Efficient Online/Offline Identity-Based Signature for Wireless Sensor Network

Joseph K. Liu Joonsang Baek Jianying Zhou Yanjiang Yang
Jun Wen Wong
Institute for Infocomm Research
Singapore
{ksliu, jsbaek, jyzhou, yyang, jwwong}@i2r.a-star.edu.sg

Abstract

In this paper, we present an *online/offline identity-based signature* scheme for the wireless sensor network (WSN). We argue that due to significant reduction in computational and storage costs, our scheme is particularly suitable for the WSN environment with severely constrained resources. One of the interesting features of our scheme is that it provides *multi-time* usage of the offline storage, which allows the signer to re-use the offline pre-computed information in polynomial time, in contrast to *one-time* usage in all previous online/offline signature schemes. As evidence of the practicality and feasibility of our scheme to be used in the WSN environment, we provide an actual implementation result of our scheme on the MicaZ platform.

1 Introduction

WSN APPLICATIONS AND SECURITY. A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations. There are many potential applications for WSNs [2]. They could be used in commercial and industrial applications to monitor data that would be difficult or expensive to monitor using wired sensors. They could be used to monitor situations in some hazard environments, such as in nuclear power plants. They could also be deployed in wilderness areas, where they would remain operation for many years (monitoring some environmental variables) without the need to recharge/replace their power supplies. They could form a perimeter about a property and monitor the progression of intruders.

WSNs are more vulnerable to various attacks due to their nature of wireless communication. In some WSN applications, providing authentication for sensed data is of prime importance. For example [23], in radiological facilities where sensors collect data on radioactive levels of nuclear power plants and transmit them to base stations or workers' dosimeters, it should be assured that the collected data are authentic and have not been altered during transmission in order to avoid malfunction or other possible hazards to the workers due to misinterpretation caused by altered data. Another example [23] is the social/health care systems where information about elderly people or patients' emergent conditions is transmitted from sensors to base stations. Again, authenticity of data transmitted through sensors is crucial in those systems in that altered/modified data could have serious consequences for the people in critical or dangerous situations.

However, since sensors usually have very constrained resources in terms of computing, communication, memory, and battery power, providing authenticity in WSN poses different challenges than traditional network/computer security [19, 17]. It requires lightweight and power-saving cryptographic algorithms to support WSN security [14, 19]. For this reason, only symmetric-key cryptographic algorithms have been

regarded as suitable tools for providing WSN with security. Contrary to this common belief, it has recently been reported that public-key cryptographic algorithms are feasible to be realized in WSNs and in fact practical if right algorithms are chosen [10, 25, 3]. A significant benefit one can obtain from using public-key cryptographic algorithms for WSN security is that it simplifies essential security services including key distribution/management and hence reduces transmission power due to less protocol overhead [10].

One important issue that should be resolved in order to fully utilize public-key cryptography in WSN is to build up a public key infrastructure (PKI) for WSN [26, 20], which is to establish trusted identity among other things. However, as pointed out in [20], the PKI for WSNs is not trivial to construct. Especially, distributing signed public-key certificates of numerous sensors could be difficult in many situations. We argue in this paper that at least for providing data authentication services, *identity-based signature* schemes will be very useful due to the feature that a signer does not have to hold a signed public-key certificate for other entities to verify signatures that the signer generates.

IDENTITY-BASED CRYPTOGRAPHY. Identity-based (ID-based) cryptography, introduced by Shamir [21], eliminates the necessity for checking the validity of certificates. In an ID-based cryptography, public key of each user is easily computable from a string corresponding to this user's identity (e.g. an email address, a telephone number, etc.). A private key generator (PKG) then computes the private keys from a master secret for the users. This property avoids the requirement of using certificates and associates an implicit public key (user identity) to each user within the system. In the case of ID-based signature (IBS), verification takes only the identity together with the message and signature pair as input and executes the algorithm directly. This is different from the traditional public-key cryptography, whereas an additional certificate verification algorithm is needed which is equivalent to the process of two signatures verification.

Identity-based cryptography could particularly be suitable for WSN. The absence of certificate eliminates the costly certificate verification process. In addition, when there is a new node added to the network, other nodes do not need to have its certificate in order to communicate in a secure and authenticated way. This can greatly reduce communication overhead and computation cost, which is a significant factor in the design of WSN. Recently, Tan *et al.* [24] proposed an identity-based encryption scheme for body sensor network (BSN), a network of sensors deployed on a person's body to collect physiological information.

ONLINE/OFFLINE SIGNATURE. In order to further reduce the computational cost of signature generation, online/offline signature is preferable in WSN. The notion of online/offline signatures was introduced by Even, Goldreich and Micali [9]. It performs the signature generation procedure in two phases. The first phase is performed offline (prior to the knowledge of the message to be signed) and the second phase is performed online (after knowing the message to be signed). In WSN, the offline phase can be executed at the base station, while the online phase is to be executed in the WSN node. The online phase is typically very fast, and hence can be executed efficiently even on a weak processor, such as a node in WSN.

Even, Goldreich and Micali proposed a general method for converting any signature scheme into an online/offline signature scheme. However, the method is impractical since it increases the size of the signature by a quadratic factor. Later, Shamir and Tauman [22] proposed a new paradigm, called "hash-sign-switch" for designing more efficient online/offline signature schemes. Both schemes are in generic setting, and thus not actually very efficient or practical to be used. Some concrete implementations have been proposed in [15, 8, 13, 5]. Among these schemes, [15] and [13] are proven secure without random oracles while [5] is the most efficient one. However, all schemes are only for traditional public-key based setting, but not targeted for identity-based setting.

1.1 Related Works

The only existing ID-based online/offline signature scheme was designed by Xu, Mu and Susilo [27] (this scheme will be referred to as the "XMS" scheme hereafter). In their scheme, the signer needs to execute the

offline phase every time when he wants to produce a signature. We call it "*one-time*" meaning the offline signature part can be used only *once* and hence, it cannot be re-used. If we apply this one-time scheme into WSN, it is becoming impractical since, assuming the offline phase is done at the base station, non-reusability of the storage implies that sensors need to go back to the base station every time for obtaining the next offline signature part. Moreover, the verification of the XMS scheme requires a *pairing* operation, which is a costly computation process with respect to a sensor node. We do not expect a node can execute such a heavy operation, which makes the signature scheme not appropriable for node-to-node signature in WSN [1].

Even worse, it was recently found by Li et al. [16] that the XMS scheme does not achieve the claimed security, i.e., the signature is *forgeable*. In other words, there is no secure concrete identity-based online/offline signature scheme existed in the literature yet.

1.2 Our Contribution

In this paper, we present an efficient online/offline ID-based signature scheme which is suitable for WSN. It enjoys the following advantages:

- It is a concrete identity-based setting which does not require any certificate attached to the signature for verification. It is proven secure in the random oracle model.
- When compared to the XMS scheme [27], which is the only one ID-based online/offline signature in the literature [2], our scheme requires less computation and storage cost (up to 50% saving). The XMS scheme requires two pairing operations in the signature verification.
- More importantly, our scheme does not require any pairing operation in both signature generation or verification. Therefore, our scheme can be easily implemented in WSN nodes. Our scheme is especially suitable for node-to-node communication in WSN, in the sense that no certificate is needed and computations are light enough to be executed.
- Our new technique allows the offline information to be *re-usable*. This way, the signer is not required to execute the offline algorithm every time when he wants to sign a new message. Furthermore, unlike most of the existing (non ID-based) online/offline signatures, our offline signing algorithm does *not* require any secret key information. That is, it can be generated by any third party. This is particularly useful for a WSN node as it does not need to return to the base station for the renewal of the offline information. The offline information can be hardcoded into the node in the manufacturing stage. This can save a lot communication bandwidth, which is considered to be an expensive cost in the WSN environment.

1.3 Organization

Our paper is organized as follow. We review some definitions in Section 2. Our scheme is given in Section 3, which is followed by the security and performance analysis in Section 4. An extension is given in Section 5. The implementation details of our scheme is presented in Section 6 and our paper is concluded in Section 7.

[1] Oliveira *et al.* [18] shows that using MicaZ node processor with TinyPBC open source code to execute a pairing operation requires at least 5.5 seconds. A pairing-based signature verification takes at least 2 pairing operations, which needs more than 11 seconds. It is too long for many practical applications.

[2] We remark that the XMS scheme has been proven *insecure* by Li et al. [16]. However, as it is the only concrete identity-based online/offline signature scheme in the literature, we also include it in our efficiency comparison.

2 Definition

2.1 Mathematical Assumption

The security of our scheme will be reduced to the hardness of the discrete logarithm (DL) problem in the group in which the signature is constructed. We briefly review the definition.

Definition 1 (Discrete Logarithm (DL) Assumption) *Given a group $\mathbb{G}$ of prime order q with generator g and element $g^x \in \mathbb{G}$ where x is selected uniformly at random from $\mathbb{Z}_q^*$, the discrete logarithm (DL) problem in $\mathbb{G}$ is to compute x. We say that the (ϵ, t)-DL assumption holds in a group $\mathbb{G}$ if no algorithm running in time at most t can solve the DL problem in $\mathbb{G}$ with probability at least ϵ.*

2.2 Security Definition

We then review the formal definition of the online/offline ID-based signature (IBS) scheme.

Definition 2 ($\mathcal{IBS}$) An online/offline ID-based signature scheme $\mathcal{IBS}$ consists of algorithms Setup, Extract, OfflineSign, OnlineSign and Verify.

- Setup: This algorithm computes a PKG's public parameter $param$ and a master key msk. Note that $param$ is given to all parties involved while msk is kept secret.
- Extract: Given an identity ID, this algorithm generates a private key associated with ID using msk, denoted by sk_{ID}.
- OfflineSign: Given the public parameter, this algorithm generates an offline signature $\bar{\sigma}$.
- OnlineSign: On input the private key sk_{ID}, the offline signature $\bar{\sigma}$ and a message m, this algorithm generates a signature σ of the message m.
- Verify: Given ID, m and σ, this algorithm outputs "accept" if σ is valid and outputs "reject" otherwise.

Note that we do not require the secret key to be the input of OfflineSign in our definition.

Next, we define the unforgeability notion for $\mathcal{IBS}$, which we call "UF-IBS-CMA (Unforgeability of $\mathcal{IBS}$ under chosen message attack)".

Definition 3 (UF-IBS-CMA) An ID-based signature scheme $\mathcal{IBS} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Sign}, \mathsf{Verify})$ is secure in the sense of existential unforgeable against chosen message attack (UF-IBS-CMA) if there is no adversary F whose running time is polynomial bounded, given the set of common parameters $param$ generated by Setup, wins the following attack game with non-negligible probability. Note that in the attack game, the adversary interacts with the challenger through queries.

1. When F issues a private key extraction query `ID` an identity, the challenger runs Extract providing ID as input, obtains a corresponding private key sk_{ID} and responds to F with it.
2. When F issues a signature generation query which consists of an identity ID and a message m, the challenger runs Extract providing `ID` as input, obtains a corresponding private key sk_{ID}. The challenger then runs the Sign algorithm providing sk_{ID} as input and gives a resulting signature σ to F.

3. At the end of the game, F outputs (ID', m', σ'), where σ' is a signature of a message m' and ID' is a corresponding identity. A restriction here is that ID' and m' have not been issued as any of the private key extraction and signature generation queries before.

F wins the attack game if σ' is a valid signature of m'. The advantage of an adversary is defined as the probability it wins the game. An adversary is said to be an $(\epsilon, t, q_e, q_s, q_h)$-forger if it has advantage at least ϵ in the above game, runs in time at most t, and make at most q_e, q_s and q_h extract, signing and random oracle queries, respectively. A scheme is said to be $(\epsilon, t, q_e, q_s, q_h)$-secure in the sense of UF-IBS-CMA if no $(\epsilon, t, q_e, q_s, q_h)$-forger exists.

3 The Proposed Online/Offline IBS Scheme

We present our scheme in this section. It contains the following 5 components.

– Setup: Let $\mathbb{G}$ be a multiplicative group with order q. The PKG selects a random generator $g \in \mathbb{G}$ and randomly chooses $x \in \mathbb{Z}_q^*$ at random. It sets $X = g^x$. Let $H : \{0,1\} \rightarrow \mathbb{Z}_q^*$ be a cryptographic hash function. The public parameters *param* and master secret key *msk* are given by

$$param = (\mathbb{G}, q, g, X, H) \qquad msk = x$$

– Extract: To generate a secret key for identity ID, the PKG randomly selects $r \in \mathbb{Z}_q^*$ at random, computes

$$R \leftarrow g^r \qquad s \leftarrow r + H(R, ID)x \bmod q$$

The user secret key is (R, s). Note that a correctly generated secret key should fulfill the following equality:

$$g^s = RX^{H(R,ID)} \tag{1}$$

– Offline Sign: At the offline stage the signer computes:

$$\hat{Y}_i \leftarrow g^{-2^i} \qquad \text{for } i = 0, \ldots, |q| - 1$$

Note that at the offline stage, we do not require the knowledge of the message or even the secret key. This can be done by other third party. It can be also regarded as part of the public parameter and prepared by the PKG instead of offline signing stage.

– Online Sign: At the online stage, the signer randomly selects $y \in \mathbb{Z}_q^*$ at random. Let $y[i]$ be the i-th bit of y. Define $\mathcal{Y} \subset \{1, \ldots, |q|\}$ to be the set of indices such that $y[i] = 1$. Compute

$$Y \leftarrow \prod_{i \in \mathcal{Y}} \hat{Y}_{i-1} \qquad h \leftarrow H(Y, R, m) \qquad z \leftarrow y + h\, s \bmod q$$

The signature is (Y, R, z).

– Verify: To verify the signature (Y, R, z) for message m and identity ID, the verifier first computes $h \leftarrow H(Y, R, m)$ and checks whether

$$g^z \stackrel{?}{=} YR^h X^{hH(R,ID)} \tag{2}$$

Accept if it is equal. Otherwise reject.

For correctness, note that $Y = g^y$. We have

$$\begin{aligned} & YR^h X^{hH(R,ID)} \\ = \; & g^y g^{rh} g^{xhH(R,ID)} \\ = \; & g^{y+h(r+H(R,ID)x)} \\ = \; & g^{y+hs} \\ = \; & g^z \end{aligned}$$

Remarks: In our scheme, the offline part can be executed by any third party. The offline information can also be re-used. Actually if we put the offline signing stage as part of the setup process which is done by the PKG (and the offline information is put as part of the public parameter), our scheme can be regarded as a *normal* identity-based signature scheme with very efficient signing algorithm that does not require any exponentiation.

4 Security and Performance Analysis

4.1 Security Analysis

Theorem 1 *The proposed scheme is $(\epsilon, t, q_e, q_s, q_h)$-secure in the sense of UF-IBS-CMA in the random oracle model, assuming that the (ϵ', t')-DL assumption holds in $\mathbb{G}$, where*

$$\epsilon' = \Big(1 - \frac{q_h(q_e + q_s)}{q}\Big)\Big(1 - \frac{1}{q}\Big)\Big(\frac{1}{q_h}\Big)\epsilon, \qquad t' = t + \mathcal{O}(q_e + q_s)E$$

and q_e, q_s, q_h are the number of extraction, signing and hashing queries respectively the adversary is allowed to make and E is the time for an exponentiation operation.

Proof. Assume that there exists a forger $\mathcal{A}$. We construct an algorithm $\mathcal{B}$ that makes use of $\mathcal{A}$ to solve discrete logarithm problem. $\mathcal{B}$ is given a multiplicative group $\mathbb{G}$ with generator g and order q, and a group element $A \in \mathbb{G}$. $\mathcal{B}$ is asked to find $\alpha \in \mathbb{Z}_q$ such that $g^\alpha = A$. We follow the proof technique from [4].

Setup: $\mathcal{B}$ chooses a hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ which behaves like a random oracle. $\mathcal{B}$ is responsible for the simulation of this random oracle. $\mathcal{B}$ assigns $X \leftarrow A$ and outputs the public parameter $param = (\mathbb{G}, q, g, X, H)$ to $\mathcal{A}$.

Extraction Oracle: $\mathcal{A}$ is allowed to query the extraction oracle for an identity ID. $\mathcal{B}$ simulates the oracle as follows. It chooses $a, b \in \mathbb{Z}_q$ at random and sets

$$R \leftarrow X^a g^b \qquad s \leftarrow b \qquad H(R, ID) \leftarrow -a$$

Note that (R, s) generated in this way satisfies the equation (1) in the Extract algorithm. It is a valid secret key. $\mathcal{B}$ outputs (R, s) as the secret key of ID and stores the value of $(R, s, H(R, ID), ID)$ in the table for consistency.

Signing Oracle: $\mathcal{A}$ queries the signing oracle for a message m and an identity ID. $\mathcal{B}$ first checks that whether ID has been queried for the random oracle H or extraction oracle before. If yes, it just retrieves $(R, s, H(R, ID))$ from the table and uses these values to sign for the message, according to the signing algorithm described in the scheme. It outputs the signature (Y, R, z) for the message m and stores the value $H(Y, R, m)$ in the hash table for consistency. If ID has not been queried to the extraction oracle, $\mathcal{B}$ executes the simulation of the extraction oracle and uses the corresponding secret key to sign the message.

Output Calculation: Finally the adversary $\mathcal{A}$ outputs a forged signature $\sigma^*_{(1)} = (Y^*, R^*, z^*_{(1)})$ on message m^* and identity ID^*. $\mathcal{B}$ rewinds $\mathcal{A}$ to the point it just queries $H(Y^*, R^*, m^*)$ and supplies with a different value. $\mathcal{A}$ outputs another pair of signature $\sigma^*_{(2)} = (Y^*, R^*, z^*_{(2)})$. $\mathcal{B}$ repeats again and obtains $\sigma^*_{(3)} = (Y^*, R^*, z^*_{(3)})$. Note that Y^* and R^* should be the same every time. We let c_1, c_2, c_3 be the output of the random oracle queries $H(Y^*, R^*, m^*)$ for the first, second and third time.

By $r, x, y \in \mathbb{Z}_q$ we now denote discrete logarithms of R, X and Y respectively, i.e., $g^r = R$, $g^x = X$ and $g^y = Y$. From equation (2),we then have

$$z^*_{(i)} = y + rc_i + xc_iH(R^*, ID) \bmod q \qquad \text{for } i = 1, 2, 3$$

In these equations, only r, y, x are unknown to $\mathcal{B}$. $\mathcal{B}$ solves for these values from the above three linear independent equations, and outputs x as the solution of the discrete logarithm problem.

Probability Analysis: The simulation of the extraction oracle fails if the random oracle assignment $H(R, ID)$ causes inconsistency. It happens with probability at most q_h/q. Hence the simulation is successful $q_e + q_s$ times (since $H(R, ID)$ may also be queried in the signing oracle if ID has not been queried in the extraction oracle) with probability at least

$$\Big(1 - \frac{q_h}{q}\Big)^{q_e+q_s} \geq 1 - \frac{q_h(q_e + q_s)}{q}.$$

Due to the ideal randomness of the random oracle, there exists a query $H(Y^*, R^*, m^*)$ with probability at least $1 - 1/q$. $\mathcal{B}$ guesses it correctly as the point of rewind, with probability at least $1/q_h$. Thus the overall successful probability is

$$\Big(1 - \frac{q_h(q_e + q_s)}{q}\Big)\Big(1 - \frac{1}{q}\Big)\Big(\frac{1}{q_h}\Big)\epsilon.$$

The time complexity of the algorithm $\mathcal{B}$ is dominated by the exponentiations performed in the extract and signing queries, which is equal to

$$t + \mathcal{O}(q_e + q_s)E.$$

□

4.2 Efficiency Analysis

We note that exponentiation is equivalent to point multiplication in Elliptic Curve Cryptosystem (ECC) and multiplication is equivalent to point addition in ECC. Since a 160-bit ECC key offers more or less the same level of security as a 1024-bit RSA, we may implement our proposed scheme using ECC with $|q| = 160$ ($|\mathbb{G}|$ can be as small as 160 in the optimal case by choosing suitable curve [6]). We adopt this setting in the following comparison with other schemes.

Table 1: Comparison of computation cost

	ST's scheme	XMS's scheme	Our scheme
Offline (One-time)	$C(h) + C(\sigma_g)$	$2E + \tilde{m}$	0
Offline (Multi-time)	$-$	$\|q\| \cdot 2E$	0
Online (One-time)	$\tilde{m}$	$\tilde{m}$	$\tilde{m}$
Online (Multi-time)	$-$	$\mathcal{O}(\|q\|) \cdot 2M + \tilde{m}$	$\mathcal{O}(\|q\|) \cdot M + \tilde{m}$
Verification	$C(h) + C(\sigma_v) + C(cert_v)$	$2P + 2E + M$	$2E + M$

Table 2: Comparison of storage cost and signature size

	ST's scheme	XMS's scheme	Our scheme
Offline Storage (One-time)	$2\|q\| + \|\sigma\| + \|cert\|$ ≥ 800 bits	$2\|\mathbb{G}\| + 2\|q\|$ ≈ 640 bits	$\|\mathbb{G}\| + \|q\|$ ≈ 320 bits
Offline Storage (Multi-time)	$-$	$2\|q\| \cdot \|\mathbb{G}\|$ $\approx 6.4k$ bytes	$\|q\| \cdot \|\mathbb{G}\|$ $\approx 3.2k$ bytes
Size of signature	$\|q\| + \|\sigma\| + \|cert\|$ ≥ 640 bits	$2\|\mathbb{G}\| + \|q\|$ ≈ 480 bits	$2\|\mathbb{G}\| + \|q\|$ ≈ 480 bits

4.2.1 Comparison with other online/offline schemes

We compare the efficiency of our scheme with two different ID-based online/offline signature schemes, namely Shamir-Tauman's (ST) scheme [22] (ID-based version, with certificate attached as part of the signature) and Xu-Mu-Susilo's (XMS) scheme [27]. Here we remark that the XMS scheme did not provide a multi-time version of the online/offline signature. However, we observe that the same technique we employed in Section 3 can be applied to their scheme and thus produce an appropriate comparison. On the other side, the ST scheme cannot be extended to multi-time version.

We denote by $C(\theta)$ the computation cost of operation θ, and by $|\lambda|$ the bits of λ. Also we denote by E the exponentiation in $\mathbb{G}$ (equivalent to scalar multiplication in ECC), M the multiplication in $\mathbb{G}$ (equivalent to point addition in ECC), $\tilde{m}$ the modular multiplication in $\mathbb{Z}_q^*$ and P the pairing operation. We omit other operations such as addition in $\mathbb{Z}_q^*$ and normal hashing.

Table 1 shows the comparison of computation cost. h represents a Chameleon hash operation, which requires at least one E computation. σ_g and σ_v represent a normal signature generation and verification respectively, which require at least one E computation for each operation. Similarly, $cert_v$ represents a certificate verification, which also requires at least one E computation.

Table 2 shows the comparison of offline storage cost and size of the signature. As stated above, $|q|$ and $|\mathbb{G}|$ are both 160 bits. $|\sigma|$ represents the length of a normal digital signature, which is at least 160 bits. $|cert|$ represents the length of a digital certificate, which is at least 320 bits.

From the above comparison, we can observe that our proposed scheme is much more efficient than Shamir-Tauman's generic construction. When comparing to the XMS scheme, we achieve about 50% improvement over space and computation efficiency of both the offline and online stage. Furthermore, in our scheme as the offline stage can be done by the PKG, the signer does *not* have any computation cost in the offline stage while the XMS scheme requires more than 320 E operations.

The most significant improvement focuses on the signature verification. We do *not* require any pairing operation while the XMS scheme does. It is particularly suitable for the WSN environment where the sensor node does not have enough computation power for a pairing operation. Without any pairing operation, we allow any node to generate and verify signature. That is, our proposed signature scheme facilitates the communication between nodes in an authenticated way.

Table 3: Comparison of computation cost and size

	HS's scheme	CC's scheme	GQ's scheme	Our scheme
Signing	$3E + M$	$2E$	$2E + 2\tilde{m}$	$\tilde{m}$
Verification	$2P + E$	$2P + 2E + M$	$4E + 2M$	$2E + M$
Size of Signature	320 bits	320 bits	2048 bits	480 bits

4.2.2 Comparison with other non-online/offline schemes

We also compare our scheme with other (non-online/offline) ID-based signature schemes that have been standardized by ISO/IEC: Hess's [12] scheme (denoted HS), Cha and Cheon's [7] scheme (denoted CC) and Guillou and Quisquater's [11] scheme (denoted GQ). The detailed comparison is given in Table 3. We use the same notation as in Table 1.

From the above, we can see that those non-online/offline schemes may not be suitable for lightweight devices, such as wireless sensors. Both HS and CC schemes require E (exponentiation) operation in the signing stage and P (pairing) operation in the verification stage, which are considered fairly heavy. Due to resource constraints, lightweight devices may not be able to execute such operations.

5 Extension for Aggregation

It would be useful if a (single) sensor node can sign multiple messages, say n messages, but the size of resulting signature is significantly smaller than n times the size of a single signature. Such an aggregated (shortened) signature is of great importance in WSNs as reducing communication overheads in WSNs is crucial for resource-constrained sensor nodes.

As an extension to our online/offline IBS scheme, we propose the following aggregation technique when a single user (node) wants to sign multiple messages.

– Setup: Let $\mathbb{G}$ be a multiplicative group with order q. The PKG selects a random generator $g \in \mathbb{G}$ and randomly chooses $x \in_R \mathbb{Z}_q$. It sets $X = g^x$. Let $H : \{0,1\} \rightarrow \mathbb{Z}_q$ be a cryptographic hash function. The public parameters *param* and master secret key *msk* are given by

$$param = (\mathbb{G}, q, g, X, H) \qquad msk = x$$

– Extract: To generate a secret key for identity ID, the PKG randomly selects $r \in_R \mathbb{Z}_q^*$, computes

$$R \leftarrow g^r \qquad s \leftarrow r + H(R, ID)x \bmod q$$

The user's secret key is (R, s).

– Offline Sign: At the offline stage the signer computes:

$$\hat{Y}_i \leftarrow g^{-2^i} \qquad \text{for } i = 0, \ldots, |q| - 1$$

As noted in the previously, this offline stage computation can be conducted by other third party or the PKG. The resulting value $\hat{Y}_i$ for $i = 1, \ldots, |q| - 1$ can also be provided as part of the public parameter.

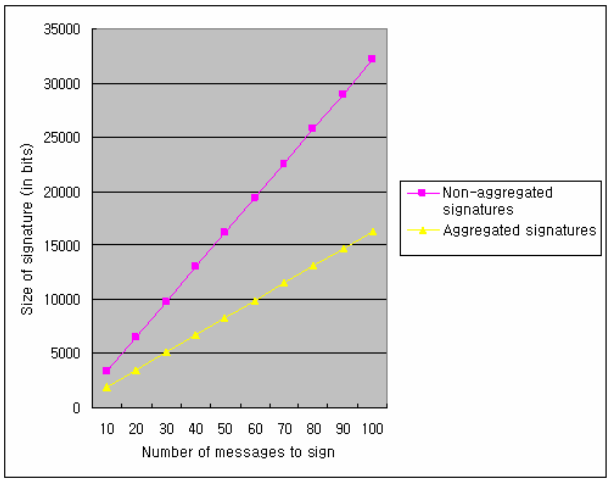

Figure 1: Comparison between Non-Aggregated and Aggregated Versions of Our Scheme

– Online Sign: At the online stage, the signer randomly selects $y_l \in_R \mathbb{Z}_q^*$. Let $y_l[i]$ be the i-th bit of y_l. Define $\mathcal{Y}_l \subset \{1, \ldots, |q|\}$ to be the set of indices such that $y_l[i] = 1$. Compute

$$Y_l \leftarrow \prod_{i \in \mathcal{Y}_l} \hat{Y}_{i-1} \qquad h_l \leftarrow H(Y, R, m_l) \qquad z_l \leftarrow y_l + h_l \, s \bmod q \quad \text{for } l = 1, \ldots, n$$

Also, compute

$$z = \sum_{l=1}^{n} z_l$$

The aggregated signature is (Y_l, R, z) for $l = 1, \ldots, n$.

– Verify: To verify the signature (Y_l, R, z) for message m_l and identity ID for $l = 1, \ldots, n$, the verifier first computes $h_l \leftarrow H(Y, R, m_l)$ and checks whether

$$g^z \stackrel{?}{=} \Big(\prod_{l=1}^{n} Y_l\Big) R^{\sum_{l=1}^{n} h_l} X^{(\sum_{l=1}^{n} h_l) H(R, ID)} \tag{3}$$

Accept if it is equal. Otherwise reject.

Note that the verification is correct: Since $Y_l = g^{y_l}$ for $l = 1, \ldots, n$, we have

$$\begin{aligned}
& \Big(\prod_{l=1}^{n} Y_l\Big) R^{\sum_{l=1}^{n} h_l} X^{(\sum_{l=1}^{n} h_l) H(R, ID)} \\
= \; & \Big(\prod_{l=1}^{n} Y_l\Big) g^{r(\sum_{l=1}^{n} h_l)} g^{x(\sum_{l=1}^{n} h_l) H(R, ID)} \\
= \; & g^{(\sum_{l=1}^{n} y_l)} g^{(\sum_{l=1}^{n} h_l)(r + xH(R, ID))} \\
= \; & g^{(\sum_{l=1}^{n} y_l)} g^{s(\sum_{l=1}^{n} h_l)} \\
= \; & g^{\sum_{l=1}^{n} (y_l + h_l s)} \\
= \; & g^z.
\end{aligned}$$

As depicted in Figure 1, when the proposed aggregation technique is employed, the size of signature is significantly reduced compared to the non-aggregated version in which the size of signature is almost *two times bigger* than the size of the aggregated signature.

6 Implementation on WSN

6.1 Basic Setting

We realized our online/offline ID-based signature scheme in the single-hop setting (see the implementation framework in Figure 2), in which each sensor node can sign messages using its secret signing key associated with its identifier information ID.

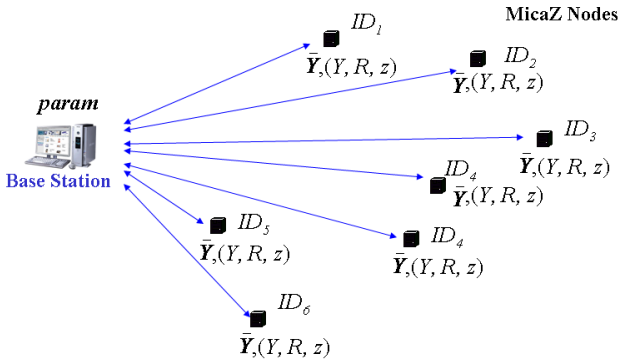

Figure 2: Overview of Implementation Scenario

We assume that the system parameter $param$ is generated by the base station and is embedded in each sensor node when they are deployed. We also assume that the signatures generated by the sensor nodes can be verified either by sensor nodes themselves or by the base station. Like the case for general WSNs, we assume that the base station is powerful enough to perform computationally intensive cryptographic operations, and the sensor nodes, on the other hand, have limited resources in terms of computation, memory and battery power. We also assume that the private key of the base station is safely stored.

The sensor nodes used in our implementation are MicaZ [3], developed by Crossbow Technology. Its RF transceiver complies with IEEE 802.15.4/ZigBee, and the 8-bit microcontroller is Atmel ATmega128L, a major energy consumer. We used a PC (Dell Dimension 9150 3.0 GHz CPU, 1GB RAM) as a base station.

The programming languages used for our implementation are nesC, C and Java. The base operating system for the MicaZ platform is TinyOS 2.0. For implementation of our scheme on the sensor node, we employed elliptic curve cryptography due to the small key size and low computational overhead. We specifically used an ECC library developed by Siemens AG [4] with 160-bit key size.

Figure 3 illustrates the data format of a packet in our implementation. The reason we split the signature into two phases instead of single phase is that the "R" part of our signature will be the same for all signatures produced from a particular sensor node; hence it will save communication overhead by sending R once at the very beginning of the communications. (Normal phase packet size for 1 stage = 123 bytes vs 2 stage = 83 bytes, 40 bytes or 320bits communication overhead saved for each signature!)

In initialize phase, the size of each packet is 43 bytes, i.e. 3 bytes for the header, 40 bytes for signature (20 bytes for R.x*, 20 bytes for R.y*), and the rest are zero padding. (Note that *R and Y are points on Elliptic curve, so we need x-y coordinate to represent it in Cartesian space.) In normal phase, the size of

[3] Further information on this platform is available at `http://www.xbow.com/`.

[4] Note that this library is not publicly available. One can use TinyECC instead, which is available as open source.

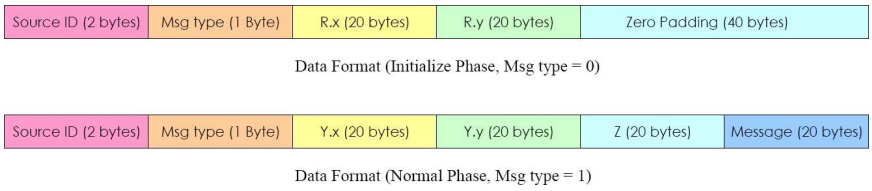

Figure 3: Data Format

each packet is 83 bytes, i.e. 3 bytes for the header, 60 bytes for signature (20 bytes for Y.x*, 20 bytes for Y.y*, 20 bytes for z), and 20 bytes for the payload.

6.2 Energy Consumption

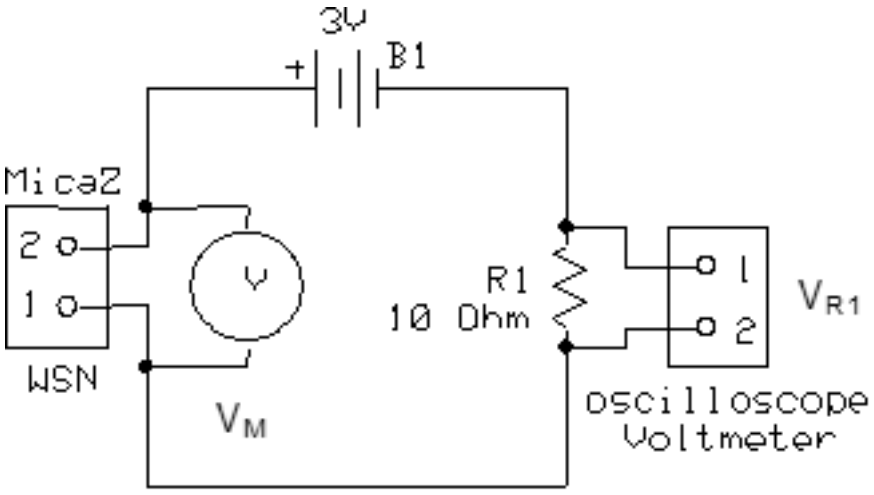

Figure 4: Power supply circuit for estimating energy consumption of MicaZ

Before we explain how to measure the energy consumption, let us review some basic formulae related to the energy and power. The energy E is defined as $E = P \cdot t$ where P denotes power and t denotes time. The unit of E is Joule. Here, $P = V \times I$ where V and I denote voltage and current respectively. Note that the unit of P is Watt. Note also that by Ohm's Law, $I = V/R$, where R is resistance. Since the actual energy consumed when running our codes in MicaZ cannot be calculated just based on its internal impedance, there is no way to estimate the impedance of logic gates. Hence, we measure the energy consumption of MicaZ indirectly. Figure 4 shows the power supply we built for estimating the energy consumption for our online/offline ID-based signature scheme. The circuit is powered by two Sanyo AA size NiMH rechargeable batteries, with fully charged and voltage level is at 2.97V. The reason we add a resistor, R_1, to the circuit instead of just connected to an Ammeter in series of the circuit is because we want to capture the current changes in the circuit and the period of changes at the same time. With this setup, we are able to measure the current flow into MicaZ indirectly by measuring the voltage drop, V_{R_1}, in the resistor R_1 using HP54520 oscilloscope. We choose a small value (10 Ohms) resistor (actual reading is 9.6 Ohms) in order to minimize the addition resistance to the circuit. After we had the current information, we measure the total voltage drop across Micaz, V_M, by using Fluke 87 voltmeter connected in parallel with MicaZ. By now, we are able to calculate the total power of the circuit in any instance. In order to get the energy consumption, we need the timing information. We program the MicaZ to sign and verify the signature periodically. With this, the oscilloscope is able to capture the computation time as the voltage across R_1 and V_{R_1} will change across

MicaZ during the computation of our online/offline ID-based signature scheme. Since we are only interested in the additional energy cost required for computing our online/offline ID-based signature scheme, we first estimate the energy consumed by MicaZ during computation and verification of the signature. After that we minus the energy consumed by MicaZ during its ideal state. By doing this we can have the actual energy consumption which is shown in Table 4.

Table 4: Time and energy consumptions of our implementation

Process Name	Time (s)	Energy (mJ)
Sign (Offline - Base Station)	0.293	nil
Sign (Online - MicaZ)	0.896	12.37
Verify (Base Station)	0.031	nil
Verify (MicaZ)	5.61	77.44

In Table 4, we summarize the time and energy consumptions of our implementation when a random message of 20 bytes is signed and verified using our online/offline ID-based signature scheme. Note that we have included two cases for signature verification, when the signature is verified by the base station and the MicaZ sensor node, respectively.

6.3 Comparison with other ECDSA implementations

Table 5: Comparison with other ECDSA implementations

Schemes	Sign (mJ)	Verify (mJ)
Our IBS implementation	Aprrox. 12.37	77.44
ECDSA implementation in [1]	46.2	58.4
ECDSA implementation in [25]	22.82	45.09

Although we have not found any implementations of IBS scheme on WSN in the available literature, one could compare our implementation with the ECDSA implementations given in [1] and [25], which may be comparable in a sense that they are all based on ECC with 160-bit key and are implemented on MicaZ. Readers are referred to Table 5 for the comparison. Note that the signing operation of our online/offline IBS consumes very little energy compared to those of [1] and [25]. This is due to the fact that in our online/offline IBS scheme, offline signing can be performed by the base station as no secret information is required and hence the sensor nodes are exempt from performing relatively "heavy" point multiplications. Note also that the verification of our scheme is only slightly more expensive. This is because our scheme is "identity-based" so more point multiplications should be performed.

7 Concluding Remark

We presented an efficient online/offline ID-based signature scheme which does not require any certificate attached to the signature for verification, and does not require any pairing operation in both signature generation or verification. More importantly, our offline signing algorithm does not require any secret key information. It can be pre-computed by any third party. The offline information can also be re-used. This is a great advantage in WSN environment as the offline information can be hard-coded to the sensor node in the manufacturing or setup stage. It can eliminate any communication between the sensor node and the base station for the offline signing, which is considered as a costly factor in the WSN.

The length of this (pre-computed) offline information, or can be considered as public parameters, is about 160 group elements. It may be considered long for signing a few messages. However, if the sensor requires to sign a thousand, or even a million of messages, these 160 group elements are just negligible when compared to those messages. Thus our scheme is particular suitable for large scale network.

References

[1] TinyECC - A configurable library for elliptic curve cryptography in wireless sensor networks (Version 1.0). Cyber Defense Laboratory, NCSU, 2007. `http://discovery.csc.ncsu.edu/software/TinyECC/`.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, 2002.

[3] J. Baek, H. Tan, J. Zhou, and J. Wong. Realizing stateful public key encryption in wireless sensor network. In *Proc. IFIP-SEC '08*, pages 95–108. Springer-Verlag, 2008.

[4] M. Bellare, C. Namprempre, and G. Neven. Security proofs for identity-based identification and signature schemes. In *Proc. EUROCRYPT '04*, volume 3027 of *Lecture Notes in Computer Science*, pages 268–286. Springer-Verlag, 2004.

[5] D. Boneh and X. Boyen. Short signatures without random oracles the SDH assumption in bilinear groups. *Journal of Cryptology*, 2:149–177, 2008.

[6] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In *Proc. ASIACRYPT '01*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer-Verlag, 2001.

[7] J. Cha and J. Cheon. An Identity-Based Signature from Gap Diffie-Hellman Groups. In *Proc. PKC'2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 18–30. Springer-Verlag, 2003.

[8] X. Chen, F. Zhang, W. Susilo, and Y. Mu. Efficient generic online/offline signatures without key exposure. In *Proc. ACNS '07*, volume 4521 of *Lecture Notes in Computer Science*, pages 18–30. Springer-Verlag, 2007.

[9] S. Even, O. Goldreich, and S. Micali. On-line/off-line digital signatures. In *Proc. CRYPTO '89*, volume 2442 of *Lecture Notes in Computer Science*, pages 263–277. Springer-Verlag, 1989.

[10] G. Gaubatz, J.-P. Kaps, E. Oztruk, and B. Sunar. State of the art in ultra-low power public key cryptography for wireless sensor networks. In *Proc. PerSec '05*, pages 146–150. IEEE, 2005.

[11] L. C. Guillou and J.-J. Quisquater. A "Paradoxical" Identity-Based Signature Scheme Resulting From Zero-Knowledge. In *Proc. CRYPTO 88*, volume 403 of *Lecture Notes in Computer Science*, pages 216–231. Springer-Verlag, 1989.

[12] F. Hess. Efficient Identity Based Signature Schemes Based on Pairings. In *Selected Area in Cryptography, SAC2002*, volume 2595 of *Lecture Notes in Computer Science*, pages 310–324. Springer-Verlag, 2003.

[13] M. Joye. An efficient on-line/off-line signature scheme without random oracles. In *Proc. CANS '08*, volume 5339 of *Lecture Notes in Computer Science*, pages 98–107. Springer, 2008.

[14] C. Karlof, N. Sastry, and D. Wagner. Tinysec: A link layer security architecture for wireless sensor networks. In *Proc. ACM SenSys '04*, pages 162–175. ACM, 2004.

[15] K. Kurosawa and K. Schmidt-Samoa. New online/offline signature schemes without random oracles. In *Proc. PKC '06*, volume 3958 of *Lecture Notes in Computer Science*, pages 330–346. Springer-Verlag, 2006.

[16] F. Li, M. Shirase, and T. Takagi. On the security of online/offline signatures and multisignatures from acisp'06. In *Proc. CANS '08*, volume 5339 of *Lecture Notes in Computer Science*, pages 108–119. Springer-Verlag, 2008.

[17] J. Lopez and J. Zhou. Wireless sensor network security. volume 1 of *Cryptology and Information Security Series*. IOS Press, 2008.

[18] L. B. Oliveira, M. Scott, J. Lopez, and R. Dahab. TinyPBC: Pairings for Authenticated Identity-Based Non-Interactive Key Distribution in Sensor Networks. Cryptology ePrint Archive, Report 2007/482, 2007. `http://eprint.iacr.org/`.

[19] A. Perrig, J. Stankovic, and D. Wagner. Security in wireless sensor networks. *Communications of the ACM*, 47(6):53–57, 2004.

[20] R. Roma and C. Alcaraz. Applicability of public key infrastructures in wireless sensor networks. In *Proc. EuroPKI '07*, volume 4582 of *Lecture Notes in Computer Science*, pages 313–320. Springer-Verlag, 2007.

[21] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proc. CRYPTO '84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1984.

[22] A. Shamir and Y. Tauman. Improved online/offline signature schemes. In *Proc. CRYPTO '01*, volume 2139 of *Lecture Notes in Computer Science*, pages 355–367. Springer-Verlag, 2001.

[23] SMEPP. Secure middleware for embedded P2P systems, 2006 to present. `http://www.smepp.org`.

[24] C. Tan, H. Wang, S. Zhong, and Q. Li. Body sensor network security: an identity-based cryptography approach. In *Proc. 1st ACM conference on Wireless Network Security*, pages 148–153. ACM, 2008.

[25] A. Wander, N. Gura, H. Eberle, V. Gupta, and S. Shantz. Energy analysis of public-key cryptography for wireless sensor networks. In *Proc. PerCom '05*, pages 324–328. IEEE Computer Society, 2005.

[26] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus. Tinypk: securing sensor networks with public key technology. In *Proc. 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 59–64. ACM, 2004.

[27] S. Xu, Y. Mu, and W. Susilo. Online/offline signatures and multisignatures for AVOD and DSR routing security. In *Proc. ACISP '06*, volume 4058 of *Lecture Notes in Computer Science*, pages 99–110. Springer-Verlag, 2006.