Accelerating Signature-Based Broadcast Authentication for Wireless Sensor Networks

Xinxin Fan and Guang Gong

Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, N2L 3G1, Canada
{x5fan@engmail,ggong@calliope}.uwaterloo.ca

Abstract. In wireless sensor networks (WSNs), broadcast authentication is a crucial security mechanism that allows a multitude of legitimate users to join in and disseminate messages into the networks in a dynamic and authenticated way. During the past few years, several public-key based multi-user broadcast authentication schemes have been proposed to achieve immediate authentication and to address the security vulnerability intrinsic to μTESLA-like schemes. Unfortunately, the relatively slow signature verification in signature-based broadcast authentication has also incurred a series of problems such as high energy consumption and long verification delay. In this contribution, we propose an efficient technique to accelerate the signature verification in WSNs through the **cooperation** among sensor nodes. By allowing some sensor nodes to **release the intermediate computation results** to their neighbors during the signature verification, a large number of sensor nodes can accelerate their signature verification process significantly. When applying our faster signature verification technique to the broadcast authentication in a 4×4 grid-based WSN, a quantitative performance analysis shows that our scheme needs $17.7\% \sim 34.5\%$ less energy and runs about 50% faster than the traditional signature verification method.

Keywords: Wireless Sensor Networks, Security, Broadcast Authentication, Elliptic Curve Cryptosystems.

1 Introduction

Wireless senor networks (WSNs) are typically composed of a few powerful base stations and a large number of resource-constrained sensor nodes [1]. In WSNs, multi-user broadcast is an efficient and common communication paradigm, in which a host of network users will join in WSNs and disseminate messages (i.e., queries or commands) into the networks dramatically for obtaining the information of their interest [12,18,19]. Unfortunately, due to the nature of wireless communication in WSNs, adversaries can easily eavesdrop the traffic, impersonate other users, inject bogus data or alter the contents of legitimate messages during the multi-hop forwarding. Hence, authentication mechanisms need to be implemented to protect broadcast messages from various malicious attacks.

J. Zheng, D. Simplot-Ryl, V.C.M. Leung (Eds.): ADHOCNETS 2010, LNICST 49, pp. 328–343, 2010.

According to the cryptographic primitives employed, three categories of solutions have been proposed in literature for addressing broadcast authentication in WSNs. Earlier research mainly focused on designing symmetric-key based broadcast authentication schemes. Typical examples are μTESLA [17] and its variants [9,11,12], which provide source authentication and message integrity by utilizing one-way hash chains and delayed disclosure of authentication keys. μTESLA-like schemes provide efficient broadcast authentication mechanisms for WSNs in terms of computational overhead and energy consumption. However, the inherent features of μTESLA-like schemes, such as the need for (loose) time synchronization and the delayed authentication, have made them vulnerable to a variety of attacks [15,18,19]. Moreover, scalability is another concern for symmetric-key based solutions [18]. The second category of solutions achieve broadcast authentication through the use of one-time signatures [3,16]. Unlike μTESLA, one-time signature based solutions do not need the time synchronization and authentication is also immediate. Unfortunately, such schemes have some undesirable features such as large key sizes and a limited number of usages per key, which make them only suitable for applications with infrequent messages at unpredictable times [14].

Considering the security and scalability of symmetric-key based broadcast authentication schemes, a couple of public-key based solutions have been proposed during the past few years [4,18,19]. The main impetus for using public key cryptosystems in WSNs comes from the advances in the manufacturing technology of wireless sensor nodes as well as the efficient implementation of public key cryptographic algorithms on sensor platforms [10,20,21,22]. Employing public-key cryptography for implementing broadcast authentication in WSNs provides simple solutions, strong security resilience, good scalability and immediate message authentication, when compared to symmetric-key based solutions. However, public-key based broadcast authentication schemes have a common shortcoming: the signature verification is much slower than the message authentication code verification used in symmetric-key based solutions. As a result, a large number of packages might wait in a message queue of a senor node for signature verifications when many users broadcast messages.

In this paper we address the issue of speeding up the signature verification for public-key based multi-user broadcast authentication schemes in WSNs by exploiting the *cooperation* among sensor nodes. The basic idea is that some sensor nodes randomly *release their intermediate computation results* to their neighbors during the signature verification. Then many sensor nodes can use the received intermediate computation results to accelerate their signature verifications. To demonstrate the performance of the proposed technique, we conduct a case study for broadcast authentication in a 4×4 grid-based WSN. The detailed quantitative analysis shows our scheme is greatly superior to the traditional signature verification method in terms of energy consumption of the entire network.

The remainder of this paper is organized as follows: Section 2 gives a brief introduction about the elliptic curve digital signature algorithm (ECDSA). Section 3 presents the system model and adversary model. In Section 4, we

describe the acceleration technique for signature verification in WSNs and discuss the selection of system parameters. Section 5 analyzes the performance of the proposed scheme by a case study. Finally, Section 6 concludes this paper.

2 Elliptic Curve Digital Signature Algorithm (ECDSA)

Let $\mathbb{F}_q$ be a finite field with $q = p^m$ elements, where $p > 3$ is a prime and m is a positive integer. An *elliptic curve* $E(\mathbb{F}_q)$ is the set of solutions (x, y) over $\mathbb{F}_q$ satisfying an equation of the form $E : y^2 = x^3 + ax + b$, where $a, b \in \mathbb{F}_q$ and $4a^3 + 27b^2 \in \mathbb{F}_q^*$, together with an additional *point at infinity*, denoted by $\mathcal{O}$. The points on an elliptic curve form an (additive) Abelian group, where $\mathcal{O}$ is the identity element and the group operation is given by the well known chord-and-tangent rule. The ECDSA is a widely standardized variant of the ElGamal signature scheme, which is described as follows:

1. *System-wide parameters.* Let $\mathbb{G}$ be a cyclic subgroup of $E(\mathbb{F}_q)$ generated by the point P with prime order n and an identity element $\mathcal{O}$. Let $H : \{0,1\}^* \rightarrow \mathbb{Z}_n^*$ be a collision-resistant hash function.
2. *Initial set-up.* Singer A randomly selects an integer $d \in [1, n-1]$ and publishes its public key $Q = dP$. The parameter d is kept secret to A.
3. *Signature generation.* Singer A uses his/her private key d to generate a signature (r, s) for a message $M \in \{0,1\}^*$.
   (a) Select a random integer $k \in [1, n-1]$, compute $R = kP$ and set r to be the x-coordinate of R.
   (b) Compute $s = k^{-1}(e + dr) \bmod n$, where $e = H(M)$.
   (c) If $r, s \in [1, n-1]$, return (r, s); otherwise, go to Step (3-a).
4. *Signature verification.* Upon receiving the message $M \in \{0,1\}^*$ and the signature (r, s) from A, a verifier B verifies the signature using A's public key Q.
   (a) Check that $r, s \in [1, n-1]$. If any verification fails, return '**reject signature**'.
   (b) Compute $R' = s^{-1}(eP + rQ)$, where $e = H(M)$.
   (c) Check that the x-coordinate of R' is equal to r. If verification succeeds, return '**accept signature**'; otherwise, return '**reject signature**'.

Note that like most ElGamal signature schemes, the signature verification of ECDSA is about twice as slow as signature generation, which is an undesirable property when using ECDSA for multi-user broadcast authentication in WSNs.

3 System and Adversary Models

System Model: We consider a large-scale WSN consisting of a base station and a large number of sensor nodes. While the base station is powerful enough to execute various complicated operations, the sensor nodes are usually have constrained resources in terms of computational capabilities, memory, bandwidth,

and power supply. We assume that users need to register and obtain necessary credentials for using the broadcast service in the WSN. We also assume that the base station is always trustworthy and the sensor nodes can be completely captured and manipulated by adversaries. Furthermore, both the base station and users may broadcast messages into the network. We further assume that a single-chip 2.4GHz IEEE 802.15.4 compliant RF transceiver [23] is used as the wireless transmission module in sensor nodes, which supports up to 102-byte payload and thus provides enough space to contain both the broadcast message and its digital signature in one package. Finally, we assume that the loose clock synchronization is available in the WSN.

Adversary Model: We assume that an adversary can launch a wide range of attacks against the signature-based broadcast authentication schemes. For example, the attacker can simply mount the DoS attacks by injecting bogus messages into the network, aiming at exhausting the limited storage and energy of sensor nodes. We assume that some efficient pre-authentication techniques like those in [7,15] have been employed in the network to mitigate DoS attacks. Moreover, it is also possible that an adversary attempts to impersonate other legitimate sensor nodes and obtains valuable information through eavesdropping, modifying, deleting, replaying, forging or blocking any network traffic. We also assume that a small fraction of user devices and sensor nodes can be compromised by an adversary and therefore the attacker can manipulate compromised devices to disseminate messages into the network. Some efficient private-key protection mechanism and user revocation scheme [4,19] can be used to thwart the potential node compromise attack in WSNs.

4 Faster Signature Verification in WSNs

In this section, we first describe broadcast authentication in WSNs and then we use the ECDSA as an example to show how to accelerate the signature verification through the *cooperation* among sensor nodes.

4.1 Problem Statement

When WSNs are deployed in hostile environments, broadcast authentication (i.e., verify the authenticity of broadcast packages) is a crucial security mechanism to ensure the trustworthiness of network applications. After registration a user first contacts with several sensor nodes in the vicinity and sends a request for the broadcast service. Then the user and the sensor nodes conduct a mutual authentication procedure, which grants the access to the WSN only to a legitimate user and, at the same time, guarantees the user of the trustworthiness of the sensor nodes. As illustrated in Figure 1, once the user and the sensor nodes establish an authenticated channel, the user will sign a query or command and forward it to the sensor nodes (e.g., nodes A, B and C). Nodes A, B and C then verify the signature of the user, respectively. If the verification succeeds, they will locally broadcast the user's query/command (within their communication range).

When some node, say D, receives the broadcast package for the first time, it will execute the same signature verification and determine whether the received package should be forwarded to other nodes (e.g., node E). This broadcast and authentication procedure continues until all reachable nodes receive the user's broadcast package. If any verification fails during the broadcast, sensor nodes will drop the package and report to the base station.

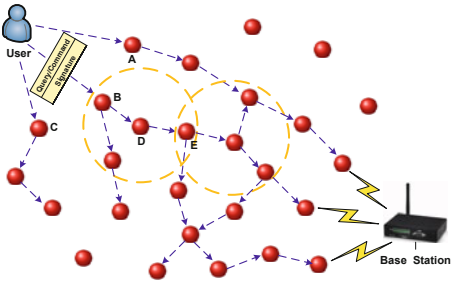

Fig. 1. User broadcast in wireless sensor networks. A broadcast package is usually forwarded multiple times through multi-hop communication.

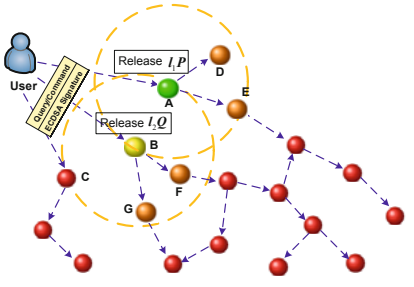

Fig. 2. Faster ECDSA digital signature verification. Nodes A and B release l_1P and l_2Q, respectively, which will significantly accelerate the signature verification of nodes D to G.

4.2 A Faster Signature Verification Scheme

In Figure 1, all sensor nodes execute the same signature verification after receiving a broadcast package. Assuming that the ECDSA is employed in WSNs, we show how to speed up the ECDSA signature verification below. Note that the proposed acceleration technique is also applicable to other public-key based multi-user broadcast authentication schemes such as those in [4,18,19].

For verifying an ECDSA signature, each node needs to calculate $R' = s^{-1}(eP + rQ) = l_1P + l_2Q$, where $e = H(M), l_1 = s^{-1}e$ and $l_2 = s^{-1}r$ (see Section 2). In other words, two scalar multiplications l_1P and l_2Q have to be computed by each node. Our acceleration technique comes from the following key observation: all sensor nodes independently execute the same signature verification procedure during the broadcast authentication. Therefore, if some sensor nodes would like to consume their energy to release some intermediate results, the signature verification of their neighboring nodes can be accelerated significantly. Moreover, the energy consumption of the entire network will be decreased as well. Our idea is clearly illustrated in Figure 2.

In Figure 2, a user's broadcast package $\langle M, r, s\rangle$ will be received by nodes A, B and C, where M denotes a user's query or command and (r, s) is the corresponding ECDSA signature of M. When all these three nodes finish the signature verification successfully, nodes A (the green node) and B (the yellow node) decide to release (i.e., locally broadcast) their intermediate computation results l_1P and l_2Q, respectively, whereas node C would like to keep silence.

By this means, nodes D and E (the orange nodes), which are the neighbors of node A, can fast verify the digital signature by performing an elliptic curve point addition $l_1P + l_2Q$, where l_2Q is computed by nodes D and E themselves and l_1P comes from the contribution of node A. Moreover, nodes F and G can also perform fast signature verification in a similar way. Hence, if some node in WSNs releases its intermediate computation result, all its neighbors can fast verify the digital signature by calculating one scalar multiplication and one elliptic curve point addition, which can achieve about 50% performance improvement as compared to the traditional signature verification procedure. For the scenario described in Figure 2, the acceleration of signature verification on nodes D to G benefits from the release of the intermediate computation results from nodes A and B. Note that some sensor nodes might receive both intermediate computation results l_1P and l_2Q from their neighboring nodes. However, the sensor nodes cannot use both received l_1P and l_2Q to fast verify the signature with one elliptic curve point addition. The reason is that an adversary can capture a sensor node and easily launch the following attack:

Step 1. The attacker generates a bogus message $\widehat{M}$;
Step 2. The attacker randomly chooses two integers $l_1', l_2' \in [1, n-1]$ and calculates $\widehat{R} = l_1'P + l_2'Q$;
Step 3. The attacker takes the x-coordinate $\hat{r}$ of $\widehat{R}$ together with some random integer $\hat{s} \in [1, n-1]$ to form the fake signature pair $(\hat{r}, \hat{s})$;
Step 4. The attacker successively releases two bogus broadcast packages $\langle \widehat{M}, \hat{r}, \hat{s}, l_1'P \rangle$ and $\langle \widehat{M}, \hat{r}, \hat{s}, l_2'Q \rangle$ to its neighbors;
Step 5. The victims compute $l_1'P + l_2'Q$ and compare the x-coordinate of the result with $\hat{r}$. As a result, the victims accept $\widehat{M}$ as a valid message.

Hence, if sensor nodes use two received intermediate computation results to verify a signature, they might accept any bogus broadcast messages from an attacker. To avoid the above attack, we only allow sensor nodes to use at most one intermediate result (i.e., l_1P or l_2Q) from their neighboring nodes for signature verification. Moreover, for the sake of simplicity of presentation, we further assume that if some sensor nodes release their intermediate computation results they will release l_2Q in the rest of this paper. We first illustrate a basic scheme of our faster ECDSA signature verification procedure in Figure 3.

Let SCA and ADD denote the elliptic curve scalar multiplication and the elliptic curve point addition, respectively. In the basic scheme, a sensor node may receive a data package $\langle M, r, s \rangle$ or $\langle M, r, s, l_2Q \rangle$. If a fresh package $\langle M, r, s \rangle$ is received, the sensor node will first compute l_1P and then wait for a very short time period τ to see whether it can obtain useful information from its neighbors for accelerating the signature verification. If it is, the node can finish the signature verification with 1 SCA + 1 ADD. Otherwise, the node will complete the verification itself with 2 SCA + 1 ADD after the time period τ. On the other hand, if a fresh package $\langle M, r, s, l_2Q \rangle$ is received, the sensor node will first calculate l_1P and then perform a fast signature verification with 1 SCA + 1 ADD. For the above two cases, if the signature is verified successfully, the sensor

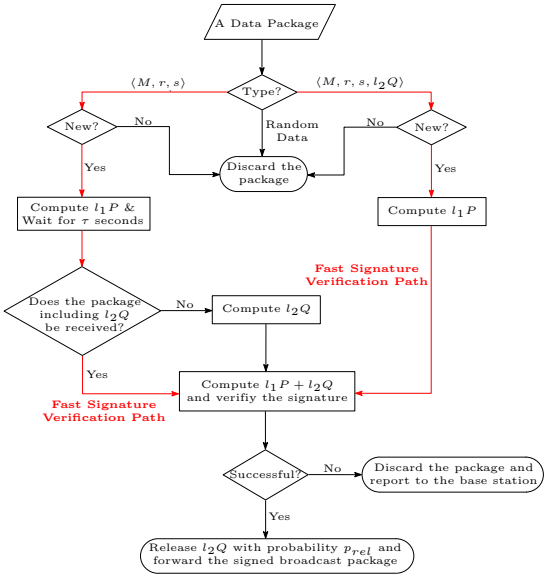

Fig. 3. Faster ECDSA Signature Verification for WSNs (Basic Scheme)

node will continue forwarding the broadcast package to its neighbors. Moreover, the intermediate computation result l_2Q will also be released with probability p_{rel}. Otherwise, if the signature verification is failed, the sensor node will send a signed report to the base station. Once the base station receives enough reports from the network, it will perform appropriate security mechanisms (see [24] for an example) to identify compromised nodes in WSNs. Although the basic scheme is simple and efficient, it is still vulnerable to the following attack:

Step 1. The attacker generates a bogus message $\widehat{M}$;
Step 2. The attacker randomly chooses an integer $k' \in [1, n-1]$, computes $R' = k'P$ and sets r' to be the x-coordinate of R';
Step 3. The attacker randomly chooses an integer $s' \in [1, n-1]$ and computes $l'_2Q = R' - s'^{-1}e'P$, where $e' = H(\widehat{M})$;
Step 4. The attacker uses (r', s') as the signature of the message $\widehat{M}$ and releases the bogus broadcast package $\langle \widehat{M}, r', s', l'_2Q \rangle$ to its neighbors;
Step 5. The victims compute $l'_1P + l'_2Q$ and compare the x-coordinate of the result with $\hat{r}$. As a result, the victims accept $\widehat{M}$ as a valid message.

The above attack works because the attacker knows that all its neighbors will first calculate $l_1P = s^{-1}eP$ and then use the released value l_2Q to accelerate their signature verifications. Consequently, the attacker chooses random r', s' and the bogus message $\widehat{M}$, and forges the broadcast package $\langle \widehat{M}, r', s', l'_2Q \rangle$ that can pass the signature verification. Note that if sensor nodes use traditional ECDSA signature verification procedure (see Section 2) the bogus broadcast package $\langle \widehat{M}, r', s', l'_2Q \rangle$ cannot pass the verification since the released l'_2Q is equal to

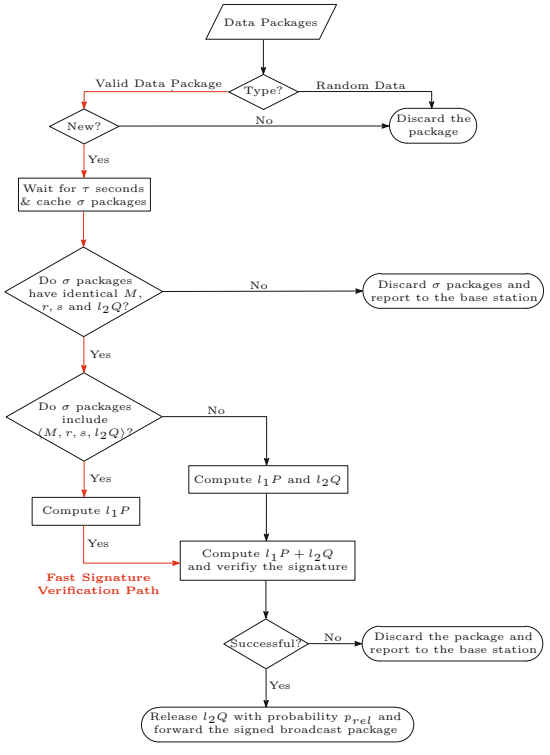

Fig. 4. Faster ECDSA Signature Verification for WSNs (Enhanced Scheme)

$s'^{-1}r'Q$ with negligible probability. To thwart the attack in the basic scheme, we propose an enhanced scheme as shown in Figure 4, which takes advantage of the redundance of broadcast packages in WSNs.

In the enhanced scheme, each sensor node first waits for τ seconds and caches σ data packages (i.e., $\langle M, r, s\rangle$ or $\langle M, r, s, l_2Q\rangle$) received from its neighbors, where τ and σ are selected such that the sensor node can receive at least one data package from an honest neighbor (see Section 4.3). The sensor node then checks whether the cached σ data packages have identical M, r, s and l_2Q. Note that the main goal of adversaries is to broadcast fake data packages into WSNs. While all honest nodes forward a correct data package $\langle M, r, s\rangle$ or $\langle M, r, s, l_2Q\rangle$ to their neighboring nodes, adversaries try to deceive their neighbors by broadcasting a bogus data package $\langle M', r', s', l'_2Q\rangle$ as described in the basic scheme. Hence, if the senor node finds that the received data packages have different M, r, s or l_2Q, it will report the potential attack to the base station immediately. On the other hand, if all the cached σ data packages have identical M, r, s and l_2Q, the sensor node will further check whether it has received useful data packages $\langle M, r, s, l_2Q\rangle$ for accelerating signature verification. If it is, the sensor node will calculate l_1P and then complete the signature verification with 1 SCA + 1 ADD. Otherwise, the sensor node will perform the traditional ECDSA signature verification with

2 SCA + 1 ADD. The remaining steps after the signature verification are the same as those in the basic scheme.

4.3 Selection of System Parameters

In this subsection, we provide the guidance for selecting the system-wide parameters τ, σ and p_{rel}.

Selection of the Delay τ and the Threshold σ. In the enhanced scheme, a sensor node needs to wait for τ seconds and cache σ data packages. We assume that on average a sensor node A has λ neighbors and half of them will broadcast data packages to A at a certain communication round[1]. We further assume that among A's $\lambda/2$ neighbors μ nodes can be compromised by adversaries and each of them can send at most ν bogus data packages to A during that communication round. Note that all compromised nodes must collude to send *identical* bogus data packages to A. Otherwise, A will discard all cached data packages and report to the base station. To thwart the collusive attacks from adversaries, the threshold σ should satisfy the following condition:

$$\lambda/2 \geq \sigma \geq \mu \cdot \nu + 1.$$

The above condition guarantees that the sensor node A can receive at least one correct broadcast package from an honest neighbor. As a result, A will not accept the bogus messages from collusive adversaries since all the catched data packages are not identical.

After the threshold σ is determined, we can choose the delay τ such that σ data packages can be received by the sensor node A. If a CC2420 IEEE 802.15.4 radio transceiver from Texas Instruments [23] is used in sensor nodes, the data transmission rate can achieve 250Kbps. We also assume that the signed broadcast package fits in the maximum allowable transmission limit (i.e., 128 bytes) of the CC2420 radio transceiver. So a 128-byte broadcast package will be transmitted in the physical layer and the transmission delay is about 4.096ms. Moreover, we also need to consider the CC2420 radio backoff that is a period of time where the radio pauses before attempting to transmit. Two backoff periods, namely initial backoff and congestion backoff, can be chosen for the CC2420 radio [23]. The initial backoff is $1 \sim 32$ backoff units[2] (i.e., 300μs - 10ms), whereas the congestion backoff is $1 \sim 8$ backoff units (i.e., 300μs - 2.5ms). Therefore, in the worst case a 128-byte broadcast package can be received by the sensor node A within around 17ms. Taking into account all these factors, the delay τ should satisfy the following condition:

$$\tau \geq 17 \cdot 10^{-3} \cdot \sigma.$$

[1] The procedure of broadcast authentication can be divided into a series of communication rounds. Although a sensor node A has λ neighbors, only half of them will broadcast packages to A on average at certain communication round and the other half will receive the broadcast package from A in the following communication round.

[2] The units of backoff are 10 jiffies (32KHz ticks).

Selection of the Release Probability p_{rel}. In the enhanced scheme, p_{rel} is the probability that a node will release its intermediate computation result, which is a predefined system parameter characterizing the trade-off between the verification speed of digital signature and the energy consumption of WSNs. Generally speaking, if a large value of p_{rel} is used, more sensor nodes will consume additional energy to broadcast their intermediate computation results and the signature verification for a large group of sensor nodes will be accelerated as a result, and vice versa. The selection of the release probability p_{rel} is closely related to the topology and the deployment of a WSN. Once the WSN is deployed, a network administrator first needs to analyze the network topology and estimates the average number of sensor nodes that might work on the signature verification during the transmission of a broadcast package. The administrator then determines a release probability p_{rel} that can achieve the optimal trade-off between the energy consumption of the network and the efficiency of the signature verification. More specifically, assuming that on average N sensor nodes work on the signature verification at each communication round, the probability that T sensor nodes will release their intermediate computation results is

$$p_T = \binom{N}{T} \cdot p_{rel}^T \cdot (1 - p_{rel})^{(N-T)}.$$

Let E_s, E_r, and E_{sca} be the energy consumption of sending and receiving one packet, and calculating one elliptic curve scalar multiplication on sensor nodes, respectively. Recall that we assume that each node has λ neighbors on average. Then we can roughly estimate the additional energy consumption/saving due to the use of our fast signature verification technique as follows:

- T sensor nodes will locally broadcast their intermediate computation results, the energy consumption of which is $T \cdot E_s$;
- About $\frac{\lambda T}{2}$ sensor nodes will receive the intermediate computation results, the energy consumption of which is $\frac{\lambda T}{2} \cdot E_r$;
- About $\frac{\lambda T}{2}$ sensor nodes will accelerate their signature verifications using the intermediate computation results, the energy saving of which is $\frac{\lambda T}{2} \cdot E_{sca}$.

Therefore, the expected additional energy consumption/saving will be

$$\overline{E} = \sum_{T=1}^{N} p_T \cdot \left(T \cdot E_s + \frac{\lambda T}{2} \cdot E_r - \frac{\lambda T}{2} \cdot E_{sca} \right). \quad (1)$$

Note that the value in the round bracket of Equation (1) might be positive or negative, which depends on the energy consumption of the microcontroller and the radio component on different sensor motes. If the above value is positive, we need to choose a release probability p_{rel} that can minimize the energy consumption $\overline{E}$. Otherwise, we select a p_{rel} that can maximize the energy saving $\overline{E}$. Here, we only provide the guidance about the selection of the release probability p_{rel} and omit the details of specific applications.

5 Security and Performance Analysis

In this section, we analyze the performance of the proposed acceleration technique for signature verification with respect to communication and computation overheads (in terms of energy consumption). The performance analysis focuses on a simple 4×4 grid-based WSN. We also compare our scheme with the traditional ECDSA signature verification when applied to the broadcast authentication in the 4×4 grid-based WSN.

5.1 Case Study

Note that the performance of our faster signature verification technique is closely related to the deployment of WSNs and the distribution of attackers in the network. To analyze the performance of our scheme, we conduct a case study for the broadcast authentication problem in a 4×4 grid-based WSN, as illustrated in Figure 5. In the grid-based sensor network, each node only can directly communicate with its one-hop neighbors. A user sends its signed broadcast package to the Node 1 at Round 0. After six communication rounds, the broadcast package will be received and verified by all sensor nodes. Furthermore, in our faster signature verification scheme we assume that one sensor node will release the intermediate computation result l_2Q in each communication round (see the green nodes $1, 2, 6, 7, 11$ and 12 in Figure 5).

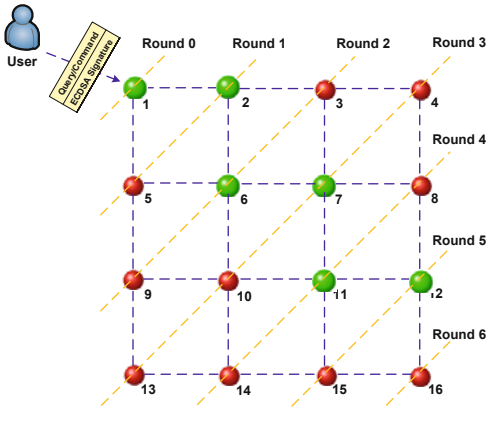

Fig. 5. Broadcast Authentication in a 4×4 grid-based WSN

To give a detailed quantitative analysis, we further assume that MICAz motes are used in the WSN. Under a typical configuration such as a 3V supply voltage and a 7.37MHz clock frequency, the MICAz mote draws a current of 12mA in an active mode (i.e., CPU is operating) [6]. Based on the formulae of calculating the energy consumption on MICAz motes [8], we obtain the following basic facts:

- A Chipcon CC2420 radio used in MICAz motes consumes $E_s = 83.6\mu$J and $E_r = 90.4\mu$J to transmit and receive l_2Q with 40 bytes[3], respectively;
- An Atmega128L microcontroller used in MICAz motes consumes about $E_{ver} = 22.68$mJ and $E_{sca} = 11.52$mJ to verify an ECDSA signature and compute a scalar multiplication on a 163-bit elliptic curve.

Note that we will not count the energy consumption of sensor nodes when sending and receiving the broadcast package throughout the whole network since it is the same for both faster and traditional signature verification schemes. We only compare the difference of both schemes in terms of communication and computation overhead in the next subsections.

5.2 Performance in the Ideal Case

In this subsection we analyze the performance of our faster signature verification technique in the ideal case (i.e., no adversaries exist), which can serve as the upper bound of performance improvement when applying our approach to broadcast authentication in the 4×4 grid-based WSN. In our scheme, some sensor nodes need to release their intermediate computation results in order to accelerate the signature verification for their neighboring nodes. Hence, our scheme consume more energy for transmitting the intermediate computation result l_2Q, when compared to using the traditional ECDSA signature verification in WSNs. In the 4×4 grid-based WSN (see Figure 5), the six green nodes (i.e., Nodes 1, 2, 6, 7, 11, 12) will locally broadcast their intermediate computation results to their one-hop neighbors, which causes an extra energy consumption of $6 \times 83.6\mu\text{J} = 501.6\mu$J in the network. Note that although some sensor nodes (e.g., Node 6) has four one-hop neighbors, only two of them (i.e., Nodes 7 and 10) will receive the intermediate computation results since the other two (i.e., Nodes 2 and 5) have finished the signature verification in the previous round (i.e., Round 1) and gone into the power-saving sleep mode. Therefore, there are totally 11 sensor nodes receiving the intermediate computation results, which causes an extra energy consumption of $11 \times 90.4\mu\text{J} = 994.4\mu$J in the WSN. In brief, our faster signature verification incurs an extra energy consumption of $501.6 + 994.4 = 1496\mu\text{J} \approx 1.5$mJ for transmitting (i.e., sending and receiving) the intermediate computation results for the WSN in question.

With respect to the computation aspect, it is not difficult to find that the signature verification on Nodes 2, 3, 5, 6, 7, 8, 10, 11, 12, 15 and 16 will be accelerated by 50% (i.e., saving one elliptic curve scalar multiplication) due to the use of the intermediate computation results from their neighboring nodes, which leads to a significant energy saving of $11 \times 11.52\text{mJ} = 126.72$mJ in the WSN, as compared to the traditional ECDSA signature verification technique. Moreover, the signed broadcast package will be forwarded to other sensor nodes only if the signature verification is successful. Therefore, the performance improvement on the signature verification will also reduce the transmission delay of the broadcast

[3] Assuming that a 160-bit elliptic curve cryptosystem is employed in WSNs, the size of l_2Q is around 40 bytes.

package by 50% in each round accordingly. Consequently, the service quality of the broadcast authentication in WSNs has been improved remarkably by using our faster signature verification technique.

To sum up, for the broadcast authentication in the target 4×4 grid-based WSN, our faster signature verification can save the energy consumption of $126.72\text{mJ} - 1.5\text{mJ} = 125.22\text{mJ}$ in total, considering both the communication and computation overheads. Therefore, using the proposed signature verification technique, one can save up to

$$\frac{125.22\text{mJ}}{16 \times 22.68\text{mJ}} \times 100\% = 34.5\%$$

energy consumption for the grid-based WSN in question.

5.3 Security and Performance under Attacks from Independent Adversaries

In this subsection, we analyze the security and performance of our scheme when there exist independent adversaries in the 4×4 grid-based WSN. To this end, we assume that two independent adversaries (i.e., 12.5% nodes are compromised and become malicious nodes.) are deployed in the WSN and each of them will broadcast a bogus package to its neighbors. Note that in the grid-based WSN each bogus package will be received by two sensor nodes in the following communication round. To maximize the influence of independent adversaries, we select Node 3 and Node 9 as two independent adversaries in the 4×4 grid-based WSN. Moreover, for those nodes (i.e., Nodes 2, 3, 4, 5, 9, 11) that can only receive one data package from its neighbors in certain communication rounds, they will verify the signature themselves in order to avoid the attack descried in the basic scheme. Other nodes (i.e., Nodes 6, 7, 8, 10, 11, 12, 15, 16) will cache two received data packages from their neighbors and decide whether they can perform faster signature verifications. Like the ideal case, we also assume that the six green nodes (i.e., Nodes 1, 2, 6, 7, 11, 12) will locally broadcast the intermediate computation results l_2Q to their neighbors.

Under the above settings, our faster signature verification still incurs an extra energy consumption of 1.5mJ for transmitting the intermediate computation results l_2Q like the ideal case. However, due to the existence of independent adversaries Node 3 and Node 9, Node 7 and Node 10 will receive two different data packages from their neighbors. Therefore, Node 7 and Node 10 have to verify the signature themselves instead of using the released l_2Q from Node 6 for faster verification. Consequently, the signature verification will be accelerated only for Nodes 6, 8, 11, 12, 15, 16 in this case, which can save the energy of $6 \times 11.52\text{mJ} = 69.12\text{mJ}$ in the WSN. Combining both communication and computation overheads, one can obtain that in the case of two independent adversaries the total energy saving is

$$\frac{69.12 - 1.5\text{mJ}}{14 \times 22.68\text{mJ}} \times 100\% = 21.3\%$$

for the 4×4 grid-based WSN. In addition, attacks from two independent adversaries have no effect on the security of our scheme and all the bogus data packages from independent adversaries are also discarded by legitimate nodes.

5.4 Security and Performance under Attacks from Collusive Adversaries

In this subsection, we analyze the security and performance of our scheme under the existence of collusive adversaries in the 4×4 grid-based WSN. Again, we assume that two collusive adversaries are deployed in the WSN and they will broadcast *identical* bogus data packages to their neighbors. To maximize the influence of collusive adversaries, we choose Node 7 and Node 10 as two collusive adversaries in the 4×4 grid-based WSN. Furthermore, we use the same assumptions as the case of independent adversaries for other nodes. Note that Node 11 will be cheated by collusive adversaries because of receiving two *identical* bogus data packages from Node 7 and Node 10. Although Node 11 continues broadcasting the bogus data package, Node 12 and Node 15 will discard it and verify the signature themselves because they receive two different data packages. As a result, bogus data packages from two collusive adversaries cannot be injected into the WSN successfully. Moreover, if Node 11 listens to the channel for one more communication round after broadcasting the bogus package, it is also possible for Node 11 to find the potential attacks itself. More specifically, after Node 12 and Node 15 verify the signature successfully at Round 5, they will broadcast the correct data package to their neighbors. Node 11 will find that all the data packages received from its neighbors (i.e., Nodes 7, 10, 12, 15) are different and therefore some attacks have happened.

Like other cases, our faster signature verification needs to consume an extra energy of 1.5mJ for transmitting the intermediate computation results l_2Q. However, due to the existence of collusive adversaries Node 7 and Node 10, Node 11 will be fooled and then broadcast bogus data packages to Node 12 and Node 15. Note that both Node 12 and Node 15 will receive two different data packages from their neighbors and therefore verify the signature themselves. As a result, the signature verification will be accelerated only for Nodes 2, 3, 5, 6, 16 in this case, which can reduce the energy consumption of the WSN by $5 \times 11.52\text{mJ} = 57.6\text{mJ}$. Taking both communication and computation overheads into consideration, one can save around

$$\frac{57.6 - 1.5\text{mJ}}{14 \times 22.68\text{mJ}} \times 100\% = 17.7\%$$

energy consumption for the 4×4 grid-based WSN. In particular, attacks from two collusive adversaries have very limited effect on the security of our scheme and those attacks can also be detected in the certain communication round.

6 Conclusions

Signature-based broadcast authentication schemes for WSNs have attracted a lot of attention in recent years due to their desirable features such as strong

security resilience, good scalability and immediate message authentication. However, the relatively slow signature verification in public-key cryptosystems causes high energy consumption and long verification delay for broadcast authentication in WSNs. In this chapter, we propose a novel and efficient acceleration technique for signature verification in WSNs. Our scheme fully exploits the *cooperation* among sensor nodes and enables the significant energy consumption saving for the whole network. As a case study, we apply our technique to the broadcast authentication in a 4×4 grid-based WSN and analyze the security and performance of our scheme under the existence of independent and collusive adversaries. While independent adversaries do not have any influence on the security of our scheme, collusive adversaries have very limited effect. Particularly, bogus data packages from adversaries cannot be disseminated successfully through the entire network in both cases. Moreover, a quantitative performance analysis shows that our scheme can save about $17.7\% \sim 34.5\%$ energy consumption and run 50% faster than traditional signature verification method.

References

1. Akyildiz, L., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A Survey on Sensor Networks. IEEE Communications Magazine 40(8), 102–116 (2002)
2. Bellare, M., Namprempre, C., Neven, G.: Security Proofs for Identity-Based Identification and Signature Schemes. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 268–286. Springer, Heidelberg (2004)
3. Chang, S., Shieh, S., Lin, W., Hsieh, C.-M.: An Efficient Broadcast Authentication Scheme. In: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security (ASIACCS 2006), pp. 311–320 (2006)
4. Cao, X., Kou, W., Dang, L., Zhao, B.: IMBAS: Identity-Based Multi-User Broadcast Authentication in Wireless Sensor Networks. Computer Communications 31(4), 659–667 (2008)
5. Courtois, N., Finiasz, M., Sendrier, N.: How to Achieve a McEliece-Based Digital Signature Scheme. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 157–174. Springer, Heidelberg (2001)
6. Crossbow Technology Inc.: MICAz – Wireless Measurement System, `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf`
7. Dong, Q., Liu, D., Ning, P.: Pre-Authentication Filters: Providing DoS Resistance for Signature-Based Broadcast Authentication in Wireless Sensor Networks. In: Proceedings of the First ACM Conference on Wireless Network Security (WiSec 2008), pp. 2–12 (2008)
8. Driessen, B., Poschmann, A., Paar, C.: Comparison of Innovative Signature Algorithms for WSNs. In: Proceedings of the First ACM Conference on Wireless Network Security (WiSec 2008), pp. 30–35 (2008)
9. Drissi, J., Gu, Q.: Localized Broadcast Authentication in Large Sensor Networks. In: Proceedings of the International Conference on Networking and Services (ICNS 2006), pp. 25–31 (2006)
10. Liu, A., Ning, P.: TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks. In: Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN 2008), SPOTS Track, pp. 245–256 (2008)

11. Liu, D., Ning, P.: Multi-Level μTESLA: Broadcast Authentication for Distributed Sensor Networks. ACM Transactions on Embedded Computing Systems 3(4), 800–836 (2004)
12. Liu, D., Ning, P., Zhu, S., Jajodia, S.: Practical Broadcast Authentication in Sensor Networks. In: Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous 2005), pp. 118–129 (2005)
13. López, J., Aranha, D., Câmara, D., Dahab, R., Oliveira, L., Lopes, C.: Fast Implementation of Elliptic Curve Cryptography and Pairing Computation for Sensor Networks. In: The 13th Workshop on Elliptic Curve Cryptography (ECC 2009), http://ecc.math.ucalgary.ca/sites/ecc.math.ucalgary.ca/files/u5/Lopez_ECC2009.pdf
14. Luk, M., Perrig, A., Whillock, B.: Seven Cardinal Properties of Sensor Network Broadcast Authentication. In: Proceedings of the Fourth ACM workshop on Security of Ad Hoc and Sensor Networks (SASN 2006), pp. 147–156 (2006)
15. Ning, P., Liu, A., Du, W.: Mitigate DOS Attacks Against Broadcast Authentication in Wireless Sensor Networks. ACM Transactions on Sensor Networks 4(1), 1:1–1:35 (2008)
16. Perrig, A.: The BiBa One-Time Signature and Broadcast Authentication Protocol. In: Proceedings of the 8th ACM conference on Computer and Communications Security (CCS 2001), pp. 28–37 (2001)
17. Perrig, A., Szewczyk, R., Tygar, J.D., Wen, V., Culler, D.E.: SPINS: Security Protocols for Sensor Networks. ACM Wireless Networks 8(5), 521–534 (2002)
18. Ren, K., Lou, W., Zeng, K., Moran, P.J.: On Broadcast Authentication in Wireless Sensor Networks. IEEE Transactions on Wireless Communications 6(11), 4136–4144 (2007)
19. Ren, K., Yu, S., Lou, W., Zhang, Y.: Multi-user Broadcast Authentication in Wireless Sensor Networks. To appear in IEEE Transactions on Vehicular Technology (2009)
20. Seo, S.C., Han, D.-G., Song, S.: TinyECCK: Efficient Elliptic Curve Cryptography Implemenation over $GF(2^m)$ on 8-bit Micaz Mote. IEICE Transactions on Information and Systems E91-D(5), 1338–1347 (2008)
21. Shirase, M., Miyazaki, Y., Takagi, T., Han, D.-G., Choi, D.: Efficient Implementation of Pairing Based Cryptography on a Sensor Node. IEICE Transactions on Information and Systems E92-D(5), 909–917 (2009)
22. Szczechowiak, P., Kargl, A., Scott, M., Collier, M.: On the Application of Pairing Based Cryptography to Wireless Sensor Networks. In: Proceedings of the Second ACM Conference on Wireless Network Security (WiSec 2009), pp. 1–12 (2009)
23. Texas Instrument Inc. 2.4 GHz IEEE 802.15.4/ZigBee-ready RF Transceiver, http://focus.ti.com/lit/ds/symlink/cc2420.pdf
24. Zhang, Q., Yu, T., Ning, P.: A Framework for Identifying Compromised Nodes in Wireless Sensor Networks. ACM Transactions in Information and Systems Security (TISSEC) 11(3), 1–37 (2008)