that enables to share the pairwise key from the pre-distributed key pool are proposed in [2–4]. For the group key based key agreement, Zhu *et al.* [5] showed the efficient key distribution model with cluster key that enables the reduced overhead of the base station. Recently, the hierarchical key management schemes, in which the sensor nodes establish the hierarchy for the key distribution, are proposed by [6,7].

Figure 1. A dynamic mobile node continuously moves in the sensor networks that the static sinks established. The unbroken line denotes the static connection between sinks and the base station. The dotted line denotes the movement of the mobile node.



However, since the above authenticated key management protocols only considered static environments, they are not sufficient to be applied to the advanced WSN with the mobile nodes. For example, Wireless Sensor and Actor Network (WSAN) brings the concept of mobility as the extension of WSN [8,9] . It is obvious that the wireless sensor network will be the combined network of static sensor network and the mobile sensor and actor networks. In such environments, handling a large overhead from frequent node re-authentication requests due to the continuous node movements and the threats of tracing the node movement are important security issues. Thus, efficient re-authentication and untraceability are important security requirements in WSN with mobile nodes. Although Fantacci *et al.* [10] studied the possible presence of mobile node and proposed the authentication protocol supporting node mobility that does not require any sink or base station for authentication and key distribution, their model still incurs large communication overhead in node re-authentication.

Therefore, our motivation is to propose an efficient node re-authentication and key distribution model that reduces communication and computational overhead for node re-authentication. After claiming the security issues in WSN with mobile nodes, we present the insufficiency of current authentication and key distribution schemes to such environments. We then propose an efficient untraceable re-authentication

and key distribution protocol that can reduce the communication overhead between a sink and the base station. Applying our protocol, a node previously authenticated by a sink can be efficiently re-authenticated with less communication and computational overhead when the node changed position and the node movement stays untraceable.

The rest of this paper is organized as follows: Section 2 briefly presents the drawbacks of previous authentication and key distribution protocols supporting mobility in WSN and identifies the security requirements. Then, We propose the efficient mobile node re-authentication protocol in Section 3, and analyze the performance and security of our protocol in Section 4. Finally, Section 5 concludes this paper.

2. Issues of Mobile Node Authentication in WSN

In this section, we present the security problems on node mobility in WSN and the limits of previous authentication and key agreement models. At first, we show a sensor network model with mobile nodes as in Figure 1. We define a static sensor node as Sink, a mobile node as Node, and the base station that is the core network. The node has linear movements in the network. The base station and sinks are static, which is the same as in Ibriq and Mahgoub's model [7]. Sinks act as the gateway and link nodes to the base station, and the base station is a kind of headquarter that manages the entire networks. When a node initially joins the network, the node connects to a sink in the network and is authenticated by the sink with the help of the base station. Afterwards, the node moves and reconnects to other sink. We assume that the sink that re-authenticates the node is the neighbor sink of the sink that previously authenticated the node. The re-authentication processes frequently happen because the node continuously moves in the network.

In practical scenarios, re-authentication happens when a node lost connection to the sink or moved and connected to other sink. For the former case, the node can be easily re-authenticated to the same sink when the connection becomes available again. For the latter case, the node request the re-authentication to other sink that is closest to the previously attached sink.

2.1. Previous Works on the Authenticated Key Agreement in WSN

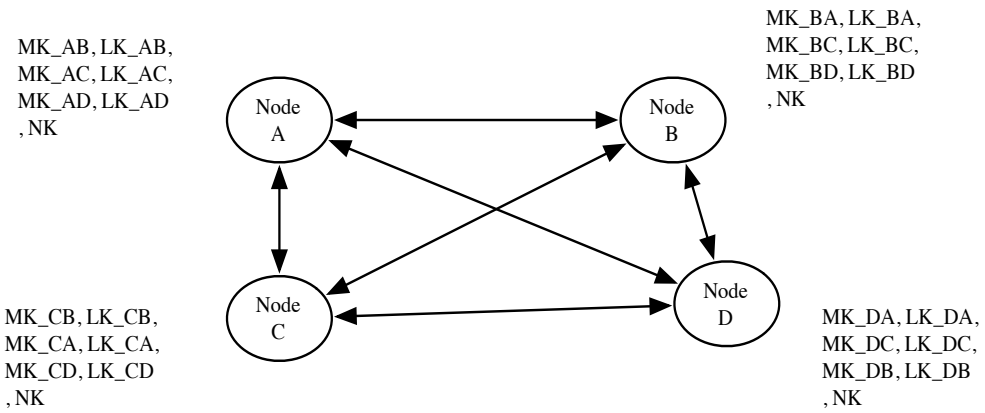
Currently, most researches on the authentication and key distribution assume WSN as a static environments. Thus, they only focused on the efficient initial authentication and key setup.

Commercially deployed Zigbee [1] specifies the key agreement architecture that pre-distribute keys. In their architecture, each node pre-installs their unique keys, such as the master key (MK) and the link key (LK), that are shared to other entities and the network key (NK) is shared to entire network by the manufacturer. In order to support node mobility using the unique key, each node has to contain the key as well as the number of nodes. Figure 2 shows the required keys in Zigbee. Seven keys (three MKs, three LKs, and a NK) were required for the secure communication in the network with only four nodes. Thus, deploying Zigbee in the large scale networks requires quite large storage for the key management.

In 2002, Eschenauer and Gligor [2] proposed the pairwise key agreement protocols based on the random key pre-distribution that enables sharing the pairwise key from the pre-distributed key pool. In the initial stage, each node stores m numbers of keys selected in a key pool. After the nodes are

deployed, each node shares the key information to its neighbor nodes. When the shared keys are found, the node establishes the secure links between sinks that share the keys. After the links are established, nodes generate the pairwise key with the sink that has no shared information via the secure link. Later, Chan *et al.* [3] improved the model by generating the pairwise key from multiple numbers of shared key, and Liu and Ning [11] proposed a model in which the pairwise key is not directly distributed but derived by a bivariate polynomial. However, the networks cannot be completely connected by probabilistic methods. The probability of failure increases in the case of irregular deployment of sensor nodes or unpredictable interruptions.

Figure 2. Each node has to store seven keys in order to support mobile nodes in the network with four sensor nodes under Zigbee. [1]



Zhu *et al.* [5] introduced the group key based key agreement model that minimized threats of compromised nodes. Every node has a unique key, pairwise keys with neighbor nodes, a cluster key shared with all neighbor nodes, and the global key shared with the entire network. However, they only assumed static networks.

In 2006, Abraham and Ramanatha [6] proposed an authentication and initial shared key establishment model in hierarchical clustered networks. In 2006, Ibriq and Mahgoub [7] proposed an efficient hierarchical key establishment model with "partial key escrow table". Using the key escrow table, a sink can self-generate the shared key for the attached nodes. Figure 3 shows the brief model of [7]. However, any sinks have to maintain the information of every node in the table to support the node mobility.

Fantacci *et al.* [10] proposed the distributed node authentication model that does not require the base station as the centralized authenticator. Figure 4 shows the brief model with no centralized authenticator. Every node shares the partial authentication information of each node based on Shamir's Secret Sharing Scheme [12], which enables node mobility support. When a node requests to be authenticated to other node, the Node 2 is the authenticator, while other nodes such as Node 5 and Node 6 are distributed authentication servers. However, the issue in this model is the overhead on each node. Since the node has to participate in the authentication procedures as authenticator or an authentication server, the computational and communication overhead can increase significantly with frequent authentication requests.

Huang *et al.* [13] proposed self-organizing algorithm by using Elliptic Curve Cryptography (ECC). Once the certificates are issued to nodes, nodes can self-establish the pairwise key by exchanging the certificates with any node. Even though the public key based security architecture requires more advanced computational power and resources, efficient applications for the sensor networks will be available in near future with light weight implementation such as TinkPK [14] and TinyECC [15].

Figure 3. Ibriq and Mahgoub' model [7]: The intermediate Sink 1 stores the partial key escrow table that stores the partial information of nodes. After the requests from nodes are received, Sink 1 request the authentication ticket to the base station. After receiving the ticket, Sink 1 authenticates and share keys with nodes.

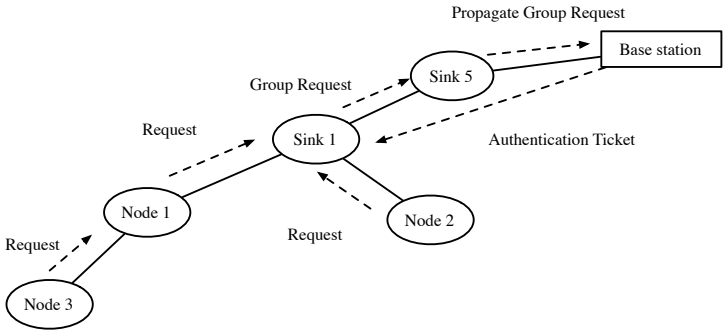
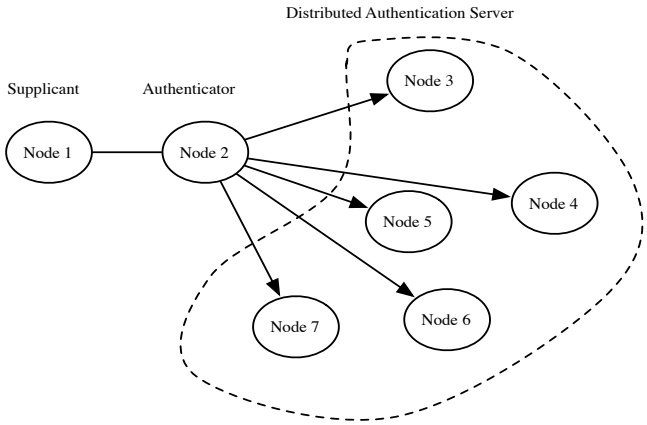


Figure 4. Fantacci *et al.*'s model [10]: When Node 1 request to join the network, Node 2 acts as the authenticator. Other nodes act as authentication server. In the initial setup of network, all node share the partial information of each node. When a node request to be authenticated, they gather the authentication information using secret sharing.



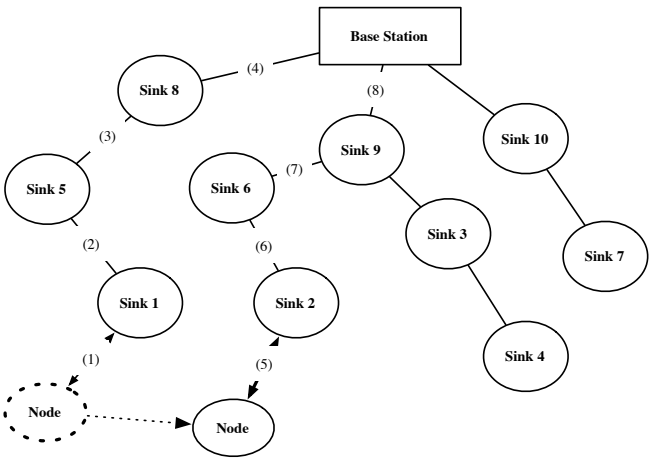
2.2. Drawbacks of Previous Protocols Supporting Mobile Node

2.2.1. Frequent Re-authentication

Since the sensor has battery of limited power and low-end processor with short-range wireless communication, reducing communication and computational overheads is important to increase the lifetime of the sensor. However, the mobile sensor node may incur large overhead for security computation due to the frequent requests of node re-authentication. When a node connects to a sink, the sink has to authenticates the node. Afterwards, the node will connect to another sink after the movement, and the new sink has to authenticate the node again. If the node moves continuously, the authentication process will also occur repeatedly. It is obvious that the frequent re-authentication processes significantly drain the resources in battery-based sensor nodes.

Current authentication and key distribution protocols lacks the consideration of node mobility and are thus insufficient to be applied in such environment. Using the current protocols such as [7], the communication pass (1)-(2)-(3)-(4) is required for the initial authentication and key distribution in Figure 5. When the node moves and reconnects to sink 2, the communication pass (5)-(6)-(7)-(8) is required for authentication and key distribution, which have the similar communication overhead to the initial authentication. Such overhead will create huge problem in the environment where large numbers of nodes moves frequently. Thus, the reduction of computational and communication overheads in re-authentication are very urgent requirement for the node mobility support in the WSN.

Figure 5. Communication pass: initial authentication (1)-(2)-(3)-(4), re-authentication (5)-(6)-(7)-(8). The unbroken line denotes the static connection, and the dotted line denotes the movement of the node.



2.2.2. Tracing Node Movements

Considering the mobility of sensor nodes, the tracking of node movement is one of the possible attacks. For example, when the mobile nodes are deployed in battle fields, the tracking by enemies is of significant threat to the networks. Also, tracking node movement threats privacy. Thus, the

authentication and key agreement protocols should provide the privacy of the mobile node. Current protocols do not consider the mobility of the node.

2.3. Security and Privacy Requirements

We define the security requirements as follows. We assume that when the node N communicates with a sink S_2 after disconnection to the sink S_1, S_1 cannot receive any message between N and S_2. S_2 is one of neighbor sinks of S_1.

Re-authentication An authenticated node N and S_2 should be able to identify each other with less communication and computational overhead than in the initial authentication.

Untraceability In re-authentication of N, S_2 only identifies that N was previously connected to S_1, and never traces the direction of N.

In addition to the requirements of "re-authentication" and "untraceability", we also define the fundamental security requirements as follows.

Confidentiality When N and S_1 are operating initial authentication, nobody can know the communication packet between N and S_1, between S_1 and BS. For re-authentication between N and S_2, nobody except S_1 can know the communication information, while S_1 out of communication range.

Message Authentication Any malicious adversaries should not be able to forge the communication packet.

Key Freshness N and S should be able to verify that the key is generated during the current session.

Node/Sink Resiliency Even N, S_1 or S_2 are compromised by a malicious adversary, they should not be able to affect to the entire network.

"Confidentiality", "message authentication", and "key freshness" are important requirements to protect against the attacks such as the replay attack or man-in-the-middle attack. "Node/Sink resiliency" is a practical threat as the sensor nodes are generally deployed in the environment out of administration.

3. Proposed Protocol

In this section, we propose our novel authentication and key distribution scheme that provides efficient mobile node re-authentication and untraceablity. In Section 3.1, we briefly overview the overall process of proposed protocol. In Section 3.2, we introduce the concept of "authentication ticket" that enables fast re-authentication. After that, we show our efficient node re-authentication protocol in Section 3.3.

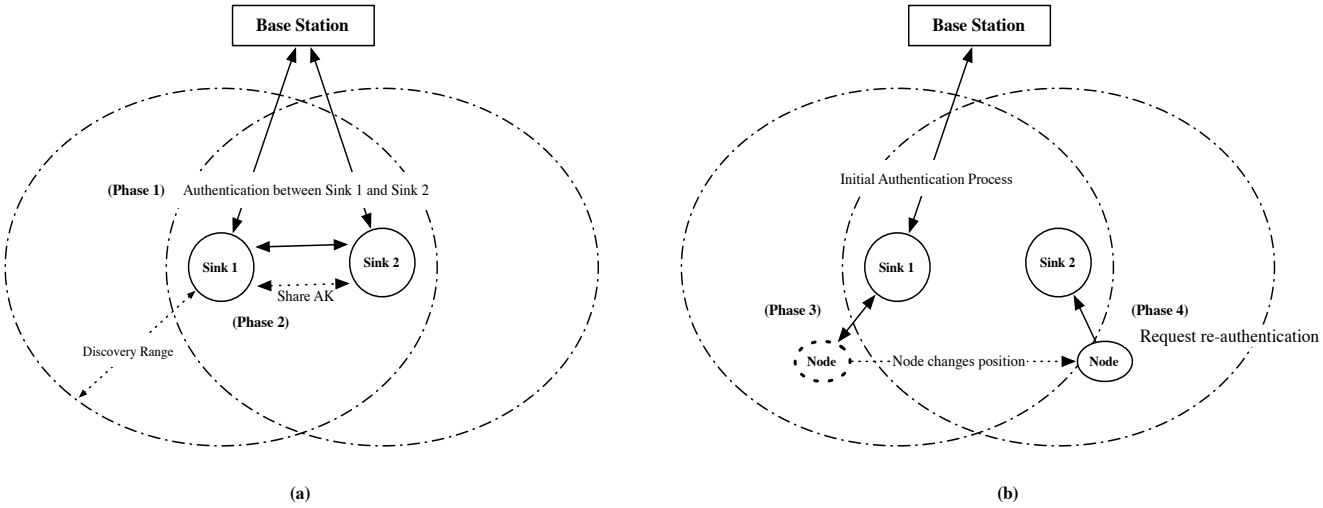
3.1. Overview of Proposed Protocol

We briefly describe the procedure of our proposed protocol in Figure 6. Assume that there are a base station BS, a sink S_1, a neighbor sink S_2, and a mobile node N in the network. We define the neighbor sink as the sink that is in the 1 hop communication range. S_1 periodically broadcasts HELLO

in Phase 0. When S_2 receives HELLO, S_2 initiates the neighbor relationship if S_1 is a newly discovered sink. After the pairwise key between S_1 and S_2 has been exchanged in Phase 1, S_1 and S_2 exchange the authentication key that is used to verify the authenticated user in Phase 2. Phase 1 and Phase 2 are only required during establishing the static sensor network. We let the establishment of the static sensor network follow any previous protocol, such as [7].

When N first joins the network, N may be connected to S_1 in the network, as in Figure 6. After receiving HELLO of S_1, N initiates the initial authentication with S_1 in Phase 3. After N is authenticated S_1, N only needs the re-authentication in Phase 4 when N continuously moves and request the authentication again. The authentication process in Phase 3 is only necessary when the re-authentication fails in certain case, e.g., when the neighbor sink is not available.

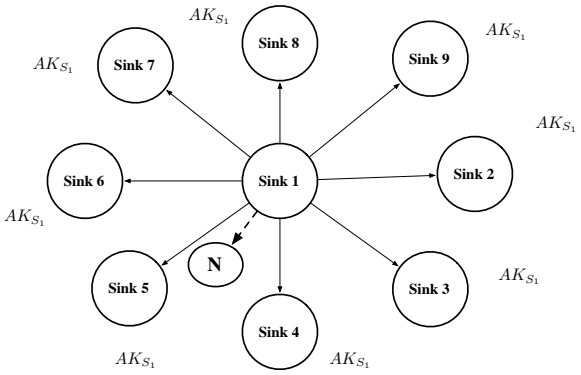
Figure 6. Protocol overview: Upon receiving HELLO of Sink 2 (S_2), (a) Sink 1 (S_1) mutually authenticates Sink 2 (Phase 1), and shares the authentication key (Phase 2). (b) Node is initially authenticated by Sink 1 (Phase 3), and requests re-authentication to Sink 2.



3.2. Authentication Ticket

The "Authentication Ticket" is used for the node re-authentication. When a node requests authentication to a sink, the sink generates the authentication ticket and sends it to the node. The authentication ticket can be verified by the authentication key that is given to the neighbor sinks. Using the authentication ticket, the node movement is untraceable. Verification of the authentication ticket is available to neighbor sinks of the sink that issued the ticket. We adopt the idea of "cluster key" in [16] that shared to neighbor sinks. The main difference is that the cluster key in [16] is used for broadcast communication in the cluster, while the key in our protocol is used for verifying the authentication ticket. Thus, we rename the key as "authentication key" because of its different use in the protocol. Figure 7 shows that neighbor sinks of Sink 1 (S_1) shares the authentication key AK_{S_1}.

Figure 7. Sink 1 shares AK_{S_1} to neighbor sinks. When N is authenticated by Sink 1, any neighbor sinks can re-authenticate N.



3.3. Protocol Description

The protocol consists of five phases as follows: **Phase 0** The common neighbor discovery, **Phase 1** Neighbor sink relationship set up, **Phase 2** Neighbor group authentication key share, **Phase 3** Initial node authentication, and **Phase 4** Node re-authentication.

The notations used in the protocol are defined in Table 1. Key IK_N is the integrity key derived from K_N, where $IK_N = KDF(K_N)$. KDF is an one-way key derivation function. We can also use a hash function for KDF.

Table 1. Notation

Term	Description	Term	Description
BS	Base Station	$E_t\{m\}$	Encrypt arbitrary message m using t
$h\{m\}$	Hash arbitrary message m	$MAC_t(m)$	Message Authentication Code using t
TS	Time stamp	K_N	Pre-shared key between N and BS
IK_N	IK derived from K_N	K_S	Pre-shared key between S and BS
IK_S	IK derived from K_S	SK	Shared session key between sinks
SIK	IK derived from SK	AK_S	Group Authentication Key of Sink
AIK_S	IK derived from AK_S	NK	Shared session key between S and N
NIK	IK derived from NK	IK	Integrity Key

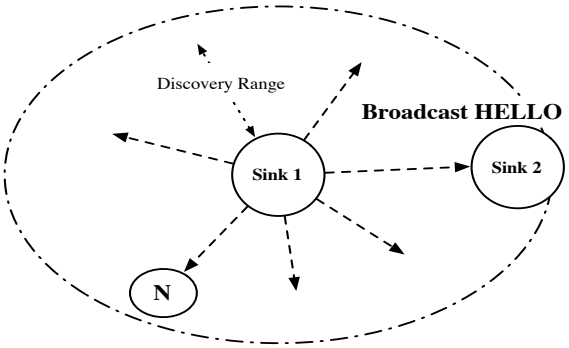
3.3.1. Phase 0: Neighbor Discovery

A sink S_1 periodically generates a random nonce R_0 . S_1 also generates $u_0 = E_{K_{S_1}}\{R_0||TS_0\}$ and $v_0 = MAC_{IK_{S_1}}(S_1||\text{HELLO}||u_0)$, where TS_0 is time stamp. u_0 and v_0 are included in the HELLO message as in Figure 8. Then S_1 broadcasts u_0 and v_0 as follows:

$$S_1 \rightarrow Broadcast : S_1||HELLO||u_0||v_0$$

Phase 0 is the periodical common procedure. When a sink receives HELLO, the sink initiates Phase 1 or Phase 2. When a node receives HELLO, the node initiates Phase 3 or Phase 4.

Figure 8. Neighbor discovery (Phase 0): sink periodically broadcasts HELLO.



3.3.2. Phase 1: Neighbor Sink Relationship Set Up

Assume another sink S_2 receives HELLO message. S_2 checks whether the sender of HELLO S_1 is known or not. If S_2 already knows S_1, S_2 discards the message. Otherwise, S_2 requests to set up the neighbor relationship as follows:

P-1.a. S_2 randomly selects R_1 and generates $u_1 = E_{K_{S_2}}\{R_1||u_0\}$, $v_1 = MAC_{IK_{S_2}}(S_2||\text{BS}||S_1||u_1||v_0)$.

$$S_2 \rightarrow \text{BS} : S_2||\text{BS}||S_1||u_1||v_1||v_0$$

P-1.b. After verifying v_1, BS decrypts u_1 and retrieves R_1 and u_0. Then, BS verifies v_0 and decrypts u_0. Finally, BS retrieves R_0 and TS_0 . BS generates and sends u_4, v_4, and v_3 to S_2 where,$u_3 = E_{K_{S_1}}\{R_1||h(TS_0)\}$, $v_3 = MAC_{IK_{S1}}(\text{BS}||S_1||u_3)$, $u_4 = E_{K_2}\{R_1||u_3\}$ and $v_4 = MAC_{IK_2}(\text{BS}||S_2||R_1||u_4||v_3)$

$$BS \rightarrow S_2 : BS||S_2||S_1||u_4||v_4||v_3$$

P-1.c. After verifying v_4, S_2 decrypts u_4, and retrieves R_1 and u_3. S_2 generates $K_{S_1S_2} = KDF(0||R_0||R_1)$ and $IK_{S_1S_2} = KDF(1||R_0||R_1)$ with R_0 and R_1. $K_{S_1S_2}$ is encryption key and $IK_{S_1S_2}$ is integrity key between S_1 and S_2. Then S_2 generates $v_5 = MAC_{IK_{S_1S_2}}(S_2||S_1||R_0||R_1)$ and sends u_3, v_3, and v_5 to S_1.

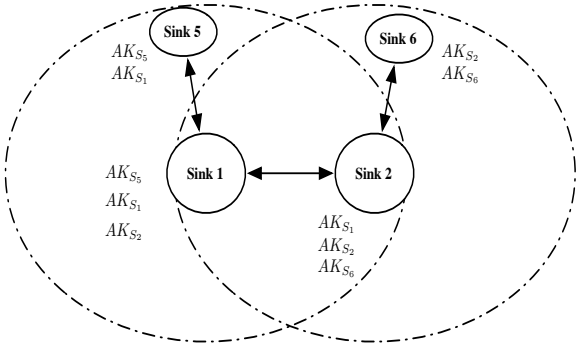
$$S_2 \rightarrow S_1 : S_2||S_1||u_3||v_3||v_5$$

P-1.d. After verifying v_3, S_1 decrypts u_3 and retrieves R_1. S_1 also generates $K_{S_1S_2}$ and $IK_{S_1S_2}$. Then S_1 verifies v_5. S_1 generates $v_6 = MAC_{IK_{S_1S_2}}(S_1||S_2||ACK||R_0||R_1)$ and sends v_6 with ACK to S_2.

$$S_1 \rightarrow S_2 : S_1||S_2||ACK||v_6$$

P-1.e. S_2 verifies v_6 and shares pairwise keys $K_{S_1S_2}$ and $IK_{S_1S_2}$.

Figure 9. Neighbor group authentication key share (Phase 2): sinks share neighbor sink's authentication keys.



3.3.3. Phase 2: Neighbor Group Authentication Key Share

Phase 2 can be operated solely or after Phase 1 is completed. In Phase 2, S_1 initiates following procedures.

P-2.a. S_1 randomly selects two nonces $ASEED_{S_1}$ and R_1. Then S_1 generates $u_1 = E_{K_{S_1S_2}}\{ASEED_{S_1}||R_1\}$ and $v_1 = MAC_{IK_{S_1S_2}}(S_1||S_2||u_1)$.

$$S_1 \rightarrow S_2 : S_1||S_2||u_1||v_1$$

P-2.b. After verifying v_1, S_2 decrypts u_1, and retrieves $ASEED_{S_1}$ and R_1. Then S_2 generates $AK_{S_1} = KDF(0||ASEED_{S_1})$ and $AIK_{S_1} = KDF(1||ASEED_{S_1})$. S_2 also generates $v_2 = MAC_{AIK_{S_1}}(S_2\ ||S_1||\text{ACK}||AR_1)$ using AIK_{S_1}.

$$S_2 \rightarrow S_1 : S_2||S_1||\text{ACK}||v_2$$

P-2.c. S_1 verifies v_2.

After the Phase 2 is completed, sinks share their neighbor sink's authentication keys as in Figure 9.

3.3.4. Phase 3: Initial Node Authentication

When N receives HELLO that S_1 broadcasts in Phase 0 and is not yet authenticated by any sink, N proceeds followings.

P-3.a. Node N randomly selects R_1 and generates $u_1 = E_{K_N}\{R_1||u_0||v_0\}$ and $v_1 = MAC_{IK_N}(N_1||S_1||u_1)$.

$$N \rightarrow S_1 : N||S_1||u_1||v_1$$

P-3.b. S_1 generates $v_2 = MAC_{IK_{S_1}}(S_1||\text{BS}||N||u_1||v_1)$.

$$S_1 \rightarrow \text{BS}{:}S_1||\text{BS}||N||u_1||v_1||v_2$$

P-3.c. After verifying v_2 and v_1, BS decrypts u_1, and retrieves R_0, u_0 and v_0. After verifying v_0, BS decrypts u_0, and retrieves R_0 and TS. BS checks the validity of TS and generates $u_3 = E_{K_N}\{R_0\}$, $v_3 = MAC_{IK_N}(BS||N||S_1||u_3)$, $u_4 = E_{K_{S_1}}\{R_1||u_3||v_3\}$ and $v_4 = MAC_{IK_{S_1}}(BS||S_1||N||R_0||u_4)$.

$$BS \rightarrow S_1 : BS||S_1||N||u_4||v_4$$

P-3.d. After verifying v_4, S_1 decrypts u_4, and retrieves R_1, u_3 and v_3. Then S_1 generates $NK_N = KDF(R_0||R_1)$. S_1 generates $t = E_{AK_{S_1}}\{TS||R_1||NK_N\}$ and $w = MAC_{AIK_{S_1}}(N||t)$. Next, S_1 also generates $u_5 = E_{NK_N}\{TS||t||w\}$ and $v_5 = MAC_{NIK_N}(S_1||N||R_0||u_5)$.

$$S_1 \rightarrow N : S_1||N||u_3||v_3||u_5||v_5$$

P-3.e. After verifying v_3, N decrypts u_3 and retrieves R_0. Then N also generates NK_N and verifies v_5. N decrypts u_5 and retrieves TS, t and w. N generates $v_6 = MAC_{NK_N}(N||S_1||ACK||R_0||R_1)$.

$$N \rightarrow S_1 : N||S_1||\text{ACK}||v_6$$

P-3.f. S_1 verifies v_6.

3.3.5. Phase 4: Node Re-Authentication

When N receives HELLO that S_2 broadcasts in Phase 0 and is previously authenticated by a sink, N proceeds followings.

P-4.a. N generates $v_1 = MAC_{NIK_N}(N||S_2||t||w||v_0)$.

$$N \rightarrow S_2 : N||S_2||t||w||v_1$$

P-4.b. S_2 verifies w and decrypts t. S_2 retrieves R_1, NK_N and TS. Using NK_N, S_2 verifies v_1. Then S_2 generates $NK' = KDF(R_1||R_0)$, also generates $t' = E_{AK_{S_2}}\{R_1||NK'_N\}$ and $w' = MAC_{AIK_{S_2}}(N||t')$. S_2 generates $v_2 = h(NK'_N||R_0)$ and $u_3 = E_{NK_N}\{R_0||v_2||t'||w'\}$, $v_3 = MAC_{NIK_N}(S_2||N||u_3)$.

$$S_2 \rightarrow N : S_2||N||u_3||v_3$$

P-4.c. After verifying v_3, N decrypts u_3 and retrieves R_0, v_2 , t' and w'. Then N generates NK'_N and verifies v_2. N generates $v_4 = MAC_{NIK'_N}(N||S_2||\text{ACK}||R_0||R_1)$.

$$N \rightarrow S_2 : N||S_2||\text{ACK}||v_3$$

P-4.d. After verifying v_4, S_2 authenticates N.

Brief procedures of Phase 3 and Phase 4 are shown in Figure 10.

4. Analysis

In this section, we show the performance and security analysis of our protocol. Section 4.1 shows the comparison to the previous protocols, and Section 4.2 shows the security analysis for the requirements and known attacks in WSN.

4.1. Performance Analysis

For the performance analysis, we compared the number of communication passes, the required message sizes, and the number of computation of the protocol. We do not count the overhead in Phase 0, since Phase 0 does not initiate the protocol. The node just ignores Phase 0 when the node receives HELLO from the sink that already authenticated the node.

Figure 10. Phase 3: Node requests initial authentication to Sink 1. Phase 4: Node requests re-authentication to Sink 2

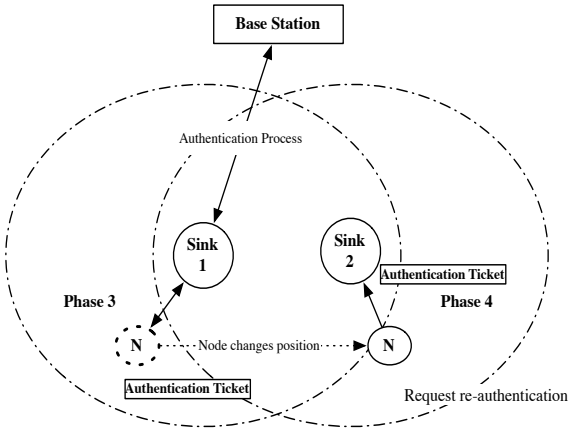


Table 2. Comparison of required communication pass for re-authentication.

	Fantacci *et al.*'s Model [10]	Ibriq and Mahgoub's model [7]	Proposed
Node	2	$2n$	$2n$
Sink	$2t+1$	$2t$	1
Base station	–	2	–

4.1.1. Communication Pass

We compared the required number of communication passes with Fantacci *et al.*'s model [10] and Ibriq and Mahgoub's model [7]. The reason is that [10] considered node mobility without requiring sinks or base station in the key distribution, and [7] showed the efficient key distribution in static networks. Table 2 shows the comparison of communication passes for node re-authentication, where n denotes the number of nodes and t denotes the number of sinks. Since nodes act as the authentication server (the base station) and the authenticator (the sink), all the communications in [10] are operated among nodes.

Comparison of required number of communication pass in initial authentication is the same as the previous models. In node re-authentication, our novel protocol has much more efficiency compared with other protocols [7,10], since our protocol does not require the communication with the base station in re-authentication.

In practical application, we can deploy the network that all nodes directly connect to any sinks (*i.e.*, $n = 1$). In that case, the communication passes in our protocol are just three passes (*challenge-response-confirmation*).

4.1.2. Message Size

We compared Abraham and Ramanatha's model [6,7] for the required message size for authentication. Based on the results in [6], we approximately compared the message sizes based on the message size

with MAC size as 4 bytes, the time stamp as 8 bytes, nonce as 8 bytes, and key size as 16 bytes. We also set the source and target IDs as 1 byte, respectively.

Tables 3 and 4 show the message sizes in the initial authentication and the message sizes in re-authentication with 2 hops between sink and base station, respectively. Table 3 shows that the performance for the initial authentication is similar to other protocols. In initial authentication (Phase 3), Abraham and Ramanatha's model [6] showed the best result—30 bytes less in message sizes than our protocol. However, as Table 4 shows, our protocol achieves about a third overall message size than other protocols. Even when we increase the size of each parameter, our protocol is still much more efficient than any other protocols in node re-authentication.

Table 3. Comparison of required message size for initial authentication (bytes).

	Abraham's model [6]	Ibriq and Mahgoub's model [7]	Proposed
Node to Sink	46	68	56
Sink to Sink	70	76	62
Sink to Base station	70	76	66
Base station to Node	92	188	180
Total message size	278	408	302

Table 4. Comparison of required message size for re-authentication (bytes).

	Abraham's model [6]	Ibriq and Mahgoub's model [7]	Proposed
Node to Sink	46	68	44
Sink to Sink	70	76	-
Sink to Base station	70	76	-
Base station to Node	92	188	64
Total message size	278	408	108

For the comparison in multi-hop environments, Figures 11 and 12 show the message sizes of initial authentication (Phase 3) and re-authentication (Phase 4) in our protocol and the comparison with other protocols, respectively. When the hop distances between the sinks to which the node is attached and the base station increase, the required message size and the communication pass also increase.

Figure 11. Comparison of message sizes with initial authentication and re-authentication per hop distance from sink to the base station increases.

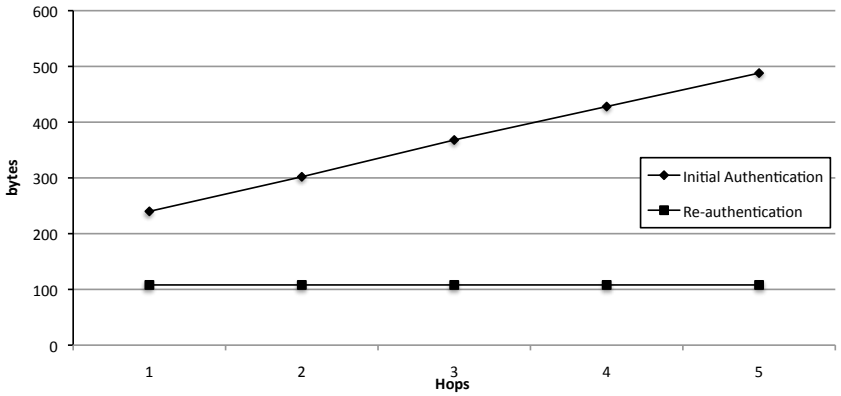
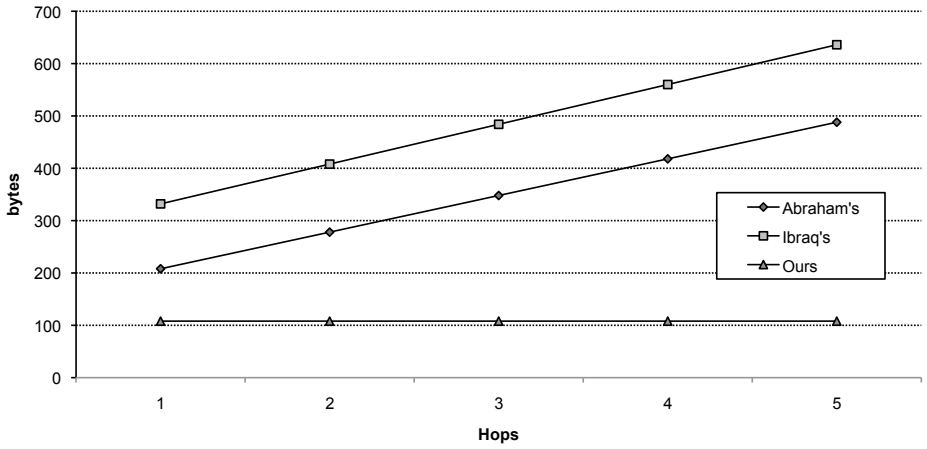


Figure 12. Comparison of message sizes with [6] and [7] per hop distance between a sink and a base station



4.1.3. Computation

Now, we compare the computational overhead of initial authentication (Phase 3) and re-authentication (Phase 4). In total, 10 times of encryption/decryption and 14 times of MAC generation/verification are required for initial authentication, while 4 times of encryption/decryption and 10 times of MAC generation/verification are required for re-authentication. For node specific operation, 3 times of encryption/decryption for initial authentication, 1 time of encryption/decryption are required. Both cases require 4 times of MAC generation/verification. Since the computation of MAC does not have significant overhead, comparing the computation of encryption and decryption, our computation is 2–3 times more efficient. The comparison of computation is shown in Table 5. We do not measure the computation time of each operation that depends on the encryption and hash algorithms in this paper. Note that we can apply TinySEC [17] and TinyHash [18] for the implementation.

Table 5. Comparison of computation between initial authentication and re-authentication (times).

	Initial Authentication	Re-authentication.
Encryption/Decryption in Total	10	4
Encryption/Decryption by Node	3	1
MAC Generation/Verification in Total	14	10
MAC Generation/Verfication by Node	4	4

4.2. Security Analysis

We show the security analysis of our protocol that holds the requirements defined in Section 2.3. "re-authentication", "untraceability", "confidentiality", "message integrity", "key freshness", and "node/sink resiliency". Then, we analyze the security of our protocol against known attacks.

4.2.1. Re-Authentication

After a node N is initially authenticated by a sink S_1 in phase 3, the node receives the authentication ticket (t, w) and v_1, where $t = E_{AK_{S_1}}\{TS||R_1||NK_N\}$, $w = MAC_{AIK_{S_1}}(N||t)$ and $v_1 = MAC_{NIK_N}(N||S_2||t||w||v_0)$. When N moves and requests re-authentication to the neighbor sink S_2, S_2 can verifies (t, w) since the authentication key of S_1, AK_{S_1} is shared to S_2. N can authenticates S_2 with u_3 and v_3 with NK_N. Finally, S_2 authenticates N after verification of v_4. In the re-authentication phase, the base station is not involved.

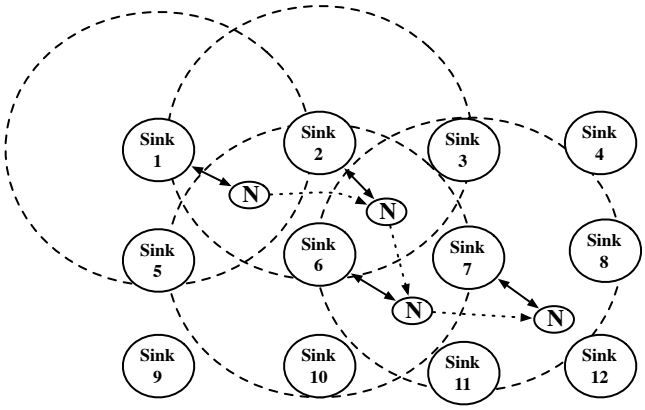
4.2.2. Untraceability

A sink S_1 issues the authentication ticket (t, w) to a node N. However, S_1 does not know the next move of N. N can be re-authenticated by any neighbor sinks of S_1. For the re-authenticated sink S_2, S_2 only knows that N was previously authenticated by S_1, but never knows the direction N ahead. Sinks only know N was previously authenticated by neighbor sinks, but never predict N's next direction as in Figure 13.

4.2.3. Confidentiality

Any sinks and nodes pre-share secret keys only with the base station. For the Neighbor discovery phase, the neighbor discovery message is encrypted using K_S that is only shared between a sink and the base station. For setting up the neighbor group and node authentication, the adversary requires shared secret key to know the information. For the node re-authentication, the responses u_3 and v_3 are encrypted using NK_N that is known to S_1. However, we assume that the re-authentication happens, where S_1 cannot involve in the communication from out-of-reach.

Figure 13. When N move in the networks, sinks re-authenticate N without knowing the node's direction.



4.2.4. Message Authentication

In our protocol, every packet is protected by 4 bytes MAC. The outside adversary should be able to forge the message to succeed in the attack. The security of the MAC depends on the security of the hash function. The recommended MAC size in [17] is 4 bytes for practical application, since only 40 forgery attempts per second are available on a 19.2 kb/s channel while 2^{31} trials are required for successful forgery. However, the performance of communication channel is increasing, and the size of MAC should be increased in future applications. Recently the efficient implementation of hash functions is introduced in [18]. Thus, our protocol is secure against the man-in-the-middle attack, as the adversary has no efficient way to forge MAC even when the part of the network is compromised by the attacker.

4.2.5. Key Freshness

In Phase 0, the sink S_1 periodically generates random nonce R_0. Thus, S_1 can verify that the requests of authentication are from the directly linked sinks or nodes. In Phase 1, two entities generate the random nonces whose freshness can be checked by both entities. In Phase 2, S_1 also generates random nonce R_1 for the freshness check. In Phase 3 and 4, the node also generates random nonce R_1 to check the freshness.

4.2.6. Node/Sink Resiliency

We can define two kinds of threat of sink capture: the sink missing case and the compromised sink case. When a sink S_1 is just missing, the node will lose the connection S_1 and find other sink such as S_2. Thus, we only need to consider the compromised sink case.

When the sink is compromised, we can assume that the keys in the sink are leaked. However, even if the group authentication key is leaked, only will the neighbor sinks be affected. The compromised sink can self-attach the fake nodes that will request re-authentication without initial authentication. For

this case, we add $h(K_N||R_1)$ in the authentication ticket that is sent to the sink when the node requests re-authentication. For suspicious nodes, the sink can check if the node is genuine with help of the base station. Also, we need to define the security policy for the extreme abnormality in deploying sensor network application. When the node is compromised, we can define that the compromised node may try to know the information of the sinks or impersonate other nodes. However, the compromised node will fail in both cases, since the node does not share any information in the protocol. Thus, our protocol has node and sink resiliency, and is practically secure against selective forwarding and acknowledgement spoofing.

4.2.7. Security Against Known Attacks

We analyze the security of our protocol against the attacks identified in [19]. Since the static parts in the networks could follow the previous models such as [7], we only focus on the security of node re-authentication in this section.

The sinkhole attack against our protocol fails without knowing the keys. An adversary A may capture the authentication ticket (t, w) that N initially sent to S_2, and A send (t, w) to S_2 or other sink S_5 that is also a neighbor sink of S_1. However, A fails in such attack without knowing AK_{S_1}. Wormhole attack on our protocol fails since the adversary cannot send the confirmation message. Spoofed, altered or replayed routing information attack also fail without knowing the encrypted nonce in our protocol. To succeed in the replay attack, the adversary has to be able to re-use the intercepted packet. We do not consider relaying through the attackers as successful attack. Sybil attack also fails from verification of identity of nodes through sinks and the base station. As for HELLO flood attacks, we can apply the global key shared to all entities in the network that many researches such as [7,16] used for the efficient message broadcast and DoS attack protection.

5. Conclusions

Node mobility is one of the emerging issues in WSN that needs to be adequately addressed. In this paper, we outlined the drawbacks of previous authentication protocols supporting mobile nodes in WSN, and identified the following requirements: efficient node re-authentication and untraceability. We then proposed our novel efficient node authentication and key distribution protocol that provides re-authentication and untraceability. Also, we analyzed our protocol by comparing it with the previous protocols. Our protocol requires only three passes of communication with one third of communication message sizes compared with previous protocols in node re-authentication. The computational overhead of node re-authentication of a single mobile node achieves about 2–3 times more efficiency than that of initial node authentication. It is obvious that deploying our protocol in the environment with large numbers of mobile nodes will achieve much higher cost efficiency than any previous methods. Our future plan is to gain the energy efficiency of sensor network in the initial authentication process of our protocol. Thus, We expect that our proposed protocol will be the efficient security solution supporting mobile nodes in WSN.

References

1. Craig, W.C. Zigbee: Wireless Control That Simply Works. *Zigbee Alliance* **2005**, *19* , 21-27.
2. Eschenauer, L.; Gligor, V. A key management scheme for distributed sensor networks". In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS)*, Washington, DC, USA, 18-22 November 2002; pp. 41–47.
3. Chan, H.; Perrig, A.; Song, D. Random Key Predistribution Schemes for Sensor Networks. In *Proceedings of IEEE Symposium on Security and Privacy* , Berkeley, CA, USA, 11-14 May 2003; pp. 197–213.
4. Du, W.; J. Deng, Y.S.H.; Varshney, P.K. A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS)*, Washington, DC, USA, 27-31 October 2003; pp. 42–51.
5. Zhu, S.; Setia, S.; Jajodia, S. LEAP: Efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, ACM: New York, NY, USA, October 2003; pp. 62–72.
6. Abraham, J.; Ramanatha, K.S. An Efficient Protocol for Authentication and Initial Shared Key Establishment in Clustered Wireless Sensor Networks. In *Proceedings of Third IFIP/IEEE International Conference on Wireless and Optical Communications Networks*, Bangalore, India, 11-13 April 2006.
7. Ibriq, J.; Mahgoub, I. A Hierarchical Key Establishment Scheme for Wireless Sensor Networks. In *Proceedings of 21st International Conference on Advanced Networking and Applications (AINA'07)*, Niagara Falls, Canada, May 2007; pp. 210–219.
8. Krishnakumar, S.S.; Abler, R.T. Intelligent Actor Mobility in Wireless Sensor and Actor Networks. In *Proceedings of IFIP International Federation for Information Processing, Wireless Sensor and Actor Networks*, Orozco-Barbosa, L., Olivares, T., Casado, R., Bermudez, A., Eds., Springer: Boston, MA, USA, 2007; pp. 13–22.
9. Das, S.; Liu, H.; Kamath, A.; Nayak, A.; Stojmenovic, I. Localized Movement Control For Fault Tolerance of Mobile Robot Networks. In *IFIP International Federation for Information Processing, Wireless Sensor and Actor Networks*; L. Orozco-Barbosa, Olivares, T., Casado, R., Bermudez, A., Eds.; Springer: Boston, MA, USA, 2007; p. 248.
10. Fantacci, R.; Chiti, F.; Maccari, L. Fast Distributed Bi-Directional Authentication for Wireless Sensor Networks. *Secur. Commun. Networks* **2008**, *1*, 17–24.
11. Liu, D.; Ning, P. Establishing Pairwise Keys in Distributed Sensor Networks. In *Proceedings of the 10th ACM Conference on Computer and Communication (CCS)*, New York, NY, USA, March 2003; pp. 52-61.
12. Shamir, A. How to share a secret. *Commun. ACM* **1979**, *22*, 612–613.
13. Huang, Q.; Cukier, J.; Kobayashi, H.; Liu, B.; Zhang, J. Fast Authenticated Key Establishment Protocols for Self-Organizing Sensor Networks. In *Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, San Diego, CA, USA, 19 September 2003.

14. Watro, R.; Kong, D.; fen Cuti, S.; Gardiner, C.; Lynn, C.; Kruus, P. TinyPK: Securing Sensor Networks with Public Key Technology. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, ACM Press: New York, NY, USA, 25 October 2004; pp. 59–64.
15. Liu, A.; Ning, P. TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks. In *Proceedings of the 2008 International Conference on Information Processing in Sensor Networks*, Saint Louis, MO, USA, April 2008; pp. 109-120
16. Zhu, S.; Setia, S.; Jajodia, S. LEAP+: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. *ACM Trans. Sen. Netw.* **2006**, *2*, 500–528.
17. Karlof, C.; Sastry, N.; Wagner, D. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, Baltimore, MD, USA, 3-5 November 2004.
18. Lee, H.; Choi, Y.; Kim, H. Implementation of TinyHash based on Hash Algorithm for Sensor Network. In *Proceedings of World Academy of Science, Engineering and Technology*, Saint Louis, Mo, USA, August 2005; pp. 135–139.
19. Karlof, C.; Wagner, D. Secure Routing in Wireless Sensor Networks. In *Proceedings of SNPA'03*, Anchorage, AK, USA, 11 May 2003; pp. 113–127.