PKC Based Broadcast Authentication using Signature Amortization for WSNs

Yongsheng Liu, *Student Member, IEEE*, Jie Li, *Senior Member, IEEE*, and Mohsen Guizani, *Fellow, IEEE*

***Abstract*—Public Key Cryptography (PKC) is widely used for broadcast authentication. Intensive use of PKC for broadcast authentication, however, is thought to be expensive to resource constrained sensor nodes. In this paper, we propose a novel PKC based broadcast authentication scheme using signature amortization for Wireless Sensor Networks (WSNs). The proposed scheme exploits only one Elliptic Curve Digital Signature Algorithm (ECDSA) signature to authenticate all broadcast messages. Thus, the overhead for the signature is amortized over all broadcast messages. Besides low overhead, the proposed scheme retains high security that is as strong as conventional PKC based broadcast authentication schemes. Moreover, the proposed scheme can achieve immediate authentication and does not require time synchronization. For the implementation of the proposed scheme, an efficient public key distribution protocol is also presented in this paper. Experimental results of a testbed show that the overhead for authenticating a broadcast message is reduced significantly.**

***Index Terms*—Broadcast authentication, signature amortization, wireless sensor networks.**

I. Introduction

A WIRELESS Sensor Network (WSN) consists of one or more powerful base stations and hundreds of sensor nodes [1]. Base stations serve as gateways between Internet users and sensor nodes. Sensor nodes integrated with microcontroller, Radio Frequency (RF) transceivers and sensing units are spatially scattered into a specific area and monitor physical and environmental conditions. Due to potential applications in environmental monitoring, target tracking and object detection, etc, WSNs have been drawing great attention in recent years.

In WSNs, broadcast transmission is one of the fundamental communication primitives. Primarily, there are two kinds of broadcast transmissions. One is the network broadcast, which may be used by a base station for committing network queries [2] and disseminate code images [3]. The other is the local broadcast, which may be used for discovering neighbors [4] and exchange routing information [2]. Eavesdropping on the omnidirectional broadcast transmission, an adversary is able to intercept broadcast messages, change the intercepted messages on behalf of itself and rebroadcast them. Even, the adversary may fabricate messages and inject them to the network. These malicious messages may cause crash of sensor nodes or destruction of the entire network. To defend WSNs against malicious messages, it is necessary to authenticate broadcast messages, referred to as broadcast authentication.

Providing broadcast authentication for WSNs, however, encounters many challenges. The primary one is stringent resource constraints on sensor nodes. It results in conventional Public Key Cryptography (PKC) based broadcast authentication schemes are believed to be too expensive for WSNs. For instance, it takes 14 seconds for an exponential operation of 1024-bit RSA on Mica1 motes [5]. Besides resource constraints, sensor nodes are vulnerable to node compromise attacks [6]. This renders application of conventional symmetric key cryptography based broadcast authentication schemes to WSNs impractical.

Perrig *et al*. [7] propose a well-known broadcast authentication scheme for WSNs, called μTESLA. μTESLA exploits symmetric key cryptographic operations to authenticate broadcast messages. Thus, it is efficient to WSNs. μTESLA, however, is resilient to node compromise attacks because of delayed disclosure of secret keys. Many variants of μTESLA are proposed for WSNs (e.g., [8] [9] [10] [11]) to improve its performance. The variants as well as μTESLA, however, are subject to the following defects. First, maintenance of time synchronization in WSNs is a complicated task. Second, distribution of the initial parameters introduces heavy overhead since it is implemented by the unicast transmission. Third, delayed authentication is inevitable.

Ren *et al*. [12] and Du *et al*. [13] propose to authenticate broadcast messages of WSNs with PKC since overhead of PKC for WSNs has significantly been reduced by using Elliptic Curve Cryptography (ECC). Nevertheless, the PKC based broadcast authentication schemes so far are not affordable by current generation of sensor nodes because of the intensive use of Elliptic Curve Digital Signature Algorithm (ECDSA). Gennaro *et al*. [14] propose signature amortization to sign efficiently digital streams. A number of improvements are developed soon (e.g., [15] [16] [17] [18] [19]).

In this paper, to avoid intensive use of ECDSA, we propose a novel PKC based broadcast authentication scheme using signature amortization for WSNs. The proposed scheme exploits only one ECDSA signature to authenticate all broadcast messages. Thus, the overhead of the signature is amortized over

Manuscript received March 9, 2011; revised June 18, August 31, and December 11, 2011; accepted January 19, 2012. The associate editor coordinating the review of this paper and approving it for publication was Z. Han.

Y. Liu is with the Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba Science City, 305-8573, Japan (e-mail: liuyongsheng@osdp.cs.tsukuba.ac.jp).

J. Li (corresponding author) is with the Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba Science City, 305-8573, Japan (e-mail: lijie@cs.tsukuba.ac.jp).

M. Guizani is with Qatar University, Qatar (e-mail: mguizani@ieee.org).

Digital Object Identifier 10.1109/TWC.2012.032812.110433

all broadcast messages (section IV). Besides low overhead, the proposed scheme retains high security that is as strong as conventional PKC based broadcast authentication schemes (section IV). Moreover, defects of μTESLA can be overcome: the proposed scheme does not require time synchronization (section III), has an efficient public key distribution protocol (section III) and can achieve immediate authentication that a receiver authenticates a broadcast message upon receiving it (section IV).

The rest of this paper is organized as follows. The system model is introduced in section II. The proposed PKC based broadcast authentication scheme using signature amortization is presented in section III. Overhead and security of the proposed scheme are analyzed in section IV. Evaluations using a testbed are illustrated in section V. We conclude this paper in section VI.

II. System Description

For simplicity, we consider only the case of one base station within a WSN. Since WSNs are generally composed of a collection of base stations and sensor nodes are connected to them, the proposed scheme can also be implemented with a WSN having multiple base stations. We assume that there is only one base station that is endowed with sufficient energy supply. Furthermore, it cannot be compromised by adversaries, which is a common assumption in the literature of WSNs (e.g., [20]). Contrary to the base station, sensor nodes are resource constrained and vulnerable to adversaries. We assume that a senor node is able to perform a limited number of PKC operations but not many. In the WSN, both the network broadcast and the local broadcast are considered. For clarity, we consider that one sender broadcasts messages to many receivers. In the network with multiple senders, each sender and its corresponding receivers are just the case we are considering. The sender may be the base station or a sensor node.

In a WSN, there may be several adversaries with powerful resources (e.g., laptops). They can launch both external as well as internal attacks. In the external attacks, adversaries can eavesdrop for sensitive information, inject forged messages, and replay previously intercepted messages. They can also launch Denial-of-Service (DoS) attacks and jam the communication channel. But, we assume that the DoS and jam attacks cannot continue constantly without being detected and removed. In the internal attacks, adversaries are able to compromise some sensor nodes and obtain the sensitive information (e.g., secret keys). Later, the sensitive information may be used for attacking the rest of senor nodes in the network.

For the sake of convenience, we make use of the following notations. The base station is denoted by bs. It holds a key pair (PR_{bs}, PU_{bs}) where PR_{bs} is the private key and PU_{bs} is the public key. PU_{bs} is assumed to be stored by all sensor nodes in the network. The sender is denoted by s. It holds a key pair (PR_s, PU_s) where PU_s is the public key and PR_s is the private key. Corresponding to sender s, receivers are denoted by a vector $R_s = [r_i]_{i=1,\ldots,c}$, in total c receivers. Broadcast messages from sender s to receivers in R_s are denoted by a vector $M = [m_i]_{i=1,\ldots,n}$, in total n messages. The broadcast messages in M are organized through extended blocks. An extended block is regarded as a unit authenticating broadcast messages in M. The extended blocks are denoted by a vector $EB = [EB_i]^T_{i=0,\ldots,k}$, in total $k+1$ extended blocks. The first extended block EB_0 contains an authenticator. An authenticator is the additional information to authenticate an extended block (e.g., the Message Authentication Code (MAC) or the signature). Each of the other extended blocks contains b broadcast messages in M and a specified authenticator. We assume that the number of broadcast messages n is an integer multiple of b, i.e., $n = kb$. If not, simply let extended block EB_k contain $(n \bmod b)$ messages provided that $k = \lceil n/b \rceil$. Encryption of a message m is denoted by $E(K, m)$ where K is the key. Similarly, decryption of a message m is denoted by $D(K, m)$ where K is the key.

Definition 1 (Authenticated Relation (AR))**.** *AR on EB consists of ordered pairs $< EB_i, EB_j >$ where authenticator in EB_i is used to authenticate EB_j.*

Definition 2 (Collision resistant hash H)**.** *H is the hash satisfying that it is computationally infeasible to find a pair of inputs (x, y) such that $x \neq y$ and $H(x) = H(y)$.*

Definition 3 (Certificate)**.** *A certificate consists of a public key and the identity of the public key's owner, both of which are signed by a trusted third party called Certificate Authority (CA).*

III. Proposed PKC based Broadcast Authentication Scheme using Signature Amortization

We propose a novel PKC based broadcast authentication scheme using signature amortization for WSNs, which meets the following properties.

- Low overhead. The computation and communication overhead is to the same degree of the Keyed-Hash Message Authentication Code (HMAC).
- Strong authenticity. Confidence of a receiver in authenticating broadcast messages is as strong as each extended block in EB is authenticated by an ECDSA signature.
- Immediate authentication. A receiver can authenticate broadcast messages upon receiving them.
- No time synchronization. Time synchronization is not required.
- Resilience to node compromise attacks. It is impossible for an adversary to exploit a compromised receiver to launch a valid broadcast authentication.

The proposed PKC based broadcast authentication scheme using signature amortization exploits one ECDSA signature to authenticate all broadcast messages. The only one signature is used to authenticate the authenticator in EB_0. The authenticator in EB_0 is used to authenticate EB_1 that contains b broadcast messages in M and one authenticator. The authenticator in EB_1, in its turn, is used to authenticate EB_2 that contains b broadcast messages in M and one authenticator. The process continues until EB_k. As a result, all broadcast messages can be authenticated with only one signature while the overhead of the signature is amortized over them.

This section is presented in three parts. The first part presents procedures of signature amortization. The second part details the ECDSA signature. The third part introduces an efficient public key distribution protocol for implementing the proposed scheme.

A. Signature Amortization

The signature amortization part is presented by three steps: generating extended blocks step, broadcasting extended blocks step and verifying extended blocks step.

1) Generating Extended Blocks Step: Based on the definition of AR, the basis to construct extended blocks is provided by theorem 1.

Theorem 1. *All broadcast messages in M will be authenticated if EB_0 is authentic and all ordered pairs $< EB_{i-1}, EB_i >$, $1 \leq i \leq k$, belong to authenticated relation AR on EB.*

Proof: This is proved by mathematical induction. The basic step is authentication of EB_1. EB_1 is authenticated by the authenticator in EB_0 which is authentic. Hence, b broadcast messages in EB_1 will be authenticated. In the inductive step, suppose EB_{j-1}, $2 \leq j \leq k$, is authentic. Then, the authenticator in EB_{j-1} can be used to authenticate EB_j since $< EB_{j-1}, EB_j >$ belongs to AR on EB. Thus, b messages in EB_j will be authenticated. In short, n broadcast messages in M will be authenticated. ■

Authenticators for extended blocks can be generated by three classes of functions roughly: message encryption, MAC and hash function. Symmetric encryption, MAC and hash function are efficient to WSNs. Symmetric encryption and MAC require two input parameters: a secret key and a message. The shared secret key between the sender and receivers is vulnerable to node compromise attacks. Therefore, we adopt a collision resistant hash to produce authenticators for extended blocks.

With the collision resistant hash to produce authenticators, extended blocks are generated as follows.

All n broadcast messages in M is partitioned into k blocks $B_1, \ldots, B_k$. Every block B_i, $1 \leq i \leq k$, contains b messages, denoted by $|B_i| = b$. These blocks comprise a vector $B = [B_i]^T_{i=1,\ldots,k}$ that is expressed in the matrix form as shown in equation (1), recalling that we assume that $n = kb$.

$$B = \begin{bmatrix} B_1 \\ \vdots \\ B_k \end{bmatrix} = \begin{bmatrix} m_1 & \ldots & m_b \\ \vdots & \vdots & \vdots \\ m_{(k-1)b+1,} & \ldots & m_{kb} \end{bmatrix} \quad (1)$$

Messages in each row of B, accurately block B_i, are concatenated into a long string, denoted by $CON(B_i)$ as shown in equation (2).

$$CON(B_i) = m_{(i-1)b+1}||...||m_{ib}, 1 \leq i \leq k \quad (2)$$

Then, $CON(B_i)$, $1 \leq i \leq k$, is padded with authenticators, denoted by $PAD(CON(B_i))$ as shown in equation (3). $CON(B_j)$, $1 \leq j \leq k-1$, is padded with digest d_{j+1} which is the digest of $PAD(CON(B_{j+1}))$, as shown in equation (4) where H is a collision resistant hash. Note that $CON(B_k)$ is padded with random characters which is denoted by d_{k+1} for consistency.

$$PAD(CON(B_i)) = CON(B_i)||d_{i+1}, 1 \leq i \leq k \quad (3)$$

$$d_{j+1} = H(PAD(CON(B_{j+1}))), 1 \leq j \leq k-1 \quad (4)$$

Block B_i and digest d_{i+1} comprise an extended block EB_i, i.e., $EB_i = [B_i\ d_{i+1}]$, $1 \leq i \leq k$. Let EB_i, $1 \leq i \leq k$, comprise a vector EB^*, i.e., $EB^* = [EB_i]^T_{i=1,\ldots,k} = [B_i\ d_{i+1}]^T_{i=1,\ldots,k}$. Because EB^* stands for an extended matrix of B, EB_i is called an extended block.

Note that exceptional EB_0 only contains an authenticator. The authenticator is digest d_1 that is the collision resistant hash value of $PAD(CON(B_1))$, i.e., $d_1 = H(PAD(CON(B_1)))$. The authenticity of d_1 is guaranteed by a signature generated by sender s with PR_s. Thus, $EB_0 = d_1||E(PR_s, d_1)$.

Until this point, all extended blocks have been generated. For clarity, let all extended blocks comprise a vector EB, i.e., $EB = [EB_i]^T_{i=0,\ldots,k}$. The complete description of generating EB is presented in algorithm 1. Fig. 1 illustrates an example of generating EB for 9 broadcast messages, in which every 3 broadcast messages constitute a block.

Algorithm 1 Generation of EB

1: Partition n broadcast messages in M into k blocks $B_1, \ldots, B_k$ as shown in equation (1)
2: Initialize d_{k+1} with a string of random characters
3: **for** $i = k; i \geq 1; i = i - 1$ **do**
4: Concatenate messages in B_i to generate $CON(B_i)$ as shown in equation (2)
5: Pad $CON(B_i)$ with digest d_{i+1} to generate $PAD(CON(B_i))$ as shown in equation (3)
6: Compute digest d_i of $PAD(CON(B_i))$ with a collision resistant hash as shown in equation (4)
7: Let $EB_i = [B_i\ d_{i+1}]$
8: **end for**
9: Sign digest d_1 with sender s's private key PR_s to generate $EB_0 = d_1||E(PR_s, d_1)$
10: Let $EB = [EB_i]^T_{i=0,\ldots,k}$

Based on theorem 1 and generating process of EB, authenticating n broadcast messages in M proceeds as follows. The signature in EB_0 is used to authenticate d_1. d_1, in its turn, is used to authenticate EB_1 that contains B_1 and d_2. Then, d_2 is used to authenticate EB_2 that contains B_2 and d_3. This process continues until EB_k. Eventually, n broadcast messages in M will be authenticated.

2) Broadcasting Extended Blocks Step: In generating extended blocks step, all ordered pairs $< EB_{i-1}, EB_i >$, $1 \leq i \leq k$, belong to AR on EB. That is EB_i's authentication depends on EB_{i-1}. Thus, EB_{i-1} should reach receivers before EB_i. This is fulfilled by the sequential broadcast and reliable broadcast described as follows.

The sequential broadcast is that extended blocks are broadcast according to AR on EB. Sender s broadcasts EB_{i-1} before EB_i. For simplicity, messages in each extended block are broadcast according to their indexes, i.e., sending sequence for EB_i is $m_{(i-1)b+1}, m_{(i-1)b+2}, \cdots, m_{ib}$. Digest d_{i+1} in EB_i could be sent together with a broadcast message in EB_i since

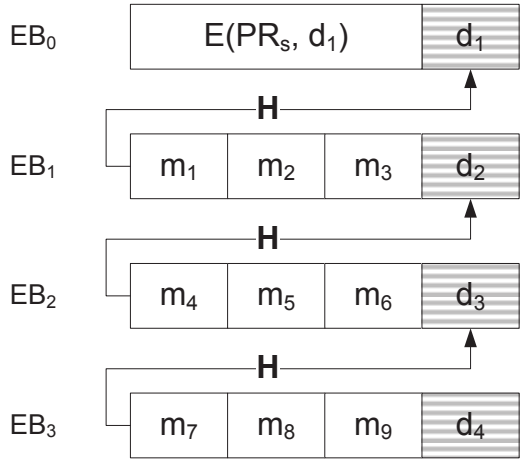


Fig. 1. An example of generating $EB = [EB_i]^T_{i=0,\ldots,3}$ for 9 broadcast messages (EB_i: extended block i; $E(PR_s, d_1)$: encryption of d_1 with private key PR_s; d_i: digest i; H: collision resistant hash; m_i: broadcast message i).

the size of a digest is relatively small. On receiving a broadcast message m_j, a receiver in R_s checks whether m_j belongs to current extended block, say, EB_i, whose digest, d_i, has been received and authenticated with EB_{i-1}. If m_j belongs to EB_i, the receiver tries authenticating EB_i and broadcasts m_j to its neighbors after a short backoff. This means all receivers exchange messages of EB_i with each other. During the exchange, a receiver may receive multiple copies of m_j from different neighbors. Thus, the receiver would obtain m_j with high probability though each transmission is not reliable. In short, the unreliable transmission is counteracted by making full use of broadcast nature. If m_j belongs the block EB_{i+1} which follows EB_i and EB_i has not been authenticated yet, that indicates all messages of EB_i has been broadcast. The receiver would not get missing messages later. Hence, the receiver buffers m_j and broadcasts an acknowledgement to ask for the missing messages belonging to EB_i. After authentication of EB_i, m_j will be broadcast.

The reliable broadcast is performed by acknowledgements and replies. To reduce communication overhead, we let one acknowledgement specify all missing messages of one extended block but the size of an acknowledgement be several bits larger than that to one missing message. The acknowledgement contains two fields. The first field specifies the identity of an extended block. The second field is a bit-vector indicating all missing messages in one extended block. The bit-vector is a mapping to all messages in one extended block. Thus, the size of the bit-vector equals the number of messages in the extended block. The receiver's neighbors are responsible for rebroadcasting the lost messages specified in an acknowledgement. Since the neighbors broadcast m_j belonging to EB_{i+1} to the receiver, they possess all messages belong to EB_i. Hence, it is unnecessary to ask sender s that may be multi-hops far away for m_j. To avoid collision, each neighbor selects a random time to delay its reply.

We employ the sequential broadcast and reliable broadcast to guarantee the successful authentication of extended blocks with low overhead. As well, they ensure the integrity of M to applications.

3) Verifying Extended Blocks Step: According to broadcasting extended blocks step, EB_0 reaches receivers in R first. d_1 in EB_0 is authenticated by the signature, that is, if $D(PU_s, E(PR_s, d_1)) = d_1$, d_1 is authentic. Extended blocks in EB^* are authenticated in an efficient way, just using a collision resistant hash. Digest d_i, $1 \leq i \leq k$, in EB_{i-1} that reaches receivers in R in advance is used to authenticate EB_i, that is, if $H(m_{(i-1)b+1}||\ldots||m_{ib}||d_{i+1}) = d_i$, EB_i is authentic.

B. The ECDSA Signature

The ECDSA signature is on the basis of ECC, which offers equivalent security with substantially smaller key size compared to RSA [21] [22] [23] (e.g., a 160-bit key of ECC offers the same level of security as a 1024-bit key of RSA). Thus, ECC has the advantages in computation, bandwidth and memory savings. Because of the advantages, d_1 in EB_0 is signed with an ECDSA signature. Here, we introduce briefly generation and verification of the ECDSA signature according to the description in [24].

Sender s and receivers in R_s establish elliptic curve domain parameters $T = (p, a, b, G, q, h)$ in advance. Storing T in sensor nodes before deployment is an option. p is a prime that specifies the finite field F_p. a and b are coefficients of the elliptic curve $y^2 \equiv x^3 + ax + b \pmod p$ where $4a^3 + 27b^2 \not\equiv 0 \pmod p$. G refers to the base point on the elliptic curve. q is a prime indicating the order of G. h is the cofactor $h = \#E_p(a,b)/q$ where $\#E_p(a,b)$ stands for the number of points on the elliptic curve.

To sign digest d_1, sender s creates the key pair (PR_s, PU_s) that satisfies $PU_s = PR_sG$ where PR_s is an integer in F_p and PU_s is a point on the elliptic curve. Then, it selects an ephemeral key pair (u, U) that satisfies $U = uG$ where u is an integer in F_p and U is a point on the elliptic curve. It computes $r \equiv x_U \pmod q$ where x_U is the x coordinate of point U, and $d_e = H(d_1)$ where H is a collision resistant hash. It sets $e = d_e$ if $\lceil log_2 q \rceil \geq L_{d_e}$ where L_{d_e} refers to the length of d_e. Otherwise, let e equal the leftmost $\lceil log_2 q \rceil$ bits of d_e. At last, it computes $w \equiv u^{-1}(e + rPR_s) \pmod q$. r and w are the ECDSA signature. The complete expression of EB_0 is $EB_0 = d_1||r||w$.

Verification of d_1 proceeds as follows. The receiver computes $d_e = H(d_1)$ where H is a collision resistant hash. Then, it sets $e = d_e$ if $\lceil log_2 q \rceil \geq L_{d_e}$ where L_{d_e} refers to the length of d_e. Otherwise, let e equal the leftmost $\lceil log_2 q \rceil$ bits of d_e. It computes $v_1 \equiv ew^{-1} \pmod q$, $v_2 \equiv rw^{-1} \pmod q$, $V = v_1G + v_2PU_s$, and $v \equiv x_V \pmod q$ where x_V is x coordinate of point V. At last, it compares v to r to verify whether d_1 is authentic.

C. Public Key Distribution Protocol

To bootstrap the broadcast authentication, receivers in R_s should get the public key PU_s from sender s. PU_s's distribution should be in an authenticated manner because adversaries may inject forged PU_s [25]. Furthermore, PU_s's distribution should introduce low overhead to resource constrained sensor nodes [26]. With these in mind, we propose to distribute PU_s through a certificate described in definition 3. The proposed

public key distribution protocol is implemented by three steps as follows, where base station bs acts as CA because it cannot be compromised by adversaries. Note that sensor nodes are preloaded with base station bs's public key PU_{bs}.

1) Sender s sends a request RE_s to base station bs and asks for generation of a certificate. The request RE_s contains sender's identity ID_s and public key PU_s, i.e., $RE_s = ID_s||PU_s$.
2) Base station bs replies with a certificate C_s, which contains the certificate's identity ID_c and request RE_s, signed by base station bs with its private key PR_{bs}. The whole expression is $C_s = ID_c||ID_s||PU_s||E(PR_{bs}, ID_c||ID_s||PU_s)$.
3) Sender s broadcasts C_s to all receivers in R_s.

The public key distribution protocol is efficient for WSNs. Public key PU_s included in C_s is distributed through broadcast (step 3). Certificate C_s remains valid over a long period unless it is explicitly revoked by base station bs. It means that private key PR_s corresponding to PU_s in C_s can be used to sign many chains of extended blocks. Moreover, if the intended receivers are changed without revocation of C_s, just rebroadcasting C_s is required. Note that receivers do not need to contact CA throughout the protocol.

The public key distribution protocol may undergo attacks that adversaries mislead traffic or jam the communication to the CA. If there are multiple CAs, i.e., more than one base station, the effect of the attacks will be relieved to an extent. Alternatively, Kong *et al*. [27] proposed to exploit sensor nodes to achieve multiple CAs.

IV. Analysis of Overhead and Security

In this section, we analyze the proposed PKC based broadcast authentication scheme using signature amortization in terms of overhead and security.

A. Overhead

The overhead of the proposed scheme is analyzed according to the following aspects.

- Computation overhead: average time to authenticate one broadcast message in M at the sender side and the receiver side.
- Communication overhead: average authenticator size per broadcast message in M.
- Authentication delay: time delay to authenticate an extended block at the receiver side.
- Memory overhead: memory size to buffer all extended blocks at the sender side and memory size required to authenticate one extended block at the receiver side.

In the computation and communication overhead, we calculate the average overhead relative to one broadcast message in order to facilitate the comparison with other broadcast authentication schemes. Note that the overhead of the public key distribution protocol is not included in the following analysis since it is trivial compared to authenticating broadcast messages.

1) Computation Overhead: Suppose the time to generate an ECDSA signature, to verify an ECDSA signature and to compute a collision resistant hash is denoted by T_{sign}, T_{verify} and T_{hash}, respectively.

The average time to authenticate one broadcast message in M at the sender side is T_s as shown in equation (5). The asymptotic upper bound of T_s is $O(T_{hash}/b)$ given that $n = kb$.

$$T_s = \frac{T_{sign} + kT_{hash}}{n} \tag{5}$$

The average time to authenticate one broadcast message at the receiver side is T_r as shown in equation (6). The asymptotic upper bound of T_r is $O(T_{hash}/b)$ given that $n = kb$.

$$T_r = \frac{T_{verify} + kT_{hash}}{n} \tag{6}$$

2) Communication Overhead: Suppose lengths of an ECDSA signature and a digest are denoted by L_{sig} and L_d, respectively. The average authenticator size per broadcast message is L_{au} as shown in equation (7). The asymptotic upper bound of L_{au} is $O(L_d/b)$ given that $n = kb$.

$$L_{au} = \frac{L_{sig} + (k+1)L_d}{n} \tag{7}$$

3) Authentication Delay: There is no delay to authenticate the first extended block EB_0 at the receiver side because it is authenticated by an ECDSA signature. Authentication delay of an extended blocks in EB^* is decided by the arrival time of the first broadcast message and last broadcast message of the extended block. Suppose the arrival time of the first broadcast message and last broadcast message of extended block EB_i, $1 \leq i \leq k$, is denoted by $t_{(i-1)b+1}$ and t_{ib}, respectively. The time delay to authenticate EB_i at the receiver side is T_i as shown in equation (8). It is obvious that T_i equals zero if $b = 1$, which means immediate authentication in lemma 2.

$$T_i = t_{ib} - t_{(i-1)b+1} \tag{8}$$

Lemma 2 (Immediate authentication)**.** *The proposed scheme achieves immediate authentication when an extended block in EB^* contains just one broadcast message in M.*

4) Memory Overhead: Suppose the size of a broadcast message in M is denoted by L_m.

The memory size for buffering all extended blocks at the sender side is LM_s as shown in equation (9).

$$LM_s = L_{sig} + nL_m + (k+1)L_d \tag{9}$$

The first extended block is authenticated by an ECDSA signature so there is no need for receivers to buffer it. Memory size required to authenticate an extended block in EB^* at the receiver side is LM_r as shown in equation (10), where $2L_d$ refers to one digest in the previous extended block and one digest in the current extended block.

$$LM_r = bL_m + 2L_d \tag{10}$$

The memory overhead is affordable to resource constrained sensor nodes in general. This is because a sensor node has a large external storage equipment although the RAM of a sensor node is relatively small. For instance, a TelosB sensor node is equipped with a measurement serial flash of 1024K

bytes. Therefore, if extended blocks are beyond the capability of the RAM at the sender side, they can be buffered in an external storage equipment on a temporary basis.

At last, we conclude the overhead analysis with a theorem below.

Theorem 3 (Low overhead)**.** *The computation and communication overhead of the proposed scheme is to the same degree of HMAC.*

Proof: The computation overhead of the proposed scheme at the sender side and the receiver side is the same, $O(T_{hash}/b)$. The communication overhead of the proposed scheme is $O(L_d/b)$. HMAC is computed by $MAC(K,m) = H((K_0 \oplus opad)||H((K_0 \oplus ipad)||m))$ according to [28]. K is a secret key, and after necessary pre-processing, it is transformed to K_0. $opad$ is the outer pad. $ipad$ is the inner pad. H denotes a collision resistant hash. Thus, the computation overhead of HMAC at the sender side and the receiver side is also the same, $2T_{hash}$. The communication overhead of HMAC is L_d. ■

B. Security

The proposed scheme is on the basis of PKC so it should inherently be resilient to node compromise attacks and provide non-repudiation. Moreover, it offers as strong authenticity as conventional PKC based broadcast authentication schemes provide.

Lemma 4 (Resilience to node compromise attacks)**.** *The proposed scheme is resilient to node compromise attacks.*

Proof: The information sensitive to broadcast authentication in a receiver is PU_{bs}, PU_s and a digest, say, d_i in EB_{i-1}. They will be disclosed to an adversary if any receiver is compromised. However, with PU_{bs}, the adversary cannot create a valid certificate. With PU_s, an adversary cannot generate the signature for EB_0 either. With d_i, an adversary cannot forge EB_i because H is a collision resistant hash. In short, an adversary cannot exploit a compromised sensor node to impersonate a valid sender. ■

Lemma 5 (Non-repudiation)**.** *Sender s cannot deny that messages in M are broadcast by itself.*

Proof: The public key distribution protocol guarantees authenticity of public key PU_s. Corresponding to PU_s, private key PR_s is held by sender s. Hence, only sender s has the right to create extended blocks. If all extended blocks are buffered by a receiver, the receiver can naturally prove the identity of sender s. Also, a receiver can identify sender s with EB_0 because it is computationally infeasible for an adversary to generate correct digest d_1 in EB_0 with forged messages. ■

Lemma 6. *The authenticity provided by the proposed scheme is equivalent to each extended block in EB is authenticated by an ECDSA signature.*

Proof: The extended blocks in EB can be divided into two groups: EB_0 and the extended blocks in EB^*. EB_0 contains an ECDSA signature to authenticate d_1. The extended block in EB^* is authenticated by the digest in its previous extended block. Without loss of generality, we prove that the authenticity of EB_i, $1 \leq i \leq k$, is equivalent to authentication by an ECDSA signature. As shown in equation (4), EB_i is the input to compute d_i in EB_{i-1}. Likewise, EB_{i-1} is the input to compute d_{i-1} in EB_{i-2}. Finally, EB_1 is the input to compute d_1 in EB_0. Because the computation in each step is conducted with a collision resistant hash, it is computationally infeasible for an adversary to forge EB_i without changing d_1. In a word, changing EB_i results in that the verification of d_1 in EB_0 with the ECDSA signature fails. ■

Theorem 7 (Strong authenticity)**.** *The proposed scheme is as secure as conventional PKC based broadcast authentication schemes.*

Proof: In the proposed scheme, we do not arbitrarily reduce the length of keys in order to meet the low overhead required by resource constrained sensor nodes. Moreover, authenticity of each extended block is equivalent to being authenticated by an ECDSA signature. Hence, the proposed scheme retains as high security as conventional PKC based broadcast authentication schemes offer. ■

V. Implementation and Experimental Results

We conduct experiments on a multi-hop testbed consisting of TelosB sensor nodes. Using the testbed, we study the performance of the proposed scheme. For simplicity, we have considered a multi-hop testbed consisting of several sensor nodes. Since this is a small set of a real life WSN, we believe that our experimental results can also be applied to the network with hundreds of sensor nodes. For comparison, we also implement a message oriented signature scheme which represents conventional PKC based broadcast authentication schemes and HMAC which represents conventional symmetric key cryptography based broadcast authentication schemes.

A. Implementation

The testbed with the maximum hop of three consists of seven TeslosB sensor nodes [29]. Microcontroller of TelosB is 16-bit 8 MHz MSP430 that has 10K bytes RAM and 48K bytes ROM. RF transceiver is CC2420 whose maximum data rate is 250 kbps and frequency band is 2400 MHz. The topology of the testbed as shown in Fig. 2 is formed by filtering messages on each sensor node that stores the addresses of its neighbors. The dashed lines in Fig. 2 indicate that two sensor nodes are connected. For example, when sensor node 2 broadcasts a message, the sender and receivers 1, 3, and 4 can receive it. In the testbed, one sensor node acts as the sender generating and broadcasting extended blocks. The broadcast extended blocks are verified by other six sensor nodes that act as receivers.

Both sides-program (sender and receiver) of the proposed scheme are implemented by NesC on TinyOS 2.1. The ECDSA signature program is improved from TinyECC [30]. The domain paraments of the ECDSA signature are secp160r1 defined by [31]. Thus, the signature size is 44 bytes. The collision resistant hash adopts 64-bit truncation of SHA-1, which has been used previously (e.g., [32]).

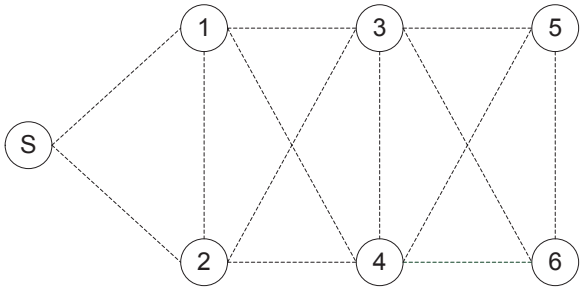


Fig. 2. Topology of the testbed (S: sender; 1, 2, 3, 4, 5, 6: receivers.)

The sender broadcasts 100 messages in total, which is enough to exhibit the trend of the overhead. Sending rate of the sender satisfies the Poisson distribution with average value of 0.2 message per second. After receiving broadcast messages, receivers rebroadcast them after a random delay within 2 seconds. The backoff for acknowledgements and replies is also 2 seconds. Each broadcast message has a payload of 20 bytes. To observe the influence of the size of an extended block, we conduct experiments with $b = 1, 2, 5, 10$, which are denoted by SA1, SA2, SA5 and SA10, respectively.

For the purpose of comparison, we also has implemented HMAC [28] and a message oriented signature scheme denoted by SIGS. Messages in SIGS are signed and verified by an ECDSA signature whose domain paraments is secp160r1. HMAC uses 64-bit truncation of SHA-1 to produce a digest.

B. Experimental Results

The experimental results are presented in terms of computation overhead, communication overhead, authentication delay and memory overhead.

1) Computation Overhead: For the purpose of comparison, the computation overhead of the proposed scheme is measured by the average execution time to authenticate one broadcast message at the sender side and the receiver side. Because the average execution time changes with different number of messages, we summarize results of 20, 40, 60, 80 and 100 broadcast messages, respectively. The average execution time at the sender side is shown in Fig. 3. SIGS constantly consumes more than 3000 ms to sign a broadcast message. The heavy overhead of SIGS is effectively amortized over broadcast messages in the proposed scheme. Moreover, the effect became more manifest as the number of broadcast messages increases. When the number of broadcast messages is 100, the average execution time of SA1, SA2, SA5 and SA10 reduces to 60 ms. Fig. 4 shows the average execution time at the receiver side. To focus on the comparison between the proposed scheme with HMAC, the overhead of SIGS that is fixed more than 3900 ms is excluded from the figure. Execution time of SA1, SA2, SA5 and SA10 approaches that of HMAC which is 20 ms constantly, as the number of messages increases. Although SA1 takes a little more time than SA2, SA5 and SA10 to authenticate one broadcast message at the receiver side, the difference is trivial. That means the size of an extended block has little effect on the execution time.

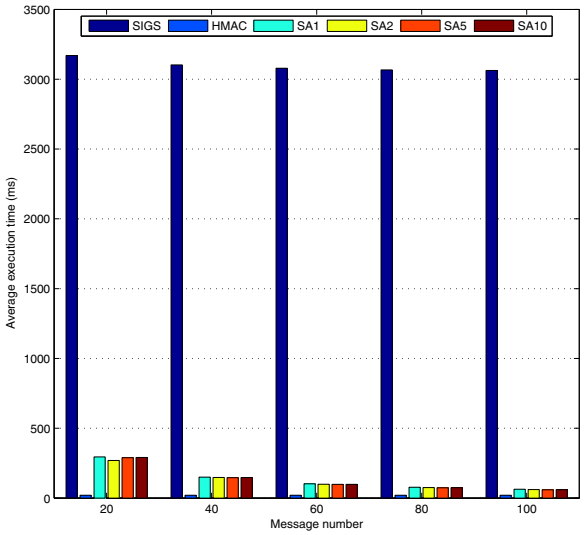


Fig. 3. Average execution time to authenticate one broadcast message at the sender side (SIGS: message oriented signature; HMAC: keyed-Hash MAC; SA1: proposed scheme with $b = 1$; SA2: proposed scheme with $b = 2$; SA5: proposed scheme with $b = 5$; SA10: proposed scheme with $b = 10$).

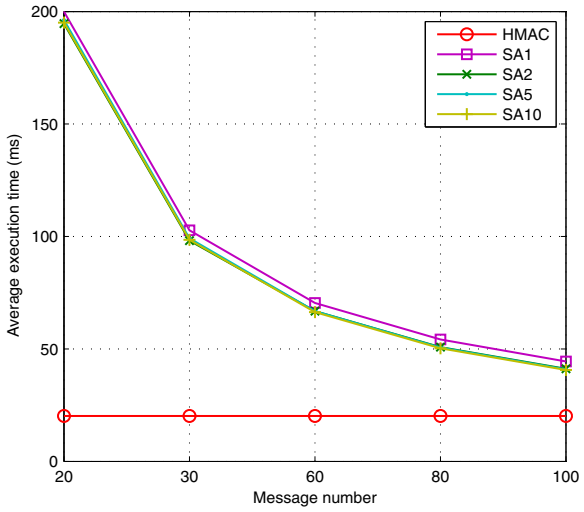


Fig. 4. Average execution time to authenticate one broadcast message at the receiver side (HMAC: keyed-Hash MAC; SA1: proposed scheme with $b = 1$; SA2: proposed scheme with $b = 2$; SA5: proposed scheme with $b = 5$; SA10: proposed scheme with $b = 10$).

2) Communication Overhead: First, the computation overhead of the proposed scheme is measured by average authenticator size per broadcast message. Because the average authenticator size changes with different number of messages, we summarize results of 20, 40, 60, 80 and 100 broadcast messages, respectively. Fig. 5 shows the average authenticator size per broadcast messages. The heavy communication overhead of SIGS, which is 44 bytes, is amortized over broadcast messages in the proposed scheme. The average authenticator size of SA1 is a little more than that of HMAC which is 8 bytes constantly but it approaches 8 bytes with the number of broadcast messages increasing. The average authenticator sizes of SA2, SA5 and SA10 are less than that of HMAC. Another observation is that the average authenticator size becomes small if an extended block contains more broadcast messages.

Second, the communication overhead is measured by the total messages versus the probability of message loss, as

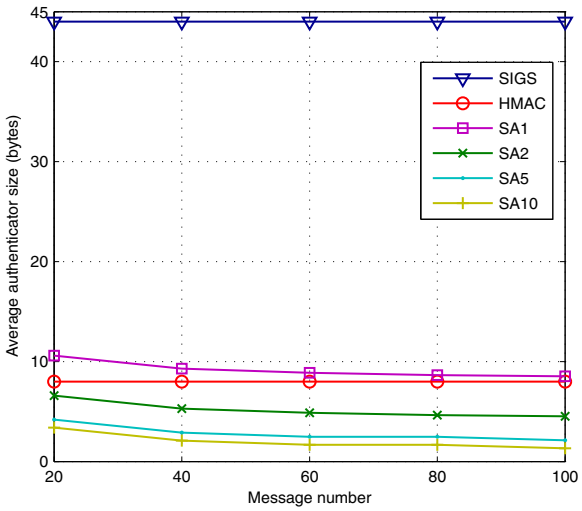


Fig. 5. Average authenticator size per broadcast message (SIGS: message oriented signature; HMAC: keyed-Hash MAC; SA1: proposed scheme with $b = 1$; SA2: proposed scheme with $b = 2$; SA5: proposed scheme with $b = 5$; SA10: proposed scheme with $b = 10$).

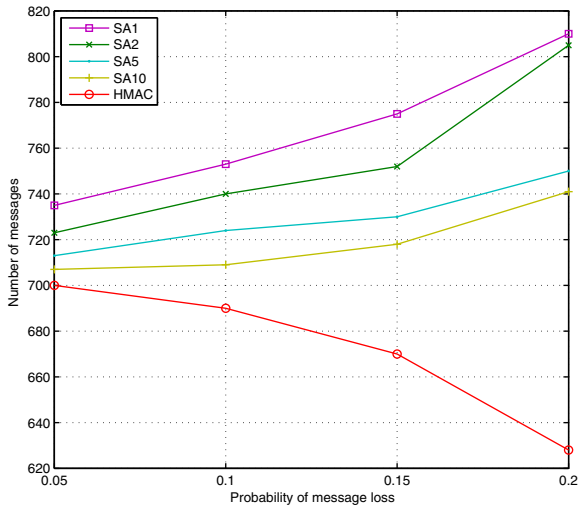


Fig. 6. Total messages versus probability of message loss (HMAC: keyed-Hash MAC; SA1: proposed scheme with $b = 1$; SA2: proposed scheme with $b = 2$; SA5: proposed scheme with $b = 5$; SA10: proposed scheme with $b = 10$).

shown in Fig. 6. The total messages for SA1, SA2, SA5 and SA10 include broadcast messages of extended blocks, acknowledgements and replies. Recalling that there are six receivers in the testbed and the sender broadcasts 100 messages in total, the total messages for SA1, SA2, SA5 and SA10 are more than 700. Obviously, SA10 needs the least acknowledgements and replies to offer the reliable broadcast, for instance, 41 acknowledgements and replies when the probability of message loss is 0.2. One reason is it leaves more time for receivers exchanging messages belonging to current extended block. The other reason is one acknowledgement can specify 10 lost messages at most. Note that the acknowledgement for SA10 is 9 bits larger than that for SA1. Each receiver in HMAC lacks about 10 messages on average when the probability of message loss is 0.2. If each lost message is retrieved by one acknowledgement and one reply. It would cause 120 transmissions.

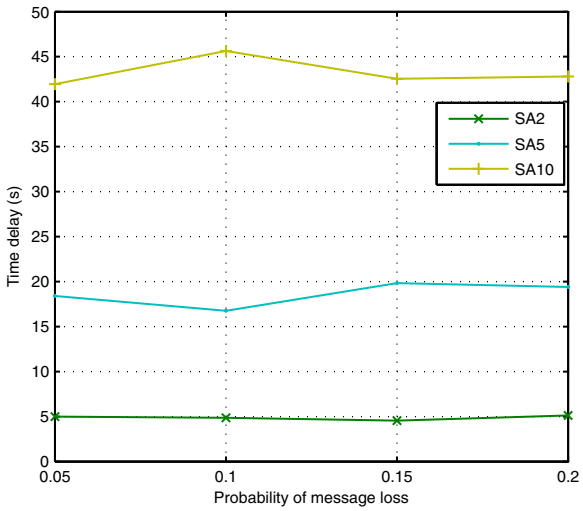


Fig. 7. Time delay to authenticate an extended block (SA2: proposed scheme with $b = 2$; SA5: proposed scheme with $b = 5$; SA10: proposed scheme with $b = 10$).

TABLE I
MEMORY OVERHEAD AT THE SENDER SIDE (BYTES)

Type	SIGS	HMAC	SA1	SA2	SA5	SA10
ROM	24650	14748	30084	30158	30176	30184
RAM	2433	657	5950	5550	5310	5230

3) Authentication Delay: The authentication delay is measured by the time delay to authenticate an extended block at the receiver side. The time delay for SA2, SA5 and SA10 is shown in Fig. 7. SA1 is not included in the figure because it achieves immediate authentication. The sending rate at the sender side meets the Poisson distribution with the average value of 0.2 message per second. The result is averaged on 5 different runs. Apparently, the time delay in Fig. 7 is determined by the size of an extended block. SA2 has the least time delay and SA10 has the largest time delay. The time delay does not increase as the probability of message loss becomes larger. The reason is that most message loss occurs between the sender and its one-hop receivers, receivers 1 and 2, due to only one sender. The message loss increases the authentication delay of the current extended block for receivers 1 and 2. However, the authentication delay for other receivers decreases when receivers 1 and 2 broadcast the current extended block.

4) Memory Overhead: Memory overhead of the proposed scheme is measured by the code size and the RAM size at the sender side and the receiver side. The code size and the RAM size at the sender side are shown in table I. The experimental results at the receiver side are shown in table II. The RAM sizes of SA1, SA2, SA5 and SA10 at the sender side are about 3K bytes more than those at the receiver side since the sender buffers all extended blocks while the receiver buffers one extended block. Note that data in both tables include additional TinyOS modules that are used for collecting experimental results. Actually, the code size of the bare HMAC module is about 1.4K bytes actually.

VI. CONCLUSION

In this paper, we proposed a novel PKC based broadcast authentication scheme using signature amortization for WSNs.

TABLE II
MEMORY OVERHEAD AT THE RECEIVER SIDE (BYTES)

Type	SIGS	HMAC	SA1	SA2	SA5	SA10
ROM	24490	14676	25042	25060	25106	25106
RAM	2197	532	2681	2701	2761	2861

The proposed scheme employs only one ECDSA signature to authenticate all broadcast messages. The overhead of the signature is amortized over all broadcast messages. Besides low overhead, the proposed scheme retains high security as strong as conventional PKC based broadcast authentication schemes. Moreover, the proposed scheme overcomes defects of μTESLA, that is, it does not require time synchronization, has an efficient public key distribution protocol and can achieve immediate authentication. Experimental results show that the overhead of the proposed scheme is to the same degree of HMAC.

ACKNOWLEDGMENT

This work has been partially supported by Grand-in-Aid for Scientific Research from Japan Society for Promotion of Science (JSPS) (no. 21300019 and 21650013), and Research Collaboration Grant from NII of Japan.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 102–114, 2002.

[2] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Trans. Networking*, vol. 11, no. 1, pp. 2–16, 2003.

[3] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proc. 2004 ACM SenSys*, pp. 81–94.

[4] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. 2000 Hawaii International Conference on System Sciences*, vol. 2, 2000.

[5] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus, "TinyPK: securing sensor networks with public key technology," in *Proc. 2004 ACM SASN*, pp. 59–64.

[6] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," *Ad Hoc Networks*, vol. 1, no. 2-3, pp. 293–315, 2003.

[7] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.

[8] D. Liu and P. Ning, "Multilevel μTESLA: broadcast authentication for distributed sensor networks," *ACM Trans. Embeded Computing Syst.*, vol. 3, no. 4, pp. 800–836, 2004.

[9] Y. Zhou and Y. Fang, "BABRA: batch-based broadcast authentication in wireless sensor networks," in *Proc. 2006 IEEE GLOBECOM*, pp. 1–5.

[10] P. Ning, A. Liu, and W. Du, "Mitigating DoS attacks against broadcast authentication in wireless sensor networks," *ACM Trans. Sensor Networking*, vol. 4, no. 1, pp. 1–35, 2008.

[11] T. Kwon and J. Hong, "Secure and efficient broadcast authentication in wireless sensor networks," *IEEE Trans. Computers*, vol. 59, no. 8, pp. 1120–1133, 2010.

[12] K. Ren, W. Lou, K. Zeng, and P. Moran, "On broadcast authentication in wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 6, no. 11, pp. 4136–4144, 2007.

[13] X. Du, M. Guizani, Y. Xiao, and H.-H. Chen, "Defending DoS attacks on broadcast authentication in wireless sensor networks," in *Proc. 2008 IEEE ICC*, pp. 1653–1657.

[14] R. Gennaro and P. Rohatgi, "How to sign digital streams," in *Advances in Cryptology-CRYPTO, Lecture Notes in Computer Science*, 1997, vol. 1294, pp. 180–197.

[15] C. K. Wong and S. Lam, "Digital signatures for flows and multicasts," *IEEE/ACM Trans. Networking*, vol. 7, no. 4, pp. 502–513, 1999.

[16] A. Perrig, R. Canetti, J. Tygar, and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," in *Proc. 2000 IEEE Symposium on Security and Privacy*, pp. 56–73.

[17] S. Miner and J. Staddon, "Graph-based authentication of digital streams," in *Proc. 2001 IEEE Symposium on Security and Privacy*, pp. 232–246.

[18] J. M. Park, E. Chong, and H. Siegel, "Efficient multicast packet authentication using signature amortization," in *Proc. 2002 IEEE Symposium on Security and Privacy*, pp. 227–240.

[19] A. Pannetrat and R. Molva, "Efficient multicast packet authentication," in *2003 Network and Distributed System Security Symposium*.

[20] X. Du, M. Guizani, Y. Xiao, and H.-H. Chen, "Two tier secure routing protocol for heterogeneous sensor networks," *IEEE Trans. Wireless Commun.*, vol. 6, no. 9, pp. 3395–3401, 2007.

[21] ——, "A routing-driven elliptic curve cryptography based key management scheme for heterogeneous sensor networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 3, pp. 1223–1229, 2009.

[22] H. Wang and Q. Li, "Efficient implementation of public key cryptosystems on mote sensors," in *Information and Communication Security, Lecture Notes in Computer Science*, 2006, vol. 4307, pp. 519–528.

[23] K. M. Finnigin, B. E. Mullins, R. A. Raines, and H. B. Potoczny, "Cryptanalysis of an elliptic curve cryptosystem for wireless sensor networks," *Int'l J. Security and Networks*, vol. 2, no. 3, pp. 260–271, 2007.

[24] Certicom Research, *Standards for Efficient Cryptography: Elliptic Curve Cryptography*, 2nd edition, 2009.

[25] H. Chen, Y. Xiao, X. Hong, F. Hu, and J. L. Xie, "A survey of anonymity in wireless communication systems," *Security and Commun. Networks*, vol. 2, no. 5, pp. 427–444, 2009.

[26] Y. Xiao, V. K. Rayi, B. Sun, X. Du, F. Hu, and M. Galloway, "A survey of key management schemes in wireless sensor networks," *Computer Commun.*, vol. 30, no. 11, pp. 2314–2341, 2007.

[27] Y. Kong, J. Deng, and S. Tate, "A distributed public key caching scheme in large wireless networks," in *Proc. 2010 IEEE GLOBECOM*, pp. 1–5.

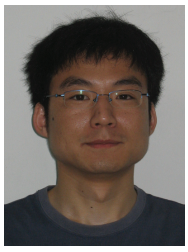
[28] National Institute of Standards and Technology, *The Keyed-Hash Message Authentication Code (HMAC), FIPS PUB 198*, 2002.

[29] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *Proc. 2005 IEEE IPSN*, pp. 364–369.

[30] A. Liu and P. Ning, "TinyECC: a configurable library for elliptic curve cryptography in wireless sensor networks," in *Proc. 2008 IEEE IPSN*, pp. 245–256.

[31] Certicom Research, *Standards for Efficient Cryptography: Recommended Elliptic Curve Domain Parameters*, 2000.

[32] A. Liu, P. Ning, and C. Wang, "Lightweight remote image management for secure code dissemination in wireless sensor networks," in *Proc. 2009 IEEE INFOCOM*, pp. 1242–1250.



Yongsheng Liu received the B.E. degree in computer science and engineering from Dalian University of Technology, Dalian, China, 2005, the M.E. degree in computer science from University of Science and Technology of China, Hefei, China, 2008, and is currently pursuing the Ph.D. degree in computer science at University of Tsukuba, Tsukuba Science City, Japan. His research interests include network security and wireless sensor networks.

Jie Li (M'96, SM'2004) is a Professor in Faculty of Engineering, Information and Systems, University of Tsukuba, Japan. He received the B.E. degree in computer science from Zhejiang University, Hangzhou, China, the M.E. degree in electronic engineering and communication systems from China Academy of Posts and Telecommunications, Beijing, China. He received the Dr. Eng. degree from the University of Electro-Communications, Tokyo, Japan. His research interests are in mobile distributed multimedia computing and networking, OS, network security, modeling and performance evaluation of information systems. He is a senior member of IEEE and ACM and a member of IPSJ (Information Processing Society of Japan). He has served as a secretary for Study Group on System Evaluation of IPSJ and on several editorial boards for the IPSJ Journal and so on, and on Steering Committees of the SIG of System EVAluation (EVA) of IPSJ, the SIG of DataBase System (DBS) of IPSJ, and the SIG of MoBiLe computing and ubiquitous communications of IPSJ. He has also served on the program committees for several international conferences such as IEEE ICDCS, IEEE INFOCOM, IEEE GLOBECOM, and IEEE MASS.

Mohsen Guizani (M'89, SM'99, F'09) is currently a Professor and the Associate Vice President for Graduate Studies at Qatar University, Qatar. He was the Chair of the CS Department at Western Michigan University from 2002 to 2006 and Chair of the CS Department at University of West Florida from 1999 to 2002. He also served in academic positions at the University of Missouri-Kansas City, University of Colorado-Boulder, Syracuse University and Kuwait University. He received his B.S. (with distinction) and M.S. degrees in Electrical Engineering; M.S. and Ph.D. degrees in Computer Engineering in 1984, 1986, 1987, and 1990, respectively, from Syracuse University, Syracuse, New York.

His research interests include Computer Networks, Wireless Communications and Mobile Computing, and Optical Networking. He currently serves on the editorial boards of six technical Journals and the Founder and EIC of *Wireless Communications and Mobile Computing* Journal published by John Wiley (http://www.interscience.wiley.com/jpages/1530-8669/). He is also the Founder and the Steering Committee Chair of the Annual International Conference of Wireless Communications and Mobile Computing (IWCMC). He is the author of seven books and more than 270 publications in refereed journals and conferences. He guest edited a number of special issues in IEEE Journals and Magazines. He also served as member, Chair, and General Chair of a number of conferences.

Dr. Guizani served as the Chair of IEEE Communications Society Wireless Technical Committee (WTC) and Chair of TAOS Technical Committee. He was an IEEE Computer Society Distinguished Lecturer from 2003 to 2005. Dr. Guizani is an IEEE Fellow and a Senior member of ACM.