An Energy-Efficient Access Control Scheme for Wireless Sensor Networks based on Elliptic Curve Cryptography

Xuan Hung Le, Sungyoung Lee, Ismail Butun, Murad Khalid, Ravi Sankar, Miso (Hyoung-IL) Kim, Manhyung Han, Young-Koo Lee, and Heejo Lee

Abstract: **For many mission-critical related wireless sensor network applications such as military and homeland security, user's access restriction is necessary to be enforced by access control mechanisms for different access rights. Public key-based access control schemes are more attractive than symmetric-key based approaches due to high scalability, low memory requirement, easy key-addition/revocation for a new node, and no key predistribution requirement. Although Wang *et al.* recently introduced a promising access control scheme based on elliptic curve cryptography (ECC), it is still burdensome for sensors and has several security limitations (it does not provide mutual authentication and is strictly vulnerable to denial-of-service (DoS) attacks). This paper presents an energy-efficient access control scheme based on ECC to overcome these problems and more importantly to provide dominant energy-efficiency. Through analysis and simulation based evaluations, we show that the proposed scheme overcomes the security problems and has far better energy-efficiency compared to current scheme proposed by Wang *et al.***

Index Terms: **Elliptic curve cryptography (ECC), public-key cryptography, user access control, wireless sensor networks (WSN).**

I. INTRODUCTION

A wireless sensor network (WSN) [1] commonly consists of a large number of sensor nodes that are densely deployed either inside the phenomena or very close to it. It can sense physical phenomena (e.g., temperature, pressure, etc.) or detect events (e.g., intruders, fire emergency, volcanic eruption, etc.) from its surroundings, process and store them, and finally provide these data to users, upon either demand or event detection. Due to privacy reason or security clearance (i.e., a status granted to individuals allowing them access to classified information), user's access restriction may be enforced with different access rights. For example, a provider of a WSN which is deployed over a large geographic area offers paid services to many users. In a precision agriculture WSN [2], farmers subscribe to services and remotely query sensors on their fields using a mobile device like PDA. In this case, only authorized users should be answered by the network [3]. Another interesting example is a deployment of WSN in a battlefield. A high ranking officer should have more access rights than a soldier. As such a soldier is given access permissions to data information related to his task only and a high-ranking officer necessitates information gathering for an overall maneuver [4]. A simple but efficient way is to rely on popular symmetric-key cryptography. However, symmetric-key based schemes suffer a number of problems. It provides low scalability, requires large memory to store key materials, faces difficulty to add or revoke key and requires a complicated key pre-distribution [4]. The recent progress in public key cryptography using 160 bit elliptic curve cryptography (ECC) has shown that an ECC point multiplication takes less than one second on 8 bit CPU Atmel ATmega128 8 MHz [5]. This proves public key cryptography is feasible for sensor security related applications. Inspired by this result, Wang *et al.* proposed an ECC-based access control for WSNs [4] (for reference hereafter, we name their scheme HBQ). Though HBQ introduces a promising access control approach based on public key cryptography, it has several limitations as follows:

- It is burdensome for sensors in terms of time delay and energy consumption, which makes it unrealistic to employ in practice.
- It does not provide mutual authentication.
- It is vulnerable to denial-of-service (DoS) attack

To overcome these problems, we propose an *ENergy-efficient Access control scheme Based on eLliptic curvE cryptography* (ENABLE). ENABLE retains all advantages of public key cryptography and also enhances the security of HBQ. More importantly, ENABLE achieves better energy-efficiency than HBQ, and almost similar to symmetric-key based approaches.

The remainder of the paper is organized as follows. In Section II, we briefly review HBQ scheme and discuss their limitations. Section III describes assumptions and an adversary model of the proposed scheme. ENABLE scheme is described in Section IV. Section V and Section VI present analysis and simulation based evaluations of the proposed scheme. Finally, Section VII concludes the paper and outlines the future work.

II. REVIEW OF HBQ SCHEME

A. HBQ Protocol

A user needs to apply for access permissions from a key distribution center (KDC) to access the network. KDC maintains an access control list (ACL) pool and associated user identifications. User's access privileges are defined in an ACL that is typ-

Manuscript received April 23, 2009.

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2009-(C1090-0902-0002)). Also, it was supported by the IT R&D program of MKE/KEIT, [10032105, Development of Realistic Multiverse Game Engine Technology]. S. Lee is the corresponding author.

X. H. Le, S. Lee, M. Han, M. Kim, and Y.-K. Lee are with the Department of Computer Engineering, Kyung Hee University, Korea, email: {lxhung, sylee, smiley}@oslab.khu.ac.kr, meeso.kim@samsung.com, yklee@khu.ac.kr.

I. Butun, M. Khalid, and R. Sankar are with the Department of Electrical Engineering, University of South Florida, USA, email: {ibutun, mkhalid}@mail.usf.edu, sankar@eng.usf.edu.

H. Lee is with the Department of Computer Science and Engineering, Korea University, Korea, email: heejo@korea.ac.kr.

1229-2370/09/$10.00

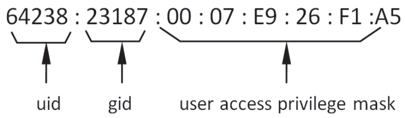


Fig. 1. An example of user access control list.

ically composed of user identifier (uid), group identifier (gid), and user access privileges mask. The user access privilege mask is a set of binary bits. Each bit represents permission of a specific information or service. An example of ACL is shown in Fig. 1.

Initially, KDC selects a particular elliptic curve over a finite field $GF(p)$ (where p is a prime) and publishes a base point P with a large order q (where q is also a prime). It picks a random number $x \in GF(p)$ as a private key, and publishes its corresponding public key $Q = xP$. To access the sensor network, a user, say *Alice*, comes to KDC and gets her public key (Q_A), private key (q_A), and the certificate of her access list and public key ($T_A = C_A||ac_A$, where '$||$' means concatenation). KDC picks a random number $c_A \in GF(p)$ and then calculates *Alice*'s public key constructor $C_A = c_A P$. Based on *Alice*'s request and her background check, KDC issues a proper ACL (ac_A) and attaches it to the public constructor C_A as a certificate. Meanwhile, a signature e_A is generated for the ACL, where $e_A = H(T_A)$ (H is a $\{0,1\}^* \rightarrow \{0,1\}^q$ hash function). Then, KDC constructs *Alice*'s private key $q_A = e_A c_A + x$ and public key $Q_A = e_A C_A + Q$. Note q_A and Q_A satisfy $Q_A = q_A P$. *Alice*'s access list T_A can be regarded as the certificate of her public key Q_A. Finally, *Alice* holds q_A, Q_A, and T_A. The HBA authentication protocol is described in Fig. 2. When *Alice* wants to access a sensor node s_l, she sends an access request with an access list T_A. Receiving T_A, s_l constructs *Alice*'s public key $Q_A = e_A C_A + Q$. To verify that *Alice* indeed holds the private key q_A, node s_l uses a challenge as follows. s_l selects a random number $r \in GF(q)$ (to be used as the session key with *Alice*) and calculates its signature $H(r)$ over $\mathrm{mod}(q)$. Node s_l then generates a temporary public key $Y_r = H(r)P$ and computes $Z_r = H(r)Q_A$. Afterwards, s_l encrypts the session key by computing $r \oplus X(Z_r)$, where $X(Z_r)$ is the x-coordinate of point Z_r. Finally, s_l sends a cipher text (z_r, Y_r) to *Alice*, attached with a message authentication code (MAC) of nonce N_A. Using the private key q_A, *Alice* can regenerate Z_r because $q_A Y_r = q_A H(r)P = H(r)Q_A = Z_r$. *Alice* then decrypts the session key $r = z_r \oplus X(Z_r)$, and verifies if $Y_r = H(r)P$. If Y_r is valid, *Alice* uses r as the session key to generate a MAC value of nonce N_A concatenated with her access privilege ac_A, and sends to s_l. Sensor s_l decrypts the MAC message and verifies N_A and ac_A. If they are valid, it proves that *Alice* is the owner of T_A. Finally, s_l replies with the information requested by *Alice*, which again is encrypted by a session key r.

B. Cryptanalysis of HBQ Scheme

Although HBQ scheme introduces a promising access control approach based on public key cryptography, it still possesses several limitations as mentioned earlier:

- **It is burdensome for sensors**: As the authors discussed in their paper [4], the authentication takes about 10.1 s and consumes 54.5 mJ for computation only. This means that HBQ scheme takes 130 times longer in authentication time and 80 times more expensive in energy consumption than symmetric-key based schemes. This makes HBQ scheme unrealistic to be employed in practice.
- **It does not provide mutual authentication**: In the scheme, *Alice* authenticates to sensor s_l, but s_l does not authenticate to *Alice*. In many cases, it is necessary to authenticate sensors to ensure that *Alice* receives correct information from a legitimate node. As an example, consider a battlefield scenario where the officer wants to make sure that detecting an alert of an enemy tank must be originated from a legitimate node.
- **It is vulnerable to DoS attacks**: In the second step, upon receiving T_A from *Alice*, sensor s_l must perform three ECC point multiplications, one XOR, and one symmetric encryption. Each ECC point multiplication on *TelosB* mote 8 MHz takes 3.5 s and consumes significant energy. An attacker could easily launch DoS attacks by sending a forged T_A to s_l and could rapidly deplete s_l's energy.

$$
\begin{aligned}
Alice \rightarrow s_l\!: &\quad T_A = (C_A||ac_A)\\
s_l \text{ computes}\!: &\quad Q_A = e_A C_A + Q\\
: &\quad \text{picks a random } r \in GF(q)\\
: &\quad Z_r = H(r)Q_A\\
: &\quad Y_r = H(r)P\\
: &\quad z_r = r \oplus (Z_r)\\
: &\quad \mathrm{MAC}(r, N_A)\\
s_l \rightarrow Alice\!: &\quad z_r, Y_r, \mathrm{MAC}(r, N_A)\\
Alice \text{ computes}\!: &\quad q_A Y_r = q_A H(r)P = Z_r\\
: &\quad r = X(Z_r) \oplus z_r\\
: &\quad \text{decrypts } \mathrm{MAC}(r, N_A)\\
Alice \rightarrow s_l\!: &\quad \mathrm{MAC}(r, N_A||ac_A)\\
s_l \rightarrow Alice\!: &\quad \mathrm{MAC}(r, reply)
\end{aligned}
$$

Fig. 2. HBQ protocol.

III. ASSUMPTIONS AND ADVERSARY MODEL

We assume a sensor network with a large number of nodes deployed in a variety of environments such as a battlefield, a forest, or office building. Each node, e.g., MICA2 mote, is strictly resource-constrained in terms of energy, memory, computational capacity and communication bandwidth. Sensors are responsible for collecting, reporting and providing information and services to users through a wireless channel. For the sake of energy conservation, sensors may collaborate with each other to perform in-network data aggregation, and only one representative node reports information to users. The sensor network is managed by a base station or a KDC, which is responsible for generating all security primitives, issuing and revoking users' access privileges. KDC is trusted and stays online all the time. This assumption is reasonable in the sense that we can de-

Table 1. Notations.

Symbol	Description
ID_A:	Identifier of entity A
k_A, Q_A:	A pair of ECC private and public keys of entity A
ac_A:	Access control list issued to entity A
$\text{sign}_A(m)$:	Message m is signed by entity A
$A \rightarrow B : m$:	Entity A sends entity B a message m
$(m)K$:	Symmetric encryption of message m with key K
MAC(K, m) :	Message authentication code of m with key K
$h(m)$:	Hashing value of message m
$\|$:	Concatenation
$\times$:	ECC point multiplication

ploy one or more replica servers so that if one of them is down, another server can act as an online KDC. Mechanisms of how those replicated KDC servers work are beyond the scope of this paper. Users are equipped with a more powerful device such as a laptop or a PDA to query information from the sensor network. To access the network, users need to apply for access permissions from KDC. KDC maintains a user access list pool and associated user identifiers. Sensors only use omni-directional wireless communication and are often deployed in hostile environments. Therefore, they are vulnerable to many attacks. We assume that if a defender can deploy many sensor nodes, then an adversary can also deploy a few malicious nodes with similar hardware capabilities as legitimate nodes. The adversary can use these malicious nodes to eavesdrop messages sent out or received by normal nodes or even inject false reports to the users [6], [7]. The adversary may also launch *Sybil* attack [8] (i.e., a malicious node illegitimately takes on multiple identities) to report wrong data to users. On the other hand, the adversary may use a fake user identifier as a legitimate one to access the network. It can also eavesdrop on the first protocol step and then replays it in an arbitrary part of the network. Moreover, the adversary can use a powerful device to launch DoS attack [9] by continuously sending requests to the sensor network so that sensors continue verifying the authority of the requests, and thereby deplete their energy quickly.

IV. PROPOSED SCHEME

Initially, KDC performs a number of basic operations as HBQ to generate a base point P, a private key k_{KDC} and a corresponding public key $Q_{\text{KDC}} = k_{\text{KDC}}P$. KDC also generates private-public keys for each sensor node. A sensor can physically connect to KDC via a wired connection by either a USB cable or a serial connector. To issue a private-public key pair for a sensor S with identifier ID_S, KDC picks up a random number $k_S \in GF(p)$ and computes $Q_S = k_S P$, where k_S is the private key assigned to sensor S and Q_S is the public key. Each sensor also has a public key Q_{KDC} of KDC which is preloaded. Notations used are explained in Table 1.

A. Key Agreement

After deployment, each sensor computes a shared secret key with KDC for afterward authentication and access control process. The proposed scheme is based on elliptic curve Diffie-Hellman (ECDH) [10] to establish a key agreement between each sensor node and KDC. ECDH is a key agreement protocol allowing two parties to establish a shared secret key that can be used for private key algorithms. It has been shown that ECDH with 160 bit key size can achieve the same security level with the conventional 1024 bits Diffie-Hellman secret sharing protocol [3]. To reduce energy consumption, public keys between KDC and sensors are mutually exchanged before the network deployment. As such, when KDC generates a public key Q_S and loads into each sensor node, it also loads its public key Q_{KDC} to the sensor. So KDC stores public key of all sensors and each sensor stores KDC's public key in advance, and thus no communication to exchange public key is required. If a new node is deployed, it is first loaded with its keys and KDC's public key in the same way before being deployed into the sensor field. To establish a shared secret key with KDC, a sensor node, say S, computes $R_S = (x_S, y_S) = k_S Q_{\text{KDC}}$. KDC also computes $R_{\text{KDC}} = (x_{\text{KDC}}, y_{\text{KDC}}) = k_{\text{KDC}} Q_S$. Since $k_S Q_{\text{KDC}} = k_S k_{\text{KDC}} P = k_{\text{KDC}} Q_S$, therefore $R_S = R_{\text{KDC}}$ and hence $x_S = x_{\text{KDC}}$. As the result, x_S is used as a shared secret key between node S and KDC.

B. Key Renewal

The shared secreted key x_S must be renewed frequently to avoid security risks. To renew the secret key, KDC selects a new ephemeral private random number k'_{KDC}, generates a corresponding public key Q'_{KDC} and broadcasts Q'_{KDC} to all the nodes. Each node computes $R'_S = (x'_S, y'_S) = k_S Q'_{\text{KDC}}$ and obtains a new shared secret key x'_S. KDC also computes $R'_{KDC} = (x'_{\text{KDC}}, y'_{\text{KDC}}) = k'_{\text{KDC}} Q_S$ and obtains a new shared secret key x'_{KDC}. Obviously, it is $x'_S = x'_{\text{KDC}}$.

C. Protocol Description

Prior to accessing the network, user *Alice* (A) comes to KDC and gets her public key (Q_A) and private key (k_A) through a secure channel. There are several ways to setup a secure channel between *Alice* and KDC. One of the simple ways is that *Alice* (e.g., *Alice* is using a PDA) directly connects to KDC using a wired connection such as USB cable. In case she cannot come to the KDC, she can connect to any trusted computer which has a wired Internet connection to connect to KDC via a secure channel such as secure socket layer (SSL). KDC can generate a private key k_A and export to *Alice*'s device. This is not possible for an attacker to 'intercept' or 'eavesdrop' the message to get the key. Based on *Alice*'s request and background check, KDC issues a proper access control list ac_A. This list has the same structure as HBQ scheme [4] (see Fig. 1). KDC generates a certificate of the list and *Alice*'s public key by signing with its private key ($cert_A = \text{sign}_{\text{KDC}}(ac_A \| Q_A)$. The certificate is then sent to *Alice*. Both *Alice* and KDC also compute a shared secret key x_A in the same way.

The authentication and access control protocol is described in Fig. 3. Supposed *Alice* wants to access the sensor *S*. The protocol

$$
\begin{aligned}
\textit{Alice}\text{ computes: }\ & L = h(x_A \oplus T_A)\\
:\ & S_1 = \text{sign}_A((r)L||cert_A)\\
\textit{Alice} \to S\text{: }\ & (r)L, T_A, S_1\\
S\text{ computes: }\ & \text{MAC}_1 = \text{MAC}(x_S, (r)L||T_A||S_1)\\
S \to \text{KDC: }\ & (r)L, T_A, S_1, \text{MAC}_1\\
\text{KDC computes: }\ & \text{check if } T_A \text{ is valid?}\\
:\ & \text{verify}(\text{MAC}_1), \text{verify}(S_1),\\
:\ & \text{verify}(cert_A), L = h(x_A \oplus T_A),\\
:\ & r = \text{decrypt}((r)L),\\
:\ & M = h(x_S \oplus T_{\text{KDC}}),\\
:\ & \text{MAC}_2 = \text{MAC}(x_S, (r)M||ID_A))\\
\text{KDC} \to S\text{: }\ & (r)M, T_{\text{KDC}}, ID_A, \text{MAC}_2\\
S\text{ computes: }\ & \text{check if } T_{\text{KDC}} \text{ is valid?}\\
:\ & \text{verify}(\text{MAC}_2)\\
:\ & M = h(x_S \oplus T_{KDC}),\\
:\ & r = \text{decrypt } ((r)M)\\
:\ & \text{MAC}_3 = MAC(r, ID_S)\\
S \to \textit{Alice}\text{: }\ & \text{MAC}_3\\
\textit{Alice}\text{ computes: }\ & \text{verify } (\text{MAC}_3)
\end{aligned}
$$

Fig. 3. ENABLE protocol.

includes the following steps.

- **Step 1**) $Alice \to S$: $(r)L, T_A, S_1$
  Alice selects a random number $r \in GF(p)$ which will be used as a session key with S, creates a secret key $L = h(x_A \oplus T_A)$ (where T_A is the current timestamp generated by *Alice*), and encrypts r with key L, $(r)L$. *Alice* then signs this encrypted value along with its certificate ($S1 = \text{sign}_A((r)L||cert_A)$) and sends to the sensor S.
- **Step 2**) $S \to \text{KDC} : (r)L, T_A, S_1, \text{MAC}_1$
  Upon receiving the message from *Alice*, S first checks if the time T_A is valid. If yes, then it builds a MAC by the shared secret key x_S ($\text{MAC}_1 = \text{MAC}(x_S, (r)L\|T_A||S1)$). The sensor then forwards the message along with MAC_1 value to KDC.
- **Step 3**) $\text{KDC} \to S : (r)M, T_{\text{KDC}}, ID_A, \text{MAC}_2$
  Upon receiving the message from S, KDC verifies MAC_1 value. If the verification is successful, then S is authentic to KDC. KDC then verifies S_1 which was signed by *Alice*. If the signature is valid, then *Alice* is also authentic. The $cert_A$ is also verified to check the validity of the access list ac_A. KDC now constructs a secret key $L = h(x_A \oplus T_A)$, and decrypts $(r)L$ to get r. It then generates a secret key $M = h(x_S \oplus T_{\text{KDC}})$ (where T_{KDC} is the timestamp created by KDC), encrypts r, and builds a MAC ($\text{MAC}_2 = \text{MAC}(x_S, (r)M\|ID_A)$). Afterward, KDC sends them to S.
- **Step 4**) $S \to Alice$: MAC_3
  When S receives the message, it verifies MAC_2 value. If it is valid, it indicates that *Alice* is authentic to S. After that, S constructs the secret key $M = h(x_S \oplus T_{\text{KDC}})$ and decrypts $(r)M$ to get r. Using this secret key, S builds a MAC ($\text{MAC}_3 = \text{MAC}(r, ID_S)$) and sends to *Alice*.

Upon receiving the MAC value from S, *Alice* verifies it by the same key r. If the verification is successful, then S is authentic to the user.

V. SECURITY ANALYSIS

This section presents security analysis of the proposed scheme and shows how it overcomes the aforementioned problems.

A. ENABLE Provides Mutual Authentication

In step 3 of the ENABLE protocol, KDC verifies the signature S_1. If S_1 is valid, then the user is authentic to KDC because only the user can generate the signature S_1 by his private key. Consequently, the user is also authentic to sensor S because S trusts KDC (step 4). On the other hand, only S shares the secret key x_S with KDC. It means that only S can decrypt $(r)M$ (where $M = h(x_S \oplus T_{\text{KDC}})$). So if S can achieve r from $(r)M$ to build MAC_3 ($\text{MAC}_3 = \text{MAC}(r, ID_S)$), then S is authentic to the user. The mutual authentication is provided through trust relations between *Alice*-KDC, and S-KDC.

B. The Proposed Scheme Can Defend against Replay Attacks

There are two possible ways for an adversary to launch replay attacks as follows:

- The adversary can intercept the message sent out from *Alice* (step 1) or from the sensor S (step 2). However, both cases are not possible in ENABLE because KDC can easily detect by verifying timestamp T_A (step 3). If T_A is older than a predefined threshold, it is invalid because it has been used for previous authentication. If T_A was changed, then S_1 ($S_1 = \text{sign}_A((r)L||cert_A)$, where $L = h(x_A \oplus T_A)$) is not valid.
- The adversary can intercept the message sent out from KDC (step 3) or from the sensor S (step 4). In the former case, node S can detect by checking timestamp T_{KDC}. If T_{KDC} is older than the predefined threshold, it is not valid. If T_{KDC} was changed to T^*_{KDC}, then the MAC^*_2 value ($\text{MAC}^*_2 = \text{MAC}(x_S, (r)M\|ID_A)$, where $M = h(x_S \oplus T^*_{\text{KDC}})$ is not consistent with received MAC_2. In the latter case, *Alice* can easily detect the replayed message by verifying MAC_3. *Alice* builds a MAC value ($\text{MAC}'_3 = \text{MAC}(r, ID_S)$) and compares with the received MAC_3. If $\text{MAC}_3 = \text{MAC}'_3$, then Alice knows that MAC_3 is not modified.

C. The Proposed Scheme Can Defend against DoS Attack

Upon receiving the message from the user (step 2), the sensor first check the timestamp T_A if it is valid. It then builds a MAC using a very fast Message Authentication Code algorithm such as CBC-MAC [16] and forwards the message to KDC. A CBC-MAC operation on MICA2 mote takes 3.12 ms [11], which is very fast and lightweight compared with ECC point multiplications used by HBQ (which in total takes 3,500 ms, about 1121 times longer). Therefore, the proposed scheme significantly reduces DoS compared to HBQ. The attacker also may launch the DoS on the KDC by sending so many fake requests with a valid timestamp to the sensor, so that the sensor forwards all fake requests to the KDC in order to make KDC busy. One of the simple solutions is that if there are so many frequent requests (e.g., tens or hundreds) at the same time or constantly, the sensor will not forward the request to the KDC to avoid any possible DoS and reduce traffic congestion. On the other hand, before the KDC performs ECC operations (i.e., verifying $cert_A$), it first computes a MAC value and compares with the received MAC_1. If the MAC_1 is not valid, it will decline the message and will not perform ECC operations. A CBC-MAC operation is very lightweight. It only took 43.32 μs on 927 MHz Pentium II machine [13] which means that a normal machine (Pentium IV 3.2 MHz) can process millions of CBC-MAC operations at the same time. Therefore even though the attack sends so many fake requests, it would not be able to make KDC busy at all.

VI. PERFORMANCE EVALUATION

This section presents the performance evaluation of the proposed scheme in terms of computational and communication costs and compare with HBQ and symmetric-key based schemes.

A. Computational Cost

Since user's equipment and KDC are powerful devices, the computational overhead is trivial compared to that of the sensors. Therefore, we only consider computation and communication overhead for sensors. We use the computational overhead (the computation time required by sensors, denoted by T) to analyze the performance of ENABLE. Notations are defined as follows:

T_H: Time to perform one-way hash function (e.g., SHA-1).

T_{MAC}: Time to generate MAC value (e.g., CBC-MAC).

T_{RC5}: Time to encrypt or decrypt by RC5 (note: Encryption and decryption take almost same duration of time [14]).

T_{MUL}: Time to perform ECC point multiplication.

According to practical implementations on MICA2 motes [5], [11], [14], [17], the computational time is mentioned in Table 2. Using this evaluation, the total computational time of the proposed scheme, HBQ and symmetric-key based schemes are shown in Table 3. For user authentication in HBQ scheme, it requires $2T_H$, $2T_{\mathrm{MAC}}$, $2T_{\mathrm{RC5}}$, and $3T_{\mathrm{MUL}}$ (total cost is approximately 2,451.04 ms). Meanwhile, ENABLE requires only T_{MAC} (approximately 3.12 ms). For node authentication, HBQ does not support, while our scheme requires $2T_{\mathrm{MAC}}$, T_{RC5} and T_H (approximately 10.136 ms). For the symmetric key cryptography access control, we follow the same approach used in [4], so user authentication costs $2T_H$, and $2T_{\mathrm{MAC}}$ (approximately 6.756 ms). In total, ENABLE takes only 13.256 ms for both user and node authentication that is much faster than HBQ scheme (which takes 2,451.04 ms) and only two times slower than symmetric key cryptography based approaches (which takes 6.75 ms).

Table 2. Execution time of security primitives.

Operation	Time (ms)
T_H	3.636
T_{MAC}	3.12
T_{RC5}	0.26
T_{MUL}	810

Table 3. Comparison of computational cost.

	ENABLE	**HBQ**	**Sym. key based schemes**
User authentication	T_{MAC} $\simeq 3.12$ (ms)	$2T_H+2T_{\mathrm{MAC}}$ $+2T_{\mathrm{RC5}}+3T_{\mathrm{MUL}}$ $\simeq 2,451.04$ (ms)	$2T_H+2T_{\mathrm{MAC}}$ $\simeq 6.75$ (ms)
Node authentication	$2T_{\mathrm{MAC}}+T_{\mathrm{RC5}}$ $+T_H$ $\simeq 10.136$ (ms)	none	none
Total	$2T_{\mathrm{MAC}}+T_{\mathrm{RC5}}$ $+T_H$	$2T_H+2T_{\mathrm{MAC}}$ $+2T_{\mathrm{RC5}}+3T_{\mathrm{MUL}}$	$2T_H+2T_{\mathrm{MAC}}$
Total time	**13.256 (ms)**	**2,451.04 (ms)**	**6.75 (ms)**

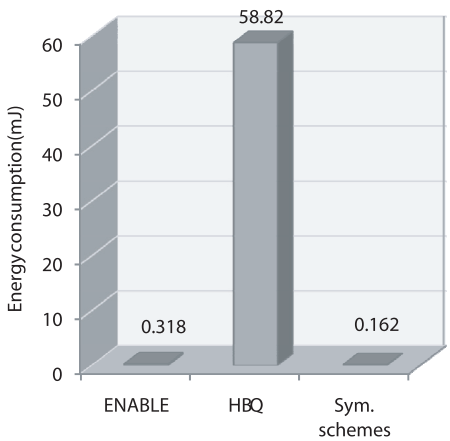


Fig. 4. Comparison of energy consumption.

We used the formula $E = UIt$ to estimate the energy consumption of security computations [13], [14]. For MICA2 mote, when processor is in active mode, $I = 8$ mA. Typically, $U = 3.0$ V if two new AA batteries are used [14]. Total energy consumption is shown in Fig. 4. ENABLE requires only 0.381 mJ of the sensor node to perform access control operations that is 184 times less than HBQ (58.82 mJ). Also, ENABLE consumes energy that is only twice more than symmetric key based schemes (0.161 mJ).

B. Communication Cost

To evaluate the communication cost, we simulated ENABLE, HBQ, and symmetric-key based schemes on SENSE simulator [12]. 300 sensor nodes are randomly deployed in a network field of 2000 m $\times$ 2000 m. For routing mechanism, we selected ad hoc on-demand distance vector (AODV) protocol. At the MAC layer, we used IEEE 802.11 distributed coordination function

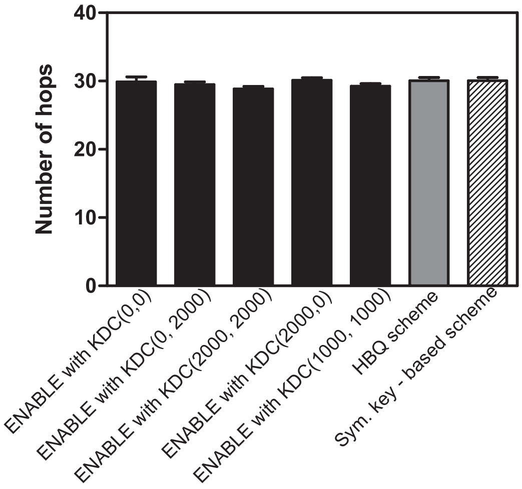


Fig. 5. Comparison of communication cost.

(DCF). Two-ray ground [18] was used as the radio propagation model. For ENABLE, HBQ, and symmetric-key based scheme, we use the same network deployment. For ENABLE, we varied KDC's location within the network. We placed KDC at one of the four corners ((x, y) coordinates were [0,0], [0,2000], [2000,2000], and [2000,0]) or at the center of the network field ([1000,1000]). For each KDC location, we ran the simulation 1,000 times in which *Alice*'s location and sensor S's location are randomly selected at each run. HBQ and symmetric-key based scheme also ran 1,000 times with the same *Alice*'s location and sensor S's locations as ENABLE.

We measured the average number of hops that the access control packets are transmitted through for each mechanism. The result is shown in Fig. 5. It indicates that for all KDC locations (either at the corner or center of the network field), the communication cost of ENABLE is almost similar to the HBQ and symmetric key based schemes. As the simulation results are shown in the Fig. 5, location of KDC does not affect to the overall performance of the proposed mechanism when a large number of user accesses are requested. As an example shown in Fig. 6, communication cost of ENABLE is $2(AB + BC)$, while it is $4AB$ in HBQ or symmetric-key based scheme. If $BC < AB$ (KDC is closer to the sensor S than *Alice*), then ENABLE has less communication cost than HBQ or symmetric-key based schemes. However, if $BC > AB$ (*Alice* is closer to S than KDC), then ENABLE brings more communication cost than others. This example leads to a solution to reduce the communication cost of ENABLE by deploying various KDC within the network so that each node S can communicate with the closest KDC to reduce overall cost.

VII. CONCLUSION AND FUTURE WORK

Public-key cryptography based access control scheme has more advantages than symmetric-key cryptography based scheme because of better scalability, low memory requirement, easy deployment of new nodes, and no key pre-distribution. HBQ is a promising public-key access control scheme based on elliptic curve cryptography but it is shown to have some major limitations. In this paper, we propose ENABLE as an access control scheme based on elliptic curve cryptography which improves security and performance of HBQ. Through analysis and evaluation, we have shown that ENABLE overcomes security limitations in HBQ, perform better than HBQ, and is comparable in performance to symmetric-key based schemes (184 times less energy consumption than HBQ, and only twice more than symmetric key based schemes).

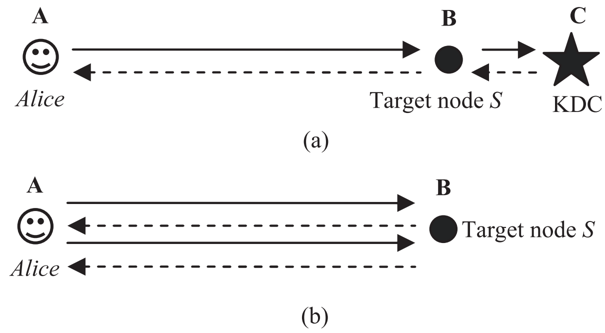


Fig. 6. Communication of ENABLE and HBQ/Sym. schemes; (a) ENABLE communication and (b) HBQ/Sym. scheme communication.

For future work, we plan to implement the proposed scheme on MICA2 motes to observe a practical performance and show precise comparison with other schemes. Although we have suggested a simple solution to get the KDC online all the time, it raises another issue that all the KDC may be down and there will not have any KDC to carry out the access control process. Therefore, a solution to get rid of the online KDC will be researched and proposed.

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] J. Burrell, T. Brooke, and R. Beckwith, "Vineyard computing: Sensor networks in agricultural production," *IEEE Pervasive Comput.*, vol. 3, no. 1, pp. 385, Jan–Mar. 2004.

[3] Z. Benenson, N. Gedicke, and O. Raivio, "Realizing robust user authentication in sensor networks," in *Proc. Workshop on Real-World Wireless Sensor Networks*, 2005.

[4] H. Wang, B. Sheng, and Q. Li, "Elliptic curve cryptography-based access control in sensor networks," *Int. J. Security and Networks*, vol. 1, nos. 3/4, pp. 127–137, 2006.

[5] N. Gura, A. Patel, A. Wander, H. Eberle, and S.C. Shantz, "Comparing elliptic curve cryptography and RSA on 8-bit CPUs," in *Proc. CHES*, 2004, vol. 3156, LNCS, pp. 119–132.

[6] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," in *Proc. SNPA*, May 2003, pp. 113–127.

[7] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: Security protocols for sensor networks," *Wireless Netw.*, vol. 8, pp. 521–534, Sept. 2002.

[8] J. Newsome, E. Shi, D. Song, and A. Perrig, "The sybil attack in sensor networks: Analysis and defenses," in *Proc. Intl. Symp. Inf. Process. Sensor Networks*, USA, Apr. 2004, pp. 259–268.

[9] A. Wood and J. Stankovic, "Denial of service in sensor networks," *IEEE Computer*, pp. 54–62, Oct. 2002.

[10] ANSI X9.63, *Elliptic Curve Key Agreement and Key Transport Protocols*, American Bankers Association, 1999.

[11] Z. Benenson, L. Pimenidis, F. Freiling, and S. Lucks, "Authenticated query flooding in sensor networks," in *Proc. the 4th IEEE Conf. Pervasive Comput. Commun. Workshops*, Pisa, Italy, 2006, pp. 644–647.

[12] G. Chen, J. Branch, M. J. Pflug, L. Zhu, and B. Szymanski, *SENSE: A Sensor Network Simulator, Advances in Pervasive Computing and Networking*, NY, USA: Springer, 2004, pp. 249–269.

[13] J. Deepakumara, H. M. Heys, and R. Venkatesan, "Performance comparison of message authentication code (MAC) algorithms for Internet protocol security (IPSEC)," in *Proc. Newfoundland Electrical and Computer Engineering Conf.*, St. John's, Newfoundland, Nov. 2003.

[14] C. Karlof, N. Sastry, and D. Wagner, "TinySec: A link layer security architecture for wireless sensor networks," in *Proc. SensSys*, Nov. 2004, pp. 162–175.

[15] A. Liu and P. Ning, "TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks," North Carolina State University, Department of Computer Science, Tech. Rep. (TR-2007-36), Nov. 2007.

[16] A. Menezes, P. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography.*, FL, USA: CRC Press, 1997.

[17] G. Prasanth, V. Ramnath, P. Pushkin, G. D. Alexander, M. Frank, and L. S. Mihail, "Analyzing and modeling encryption overhead for sensor network nodes," in *Proc. Second ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, Sept. 19, 2003, pp. 151–159.

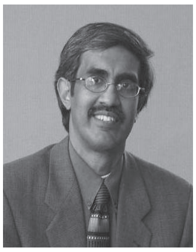
[18] T. S. Rappaport, *Wireless Communications, Principles, and Practice*, Prentice Hall, 1996.

Xuan Hung Le received B.S. degree in Computer Science from Hanoi University, Vietnam, in 2003, M.S., Ph.D. degree in Computer Engineering from Kyung Hee University, Korea, in 2005 and 2008, respectively. From 2002 to 2003, he served as a software engineering at FPT corp., Vietnam. From Dec. 2008 to Aug. 2009, he was a Research Professor at the Department of Computer Engineering, Kyung Hee University, Korea. Since Sept. 2009, he has been a Postdoctoral Research Associate at the Department of Electrical Engineering, University of South Florida, USA. His research interests are wireless sensor networks, cryptography, and information security.

Sungyoung Lee received his B.S. in Material Science from Korea University in 1978, M.S., Ph.D. in Computer Science from Illinois Institute of Technology, USA in 1987, and 1991, respectively. Since 1993, he has been a Professor at the Department of Computer Engineering, College of Electronics and Information, Kyung Hee University, Korea. From 1992 to 1993, he was a Assistant Professor at the Department of Computer Science, Governors State University, University Park, USA. His research interest includes ubiquitous computing middleware, java virtual machine, real-time system, and embedded system.

Ismail Butun received his B.Sc. and M.Sc. degrees in Electrical and Electronics Engineering from Hacettepe University, Turkey in 2003 and 2006, respectively. He is currently pursuing his Ph.D. degree in Electrical Engineering at University of South Florida, USA. His research interests focus on the following areas: Cryptography, information security, digital signatures, and wireless sensor network security.

Murad Khalid received B.S. and M.S. degrees in Electrical Engineering from the City College of the City University of New York, USA, in 1993 and 1995, respectively. He is currently a Research Assistant working towards a Ph.D. degree in Wireless Communication and Networking area in the Department of Electrical Engineering at the University of South Florida. His research interests include mobile wireless communications and networks, with emphasis on cross-layer design, resource allocation and performance optimization of wireless ad hoc networks. He was with Bahria University as the Assistant Professor in Computer Engineering Department. Before that he served as Senior Systems Engineer for Motorola, Bosch Telecom, and Ericsson Radio Systems.



Ravi Sankar received the B.E. (Honors) degree in Electronics and Communication Engineering from the University of Madras, India in 1978, the M.Eng. degree in Electrical Engineering from Concordia University in 1980, and the Ph.D. degree in Electrical Engineering from the Pennsylvania State University in 1985. Since then, he has been with the Department of Electrical Engineering at the University of South Florida, Tampa. He is currently a USF Theodore and Venette Askounes-Ashford Distinguished Scholar Award winning Professor of Electrical Engineering, and Director of the Interdisciplinary Communications, Networking, and Signal Processing (iCONS) Research group (http://icons.eng.usf.edu) and Interdisciplinary Center of Excellence in Telemedicine (ICE-T). His main research interests are in the areas of wireless communications, networking, signal processing, and its applications.

Miso (Hyoung-IL) Kim received his M.S. in Computer Engineering from the Kyung Hee University, Korea in 1996. Currently, he is an Sensor Research Engineer in the telecommunication R&D Center at Samsung Electronics. His research interests are personal navigation and activity recognition for ubiquitous computing.

Manhyung Han received his B.S. and M.S. degree in Computer Engineering from Kyung Hee University, Korea, in 2005 and 2007, respectively. In 2008, he was a Visiting Researcher at University of Virginia, VA, USA. Currently, He is currently pursuing his Ph.D. degree in Computer Engineering at Kyung Hee University, Korea. His research interests are ubiquitous computing, context-aware middleware, activity recognition, autonomic computing, sensor middleware, embedded OS, and WSN.

Young-Koo Lee received his B.S., M.S., and Ph.D. in Computer Science from Korea Advanced Institute of Science and Technology (KAIST), Korea in 1988, 1994, and 2002, respectively. Since 2004, he has been an Assistant Professor at the Department of Computer Engineering, College of Electronics and Information, Kyung Hee University, Korea. From 2002 to 2004, he was a Post Doctoral Fellow Advanced Information Technology Research Center (AITrc), KAIST, Korea, and a Postdoctoral Research Associate at the Department of Computer Science, University of Illinois at Urbana-Champaign, USA. His research interests are ubiquitous data management, data mining, activity recognition, bioinformatics, on-line analytical processing, data warehousing, database systems, spatial databases, and access methods.

Heejo Lee received his B.S., M.S., and Ph.D. in Computer Science and Engineering from Pohang University of Science and Technology (POSTECH), Korea in 1993, 1995, and 2000, respectively. Since 2004, he has been an Assistant Professor at the Department of Computer Science and Engineering, Korea University, Korea. From 2001 to 2003, he was at Ahn Lab, Inc. as Chief Technology Officer (CTO) and Director of Technology Planning Department. From 2000 to 2001, he was a Post Doctoral Research Associate at the Network Systems Lab of the Department of Computer Sciences and at the Center for Education and Research in Information Assurance and Security (CERIAS), Purdue University. His research interest includes computer and communication security, parallel scientific computing, and fault-tolerant computing.