Secure and Lightweight Network Admission and Transmission Protocol for Body Sensor Networks

Daojing He, Chun Chen, Sammy Chan, Jiajun Bu, and Pingxin Zhang

***Abstract*—A body sensor network (BSN) is a wireless network of biosensors and a local processing unit, which is commonly referred to as the personal wireless hub (PWH). Personal health information (PHI) is collected by biosensors and delivered to the PWH before it is forwarded to the remote healthcare center for further processing. In a BSN, it is critical to only admit eligible biosensors and PWH into the network. Also, securing the transmission from each biosensor to PWH is essential not only for ensuring safety of PHI delivery, but also for preserving the privacy of PHI. In this paper, we present the design, implementation, and evaluation of a secure network admission and transmission subsystem based on a polynomial-based authentication scheme. The procedures in this subsystem to establish keys for each biosensor are communication efficient and energy efficient. Moreover, based on the observation that an adversary eavesdropping in a BSN faces inevitable channel errors, we propose to exploit the adversary's uncertainty regarding the PHI transmission to update the individual key dynamically and improve key secrecy. In addition to the theoretical analysis that demonstrates the security properties of our system, this paper also reports the experimental results of the proposed protocol on resource-limited sensor platforms, which show the efficiency of our system in practice.**

***Index Terms*—Body sensor networks (BSNs), efficiency, key update, network admission and transmission, security.**

I. Introduction

RECENTLY, with the rapid development in biosensors and wireless communication technologies (e.g., Bluetooth and Zigbee), wireless body sensor networks (BSNs) (also called body area networks or medical sensor networks) have emerged as a promising technique for pervasive monitoring of patients' personal health information (PHI) (see, e.g., [1]). Instead of being measured face-to-face, with BSNs, patients' PHI can be monitored remotely, continuously, and in real time, and then processed and transferred to healthcare centers. A BSN is a wireless network of mainly implanted or wearable biosensors designed to deliver PHI to a local processing unit (e.g., tablet PC, laptop PC, and smartphone), which is referred to as the personal wireless hub (PWH) [2].

Security and privacy for a BSN is very important, because the data collected are directly associated with a particular patient, which play a critical role in medical diagnosis and treatment. Due to the open and dynamic nature of BSNs, they are subject to various cyber attacks such as malicious modification. Failure to obtain authentic and correct medical data will possibly prevent a patient from being treated effectively, or even lead to wrong treatments. If patients' privacy is not strongly protected, their health data can be misused and the public acceptance of BSN is significantly hindered. Therefore, BSN applications must meet a set of mandatory privacy requirements of healthcare alliances such as HITRUST [3] and legal directives such as those adopted in the U.S. [4] and Europe [5]. Ensuring security and privacy in BSNs is important for all walks of life [6]. For an ordinary patient, his/her medical sensor data may be useful to some parties such as insurance companies. An adversary may profit financially by selling these data obtained through eavesdropping on the BSN. For a patient with an important status, such as a country's top administrator, an adversary may target to harm him physically by misreporting or spoofing his/her medical sensor data, resulting in improper diagnosis and/or treatment.

Also, hacking the information transmitted in a wireless BSN might be much easier than hacking the information collected at a server due to the open and dynamic nature of a BSN. More importantly, compared to a server, physical compromise of a biosensor node is much easier. To address the above issues, it is vital to provide secure network admission and transmission as follows:

1) *Secure network admission:* It restricts network admission only to eligible PWHs and biosensors.
2) *Secure transmission:* It provides confidential, authenticated, and integrity-protected transmission between each biosensor and PWH.

However, designing a secure network admission and transmission protocol for BSNs is not an easy task. Generally, there are three major practical issues challenging the design. First, such a solution should take into account the rather limited memory space and computational capability available in biosensor nodes. Especially, its energy consumption should be minimal since biosensor nodes are powered by small batteries and required to operate for a long period of time. As a consequence, any security mechanism for BSNs should be carefully designed

Manuscript received July 11, 2012; revised October 14, 2012; accepted December 10, 2012. Date of publication December 21, 2012; date of current version May 1, 2013. This work was supported by Scholarship Award for Excellent Doctoral Student granted by Ministry of Education, National Science Foundation of China under Grant 61070155, Program for New Century Excellent Talents in University under Grant NCET-09-0685, and a grant from the Research Grants Council of the Hong Kong (Project No. City U 111208).

D. He, C. Chen, J. Bu, and P. Zhang are with the Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, Zhejiang 310027, China (e-mail: hedaojinghit@gmail.com; chenc@zju.edu.cn; bjj@zju.edu.cn; pingxin409@gmail.com).

S. Chan is with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong (e-mail: eeschan@cityu.edu.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JBHI.2012.2235180

TABLE I
IMPLEMENTATION RESULTS OF 1024-BIT RSA AND 160-BIT ECC CRYPTOSYSTEMS ON MICAZ AND TELOSB MOTES

		TelosB mote	MicaZ mote
processor		16-bit, 8 MHz MSP430 microcontroller	8-bit, 8 MHz Atmel microcontroller
RAM		10 KB	4 KB
ROM		48 KB	128 KB
160-bit ECC	Signature generation	1.55 s	1.35s
	Signature verification	2.25 s	1.96 s
1024-bit RSA	Public key operation	-	0.79 s
	Private key operation	-	21.5 s

so that it does not put too much burden on the already constrained biosensor resources. Second, a key design criterion is to minimize delays introduced by the security mechanism in order to comply with BSN latency requirements. Moreover, due to the limited bandwidth available in a BSN, low communication overhead is required. For example, secure BSN setup must be carried out in less than 1 s and the maximum allowable latency for ECG (electrocardiogram) transmission is 250 ms [7]. Emergency situations in a BSN require the capability for fast medical reaction without disabling security. Cryptographic algorithms used by these nodes must be, therefore, computationally efficient in order to satisfy these requirements. Moreover, if the message authentication or encryption/decryption mechanisms are not fast enough, an adversary may launch a denial-of-service (DoS) attack to exhaust the resources of legitimate nodes and render them less capable of carrying out their intended functions. Unfortunately, most commonly used cryptographic techniques (e.g., public key cryptosystems) are inapplicable in BSNs. This is evidenced by Table I, which presents the implementation results of RSA and elliptic curve cryptography (ECC) public-key cryptosystems of the WM-RSA and WM-ECC library [8] on two popular sensor nodes: MicaZ and TelosB motes. For example, ECC signature generation and verification on a MicaZ mote already take 1.35 and 1.96 s, respectively. Third, compared to the generic wireless sensor networks (WSNs) (often deployed in remote and inhospitable areas (e.g., forest) for environment monitoring), it is easier to launch node compromising attacks for a BSN due to its open, dynamic, and small-scale nature. Hence, resilience against physical compromise of a node is a basic security requirement for BSNs.

Security research in BSNs is still in its early stage, especially with respect to secure network admission and transmission. More specifically, to the best of our knowledge, until now no secure network admission method for BSNs has been proposed. Additionally, it is usually not a good practice to use a fixed individual key to secure the PHI transmission for a long period of time. First, a single encryption key will provide a large amount of cipher text for the adversary to attempt to crack. Second, if the encryption key is compromised, all previously transmitted data with the same key are also compromised. In the literature, no efficient protocol has been proposed to update the individual key in wireless networks. For example, the session key exchange in most existing key update schemes does not improve security against key leakage. Although some techniques have been proposed to secure BSNs, we observe that there are some security weaknesses and efficiency problems. Also, despite the significant progress in securing WSNs (see, e.g., [9] and [10]) and mobile ad hoc networks (MANETs) (see, e.g., [11] and [12]), they are not applicable to BSNs due to the fact that biosensors operate with extremely stringent constraints. The detailed analysis to arrive at the above conclusions will be given in Sections II and IV. This makes the issue more serious given the trend that more and more BSNs are being deployed.

To address the above challenges, this paper makes three main contributions:

1) Motivated by the observation that the scale of each BSN is very small (i.e., from several to tens of biosensors), we employ a polynomial-based authentication scheme to develop a secure network admission and transmission subsystem in BSNs to provide node authentication and support the establishment of two types of keys for each biosensor node—an individual key shared with PWH and a pairwise key shared with another biosensor node. Further, to reduce the computation and communication overhead, some additional mechanisms such as subkeyed hash function and the hardware-implemented advanced encryption standard (AES) symmetric algorithm are incorporated into the design of the proposed system.
2) An adversary eavesdropping on the wireless communication channel faces inevitable channel errors and hence would miss some information transmitted in the channel, especially in BSNs due to their unique characteristics. Therefore, we propose the eavesdrop-bounded adversary model for BSNs to capture the essence of incomplete eavesdropping for the first time in the literature. In this model, the adversary cannot eavesdrop all communications of the biosensor nodes. Based on this model, instead of trying to defend against every key leakage possibility, this paper proposes to exploit the adversary's uncertainty regarding the PHI transmission to update the individual key dynamically and improve key secrecy. That is, the update of an individual key is based on the secrets, which are constantly extracted from the PHI collection process in real time. The updated key is then used to secure the future PHI transmissions from each biosensor to PWH. Recall that in a conventional setting, one operational mistake or a single vulnerability is sufficient to dismantle the entire security. This is the so-called single point of failure problem. The adversary predominates in this situation because it can choose any weakness to attack while the user needs to defend against all possibilities. With the above key update procedure, the adversary must fight against any factor that may cause information loss. In other words, it is the adversary who suffers from the single point of failure problem. From the adversary's point of view, as transmission goes on, the information loss accumulates

and the chance to recover the dynamic individual keys decreases.

3) In addition to the theoretical analysis that demonstrates the security properties of our system, this paper also reports the experimental results of the proposed protocol on resource-limited sensor nodes and laptop PCs, which show the efficiency of our system in practice. Accordingly, some suggestions on how to set the parameters of the proposed protocol are provided.

The rest of this paper is structured as follows. In Section II, we first survey and analyze the related work and then discuss their security weaknesses and efficiency problems. Section III presents the network and adversary models, as well as the unique features of BSNs. Section IV describes our proposed protocol. Section V provides theoretical analysis of the security properties of the proposed protocol. Section VI discusses some important issues. Section VII presents the implementation and experimental results of the proposed protocol via real sensor platforms. Finally, Section VIII concludes this paper.

II. Related Work

In the literature, various schemes have been proposed to address different aspects of securing BSNs. For example, recently, an architecture called "SNAP" (Sensor Network for Assessment of Patients) [13] has been proposed to address the security challenges faced by a WSN for wireless health monitoring. SNAP protects the privacy, authenticity, and integrity of medical data, with low-cost and energy-efficient mechanisms. However, we observe that the PHI from a biosensor node is transmitted to PWH in plaintext. Thus, an adversary can easily modify the PHI and/or inject polluted PHI into the network. Some researchers (see, e.g., [2], [14], and [15]) utilize physiological signals (e.g., heart rate interval, blood flow, and electrocardiography) obtained from the patient to enable biosensors to agree upon a pairwise symmetric key. However, they demand that each biosensor can measure the same physiological parameter type; this assumption is rather restrictive and makes this approach not suitable for many BSN applications. The authors of [15] propose to apply HMAC-MD5 on electrocardiography blocks to achieve the key agreement. However, MD5 is weak in collision resistance.

Based on public key cryptography, some novel mechanisms (see, e.g., [6] and [16]) have been proposed to ensure security of BSNs. The authors of [6] propose to use ECC to set up symmetric keys between sensor nodes and the base station, and RC5 block cipher for the symmetric encryption/decryption for protecting data confidentiality and integrity. However, they are computation inefficient and cannot fulfill the stringent delay requirements in BSNs due to the use of the public key cryptography. For instance, as reported in [6], the ECC key agreement takes 7.198 s on a Tmote Sky mote, which features a 16-bit, 8-MHz MSP430 processor. In [16], identity-based public key is used to encrypt all medical data, and the method balances security and privacy with accessibility.

Also, traditional cryptographic mechanisms do not suffice given the unique characteristics of BSNs, and the fact that

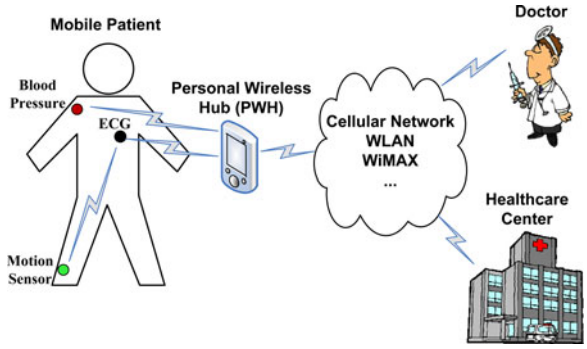


Fig. 1. System overview of a general BSN.

BSNs are susceptible to a variety of node misbehaviors. An application-independent and distributed trust evaluation model for BSNs has been proposed to identify malicious behaviors and then exclude malicious nodes [17]. Similar to most security schemes, trust management methods themselves can be vulnerable to attacks. To resolve this issue, an attack-resistant and lightweight trust management scheme has also been presented in [18].

Although there are a lot of works about generic WSNs and MANETs security (see, e.g., [9]–[12]), these mechanisms are not directly applicable in BSNs due to the unique and challenging operational and security requirements of BSNs. In particular, biosensors are limited in battery lifetime, computation, and communication capabilities, especially for implanted biosensors. For example, the random key scheme (see, e.g., [9]) is a major class of key establishment protocols for WSNs. For the efficiency of these schemes, the probability that each node shares at least one key with a neighboring node (referred to as key-sharing probability) should be high. When the key pool size is large, each sensor needs to preload a large number of keys to achieve a high key-sharing probability. Moreover, many keys are exchanged between sensor node pairs to establish the pairwise key and, inevitably, high communication, computation, energy overhead is incurred. Very recently, based on the random key scheme, a software design was presented for securing BSNs [19]. Unfortunately, these requirements are too demanding for the biosensor nodes. In [11], a self-contained public key management scheme has been proposed for wireless ad hoc networks, in which a small number of cryptographic keys are stored offline at individual nodes before deployment. Also, to avoid the weaknesses of a public key infrastructure (PKI), as a special form of public key cryptography, identity-based cryptography has been used in various areas of securing MANETs [12]. However, as described before, because of the use of the public key cryptography, they are computation inefficient, cannot fulfill the stringent delay requirements in BSNs, and are vulnerable to DoS attacks.

III. Network, Adversary Models, and Unique Features of BSNs

A. Network Model

As shown in Fig. 1, a BSN is a multihop wireless network of physiological and environmental monitoring biosensor nodes

that are worn and/or implanted on a patient. We assume that each biosensor does not have any information about their immediate neighboring nodes in advance. A BSN is operated by the BSN administrator (e.g., the patient himself, the patient's relative, eHealth service provider, or medical practitioner). The biosensors collect PHI at regular intervals and forward it to PWH. Then, PWH transmits the aggregated PHI to the remote healthcare center over different wireless networks such as cellular, WLAN, and WiMAX. We assume that the biosensors communicate with PWH wirelessly, as wires running in a BSN will make it obtrusive. The wireless medium is, however, not trustworthy. Note that, in this paper, we focus solely on securing the network admission and transmission within the BSN. Communication from PWH onwards can utilize conventional security mechanisms such as secure socket layer, given the considerable capabilities of the entities involved. All biosensor nodes in a BSN have limited power supply, memory space, and computational capability. Due to the constrained resources, computationally expensive and energy-intensive operations such as public key cryptography are not preferred for such nodes.

B. Adversary Model

Due to the sensitive nature of the data BSNs collect and the broadcast nature of the wireless medium, BSNs potentially face many threats. They are imposed by either outside or inside attackers. Outside attackers can eavesdrop messages, drop messages by jamming the communication channel, modify messages, inject forged messages, or replay old messages. Regarding eavesdropping, it is generally assumed that the adversary picks up all radio communications of the nodes without any loss. However, we suggest here that this is not the case due to the following three reasons. First, the wireless channel is inherently error prone. Second, the radio quality of a biosensor is not very good (see Section VI-A) and its coverage area is small because often its transmitting power is set to a low level to maximize the battery lifetime. Yet, in order to remain undetected, the adversary needs to keep a distance from the BSN. Third, it is extremely difficult for the adversary to predict a patient's movement and follow him/her everywhere. As a result, an outside attacker inevitably suffers from information loss. That is, the outside attacker picks up the radio communications of biosensor nodes with some loss. We refer such attackers as the *eavesdrop-bounded adversaries*. For inside attacks, the adversary may compromise PHW and a limited number of biosensor nodes to obtain their data and keying materials. Once a node is compromised, the adversary may discard its sensed data or packets received from other nodes. However, we assume that the BSN administrator will not be compromised.

In practice, the adversary could plant a rootkit or Trojan into a networked device. If the adversary can establish a link to directly retrieve unsecured data and have full control over the device, no security mechanism will work. Such a complete security breach is highly intrusive and susceptible to detection because the behavior of the victim's device is manipulated. In wireless communication environment, it is more often that the adversary steals the system secret and then uses the secret to decrypt the eavesdropped data or inject malicious messages. In such circumstances, using dynamic individual key significantly restricts the adversary. Also, we assume that the physical layer of a BSN could use techniques such as spread spectrum [20] to prevent physical jamming attack if necessary.

TABLE II
NOTATIONS

Notation	Description
$E(X, K)$	encrypting message X with a symmetric key K
$D(X, K)$	decrypting cipher text X with a symmetric key K
, or $\|$	concatenation operator of the two bit streams
$h(.)$	public one-way collision-resistant hash function (e.g., SHA-1)
$h(K, X)$	keyed hash function with a symmetric key K for message X

C. Unique Features and Application Requirements of BSNs

In this section, some differences between BSNs and MANETs (or generic WSNs) are listed as follows [17], [18]. 1) Data Rate: Many MANETs and WSNs are employed to monitor events which often happen at irregular interval. On the other hand, BSNs are employed for monitoring humans' physiological activities and actions, which may occur in a more periodic manner. Hence, the applications' data rates are relatively more steady. 2) Mobility: Even though a patient with a BSN may move around, all biosensor nodes in the network are static relative to the patient. 3) Effectiveness and efficiency: The signals that body sensors collect can be effectively processed to obtain reliable and accurate physiological estimations. Also, their ultra low power consumption makes their batteries long lasting. 4) Latency: This requirement is dictated by the applications and may be traded for improved security and energy consumption. However, while energy conservation is always important, replacement of batteries in BSNs nodes is much easier than in WSNs, whose nodes can be physically unreachable after deployment. Thus, it may not be necessary to maximize battery lifetime in a BSN at the expense of higher latency.

Besides, compared to MANETs and generic WSNs, a BSN is a network with small-scale structure and very short range of communications. Its nodes are limited in their power, computation, communication, and memory capabilities, especially for those implanted into the body.

IV. Proposed Protocol

Our proposed protocol consists of two phases. One is the network admission and transmission control, and the other is the individual key update. The notations used throughout this paper are listed in Table II.

A. Network Admission and Transmission Control

We propose a polynomial-based authentication scheme which is inspired by [21]. Before deployment, PWH and all biosensor nodes are loaded with a unique polynomial share of the same bivariate t-degree polynomial. Once deployed, each node identifies its neighboring nodes and PWH by mutual authentication and then establishes two kinds of secret keys. The detailed description is as follows.

We require that in the system initialization phase, for each BSN, the BSN administrator randomly generates a bivariate t-degree polynomial $f(x,y)=\sum_{i,j=0}^{t} a_{ij}x^i y^j$ over a finite field F_p, where p is a large prime number, such that it has the property of $f(x,y)=f(y,x)$. Also, the BSN administrator keeps the bivariate t-degree polynomial secretly. At the point of deployment, each biosensor node, say S_j, is embedded with a polynomial share of $f(x,y)$, that is, $f(sid_j,y)$ and authenticated at a secure place, where sid_j indicates the identity of node S_j. In the same way, PWH is embedded with a polynomial share of $f(x,y)$, that is, $f(pid_i,y)$, where pid_i indicates the identity of PWH. PWH will broadcast pid_i periodically to declare PWH service existence.

Our system supports the establishment of two types of keys for each biosensor node: an individual key shared with PWH and a pairwise key shared with its neighboring nodes. The details of establishing these keys are given as follows.

When a new biosensor, say S_j, is added into the BSN, upon receiving the broadcast message pid_i, node S_j computes the individual key $K_j=f(sid_j,pid_i)$ by evaluating $f(sid_j,y)$ at pid_i. The individual key can be used for secure communication between node S_j and PWH. According to the data rate feature of a BSN described in Section III-C, we assume that time is divided into equal and fixed *collection rounds* and each biosensor collects a single data item per round. At round r, every biosensor, say S_j, generates the cipher text c_j^r with the individual key K_j as follows:

$$c_j^r=E(\{data_j^r,r\},K_j),h(data_j^r,K_j) \tag{1}$$

where $data_j^r$ is the collected data item by node S_j at round r. Subsequently, it delivers $\{sid_j,pid_i,c_j^r\}$ to PWH, where sid_j is the source ID while pid_i is the destination ID. One purpose of the round index r is to prevent replay attacks. Upon receiving such a message, PWH generates the individual key $K_j=f(pid_i,sid_j)=f(sid_j,pid_i)$ by evaluating $f(pid_i,y)$ at sid_j. Then, PWH uses K_j to perform $D(E(\{data_j^r,r\},K_j),K_j)=\{data_j^r,r\}$ to decrypt the cipher text. After that, PWH uses K_j to compute $h(data_j^r,K_j)$ and then compares it with the received $h(data_j^r,K_j)$. If the result is positive, PWH believes this message is from node S_j and has never been modified by the adversary. At the same time, if this is the first message from node S_j, PWH will record the mapping $<sid_j,K_j>$ for future use.

As described above, a hash value is calculated from and transmitted along with $data_j^r$. When the SHA-1 keyed hash functions is used, the corresponding hash value is 20 bytes long. In general, this kind of overhead is often not desirable in an already resource constrained BSNs. Data transmission is a costly operation in wireless networks; sending one bit over a wireless medium requires over 1000 times more energy than a single 32-bit computation [22]. In order to reduce the transmission overhead, we propose the use of subkeyed hash function. A subkeyed hash function only returns some bits of a hash value produced by a keyed hash function. For example, in the proposed system, the first λ bytes of each hash value is used as subkeyed hash value. Thus, we reduce the overhead by transmitting only a part of the actual hash value rather than the entire hash value. Here, we consider $\lambda=4$ as an example. Certainly, an adversary can more easily forge a 4-byte hash value than a 20-byte one. Thus, the size of the subkeyed hash value is directly related to not only the transmission overhead, but also the strength of the data integrity protection. A balance needs to be achieved between the desired security level and the transmission overhead.

Next, we focus on establishing pairwise keys that are shared only between biosensors and their immediate neighboring nodes (i.e., one-hop neighboring nodes). When a new biosensor, say S_j, is added into the BSN, S_j tries to discover its neighboring nodes. It broadcasts a HELLO message which contains a random number $nonce$ and sid_j, and waits for each neighboring node, say S_l, to respond with an acknowledgment (ACK) message including the identity (sid_l) of node S_l and $Aut=h(K_{jl},nonce)$, where $nonce$ is used to prevent replay attacks and node S_l generates the pairwise key $K_{jl}=f(sid_l,sid_j)$ by evaluating $f(sid_l,y)$ at sid_j. At the same time, node S_l can add the mapping (sid_j,K_{jl}) into its one-hop neighbor table. Upon receiving the ACK from S_l, node S_j computes the pairwise key $K_{jl}=f(sid_j,sid_l)=f(sid_l,sid_j)$ by evaluating $f(sid_j,y)$ at sid_l. Subsequently, node S_j uses the pairwise key to compute $Ver=h(K_{jl},nonce)$ and compares it with Aut. Only if this verification is successful, node S_j believes node S_l is its one-hop neighboring node and then adds the mapping (sid_l,K_{jl}) into its one-hop neighbor table.

It should be noted that no secret message is exchanged between node S_j and node S_l in the above procedure. Also, node S_j does not have to authenticate itself to node S_l by sending a special message, because any future messages encrypted with K_{jl} by node S_j will prove node S_j's identity. After the above steps, node S_j will have established a pairwise shared key with each of its neighbors and the pairwise key is used for securing data exchanged between them. Also, in the proposed system, for each biosensor node, the pairwise key (using subkeyed hash function) can be used to decide whether it should transfer the received PHI packet to PWH according to whether the source of the PHI packet is one of its one-hop neighboring nodes.

B. Individual Key Update Phase

With the transmitted PHI from each node to PWH, this paper proposes an extremely lightweight algorithm to update individual keys. The individual key for each biosensor is iteratively XORed with the hash value of the transmitted PHI to dynamically update the key. The idea of using the PHI for key agreement comes from the observation that the human body is dynamic and complex, and the PHI state of a patient is quite unique at a given time. Thus, the PHI used to update the individual key provides good degree of randomness so that an adversary would not be able to guess it and compromise the security of the system easily.

In the end of round r, the individual key of node S_j is computed as

$$K_j^r=K_j^{r-1}\oplus h(PHI_j^r)\quad r=1,2,\ldots \tag{2}$$

where PHI_j^r denotes the PHI from node S_j to PWH at round r while $h(.)$ is a one-way hash function which is publicly known.

Then, K_j^{r-1} is securely erased. Recall that $K_j^0 (= K_j)$ is the initial individual key using the polynomial share, we reinterpret (2) as

$$K_j^r = K_j^{r-1} \oplus h(PHI_j^r) = K_j^0 \oplus \bigoplus_{n=1}^{r} h(PHI_j^n). \quad (3)$$

Equation (3) is the same as Shannon's one time pad encryption. When key leakage happens (e.g., node S_j is compromised), the adversary knows K_j^0. In this case, $\bigoplus_{n=1}^{r} h(PHI_j^n)$ acts as the one time pad to prevent the adversary from deducing K_j^r.

Assume that at round r, a biosensor node, say S_j, hopes to deliver the data item $\{data_j^r\}$ to PWH. Node S_j generates the cipher text c_j^r as follows:

$$c_j^r = E(\{data_j^r, r\}, K_j^{r-1}), h(data_j^r, K_j^{r-1}). \quad (4)$$

Then, node S_j delivers $\{sid_j, pid_i, c_j^r\}$ to PWH. Note that PWH can update the individual key with node S_j in the same way. The above algorithm is implementation-friendly because bitwise-XOR and hash function are readily supported by most computing hardware. Moreover, the computational cost is very low due to the simplicity of the algorithms.

From (2) and (4), we can see that PHI encryption keys are updated whenever new PHI is received. Encryption keys are never reused, thus minimizing the risk of key discovery attacks. This leaves the adversary the only choice of brute-force attacks. Since keys are hash values, dictionary attacks do not apply. With a reasonable length of hash values, such as 160 bits coupled with a strong encryption algorithm, it will be very difficult for the adversary to crack keys. From (3), the individual key (K_j^r) is determined by all previous PHI transmissions and the initial individual key K_j. Also, if an adversary faked or stole the updated individual key at time $t = t_0$, to enable an impersonation attack at time $t = t_1$ $(t_1 > t_0)$, it must tap and obtain all PHI transmissions between t_0 and t_1. This task turns out to be more and more difficult as $t_1 - t_0$ becomes large.

V. Security Analysis of the Proposed Protocol

We evaluate the security of the proposed system by analyzing its fulfillment of the security requirements described in Section I.

Secure network admission: The security proof in [21] ensures that the proposed network admission and transmission subsystem is unconditionally secure and t-collusion resistant. That is, the coalition of no more than t compromised biosensor nodes knows nothing about the pairwise key between any two noncompromised nodes. At the same time, the coalition of no more than t compromised biosensor nodes knows nothing about the initial individual key between any noncompromised biosensor node and PWH. Because the scale of each BSN is very small (i.e., from several to tens of biosensor nodes), in our system, it is suggested that t should be equal to the total number of the biosensor nodes in a BSN. Obviously, in this case, unless all biosensor are compromised, no pairwise or individual key could be revealed by adversaries. As described in Section IV-A, it is critical for our system to restrict the network admission only to PWH and biosensors, which has the knowledge of a polynomial share of the same bivariate t-degree polynomial.

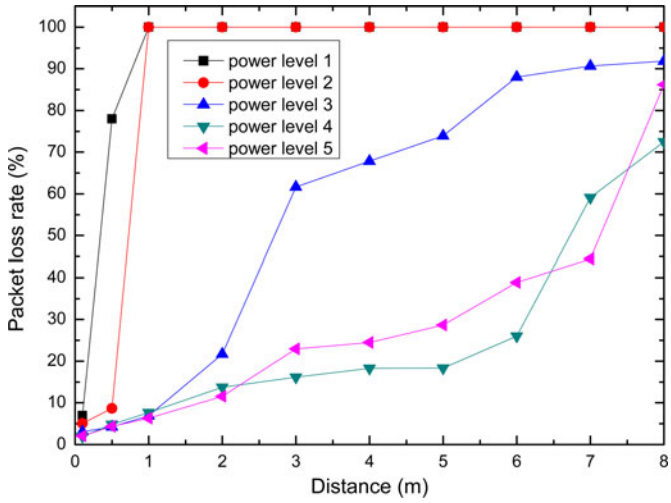


Fig. 2. Packet loss rate with the distance of different lengths.

Secure transmission: Same as the proof of the secure network admission, in the proposed protocol, symmetric encryption and subkeyed hash function are used to ensure confidential, authenticated, and integrity protected transmission between each biosensor and PWH.

VI. Discussion

So far, we have elaborated the operations of the proposed system. By the system, we can achieve secure network admission and transmission. However, some important issues need further discussion.

A. Transmission Range of a BSN

To investigate the transmission range of a BSN, an indoor experiment has been conducted, where a TelosB mote broadcasts 10 000 packets without waiting for ACKs, while the base station (performed by a TelosB mote) acts as the message collector. The data sending node is deployed at a distance (0.1 m, 0.5 m, and from 1 to 8 m in increments of 1 m) from the base station. The transmitting power level of the sending node l was set from 1 to 5, and the delivery rate of packets is 10 packets/s. On the other hand, the payload size of each packet is set to 20 bytes. Fig. 2 shows the packet loss rates at different distances. For example, when the power level was 1 or 2, the packet loss rate was 100% if the distance between the motes was more than 1 m. When the power level was 3, the packet loss rate was 91.855% if the distance between the motes was more than 8 m. Also, when the power level was 2 (respectively, 3), the packet loss rate was 8.65% (respectively, 6.91%) if the distance was 0.5 m (respectively, 1 m).

Consider a scenario that at time t_0, an adversary successfully steals the updated individual key of node S_j and can eavesdrop on the PHI transmission with probability μ. Also we assume node S_j transmits Num PHI packets to PWH during the interval $t_1 - t_0$. Thus, the probability for the adversary to successfully

generate the individual key at time t_1 is μ^{Num}. For example, when $\mu = 80\%$ and $Num = 10$, the probability is 10.74%.

In practice, because a data packet (about 100 bytes) usually contains more bits of entropy than an individual key (about 16 bytes). It takes only one packet loss for the adversary to be completely confused about the next updated individual key.

It is worth noting that session key exchange does not improve security against key leakage, but the above procedure does. Session keys are derived from a master key through a dedicated key exchange process. Once the master key is known to an adversary, session keys can be deduced through knowledge of the session key exchange processes. In practice, session key exchanges only account for a small portion of total transmissions. The adversary can gather the necessary information to deduce session keys by eavesdropping these portions of the transmissions. Moreover, an adversary can always initiate a new session using a leaked master key.

In practice, PWH may lose some packets from a biosensor node, say S_j. To address this problem, an efficient ACK method can be implemented in our system. At the end of each round, say r, node S_j waits for an ACK message from PWH. According to the system configuration, node S_j transmits β PHI packets to PWH during each round. The ACK message can be β bits, with a bit of 1 indicating a received packet and a bit of 0 indicating a missed message. When node S_j receives the ACK, as shown in (2), it uses only those acknowledged packets to update the individual key. Note that this ACK message is also encrypted with the individual key K_j^{r-1}.

B. Ensuring Secure Transmission in a Usable (Plug-n-Play, Transparent) Manner

As described in Section IV, only simple predeployment configuration is required for the proposed system. That is, only a polynomial share is installed into each biosensor node and PWH. In some cases, *usable security* (one that is plug-n-play and largely transparent) should be provided. With this feature, the BSN administrator is able to add, remove, and adjust the biosensors on his/her BSN, as and when required, without reconfiguring any part of the network. To achieve this goal, a simple modification is required for the proposed system as follows. In the system initialization phase, the BSN administrator does not need to generate a bivariate t-degree polynomial and no polynomial share is installed into any biosensor node. Further, no initial individual key between each biosensor node and PWH is generated. That is, for (2), in the end of the first round, the individual key K_j^1 of node S_j is computed as $h(PHI_j^1)$, not $K_j^0 \oplus h(PHI_j^1)$. More exactly, (3) should be modified into $K_j^r = \bigoplus_{n=1}^{r} h(PHI_j^n)$. In this case, our system just utilizes the transmitted PHI for enabling each biosensor and the PWH to establish and update the individual key.

VII. Implementations and Performance Evaluation

We evaluate the proposed protocol by implementing all its components in an experimental testbed.

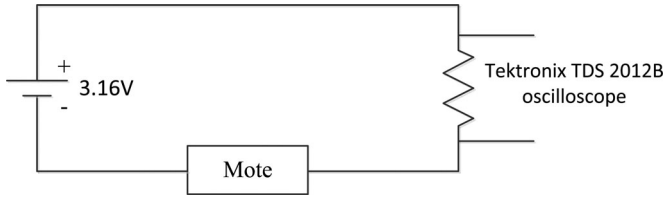


Fig. 3. Experimental setup for investigating the energy consumption.

A. Implementation and Experimental Setup

In order to investigate the feasibility of the proposed mechanisms on biosensor nodes, same as the existing studies [6], [13], [16], [19] on securing BSNs, we choose two common resource-limited sensor nodes: TelosB and MicaZ motes. Our motes run TinyOS [23] version 2.x.

To measure the energy consumption of executing various operations considered in this paper, similar to [24], we have built a circuit as shown in Fig. 3 for our experiments on resource-limited sensor nodes. The circuit connects batteries, a sensor node, and a 20.36-Ω resistor in series. A Tektronix TDS 2012B oscilloscope is used to accurately measure the voltage across the resistor. From this measurement, the current I through the circuit and the voltage V_m across the mote can be obtained. The power consumed by the sensor node is calculated using the equation $P = V_m \times I$. Here, the voltage across the mote $V_m = V_b - V_r$, where V_b is the voltage of the batteries while V_r is the voltage across the resistor. In our experiments, V_b =3.16 V. To measure the energy consumption of an operation, we also measure the execution time of the operation and then multiply it with the power consumption. Throughout this paper, unless otherwise stated, all experiments on sensor nodes were repeated 1000 times for each measurement in order to obtain accurate average results.

B. Evaluation Results and Performance Comparison

As biosensor nodes are usually resource constrained, they may not be able to execute expensive cryptographic operations efficiently and thus become the bottleneck of a security protocol. We use the following four metrics to evaluate the proposed protocol, namely, memory overhead, communication overhead, execution time, and energy overhead. The memory overhead measures the exact amount of data space required in the real implementation. Similarly, the execution time measures the time duration of each mechanism.

Table III gives the execution times of evaluating t-degree polynomials in laptop PCs when t varies. This requires $2t$ modular multiplications and t modular additions in a finite field F_p. For example, the execution time on an 800-MHz laptop PC is 0.989 ms when $t = 300$. Thus, evaluation of the polynomial is very fast. Considering that the clock frequency of a typical smartphone is more than 800 MHz, our protocol is efficient for most PWHs. Tables IV and V give the execution times of evaluating a t-degree polynomial on MicaZ and TelosB motes, respectively. For example, the execution times on a MicaZ mote and a TelosB mote are 3.6715 and 8.225 ms when

TABLE III
COMPUTATION TIMES FOR EVALUATING A t-DEGREE POLYNOMIAL ON PWH

	800-MHz Processor				1.6-GHz Processor				2.4-GHz Processor			
	t=50	t=100	t=300	t=500	t=50	t=100	t=300	t=500	t=50	t=100	t=300	t=500
Time (ms)	0.181	0.333	0.989	1.645	0.085	0.148	0.454	0.799	0.054	0.102	0.331	0.517

TABLE IV
COMPUTATION TIMES FOR EVALUATING A t-DEGREE POLYNOMIAL ON MICAZ MOTES

t	10	20	30	40	50	60	70
Time (ms)	0.8301	1.5415	2.2479	2.9599	3.6715	4.3854	5.0989

TABLE V
COMPUTATION TIMES FOR EVALUATING A t-DEGREE POLYNOMIAL ON TELOSB MOTES

t	10	20	30	40	50	60	70	80	90	100	110	120
Time (ms)	1.873	3.408	5.075	6.724	8.225	9.893	11.601	13.119	14.758	16.394	18.114	19.703

TABLE VI
ENERGY CONSUMPTION FOR EVALUATING A t-DEGREE POLYNOMIAL ON MICAZ MOTES

t	10	20	30	40	50	60	70
Energy (mJ)	0.0380	0.0706	0.1029	0.1355	0.1681	0.2008	0.2334
$(10^{-5})\%$	0.0563	0.1046	0.1524	0.2007	0.2490	0.2975	0.3458

TABLE VII
ENERGY CONSUMPTION FOR EVALUATING A t-DEGREE POLYNOMIAL ON TELOSB MOTES

t	10	20	30	40	50	60	70	80	90	100	110	120
Energy (mJ)	0.1067	0.1942	0.2892	0.3831	0.4686	0.5637	0.6609	0.7475	0.8409	0.9341	1.0321	1.1226
$(10^{-4})\%$	0.0158	0.0288	0.0428	0.0568	0.0694	0.0835	0.0979	0.1107	0.1246	0.1384	0.1529	0.1663

$t = 50$, respectively. For each of these results, we perform the same experiment for 10 000 times and take an average over them. Obviously, this operation is quite affordable on resource-limited motes. For memory overhead, irrespective of the value of t, the implementations of evaluating a t-degree polynomial on a MicaZ mote and a TelosB mote occupy 18 596 and 18 860 bytes of ROM, respectively. These values correspond to only 37.834% and 14.389% of the ROM capacities of MicaZ and TelosB motes, respectively. On the other hand, the RAM sizes of the implementations of evaluating a t-degree polynomial depend on the value of t. For example, the implementations of evaluating a polynomial of degree 50 on a MicaZ mote and a TelosB mote occupy 3106 and 3084 bytes of RAM, respectively. From these results, considering that the total number of biosensor nodes in a generic BSN is less than 50, our protocol is efficient for the biosensors of most BSNs.

Next, the energy consumption of evaluating a t-degree polynomial is investigated. When a MicaZ mote is used in the circuit, $V_r = 456$ mV, $I = 22.3969$ mA, $V_m = 2.544$ V, and $P = 56.9777$ mW. When a TelosB mote is used, $V_r = 352$ mV, $I = 17.2888$ mA, $V_m = 2.648$ V, and $P = 45.7807$ mW. As illustrated in Tables VI and VII, by multiplying the power with the execution time, we determine the total energy consumption of evaluating a t-degree polynomial on MicaZ and TelosB motes, respectively. For example, the energy consumption of evaluating a polynomial of degree 50 on MicaZ and TelosB motes are 0.2092 and 0.3765 mJ, respectively.

Batteries are the standard power source for the nodes of a BSN. MicaZ and TelosB motes are powered by 2 AA alkaline batteries, the rated capacity of each of which is about 2500 mAh. This capacity is defined by the manufacturers as the amount of energy that can be delivered until the voltage of a single AA battery reaches 0.8 V. However, as reported in [25], the supply voltage has to be at least 2.7 V for the proper operation of both MicaZ and TelosB motes, and energy capacity available for a node powered by two AA alkaline batteries is effectively 6750 J. Tables VI and VII provide the percentages of total available energy capacity of the batteries consumed by each evaluation of a t-degree polynomial on MicaZ and TelosB motes, respectively. For example, the percentage of the energy consumption of evaluating a polynomial of degree 50 operation on a TelosB mote is $0.249\times(10^{-4})\%$. That is, the batteries can support 4 016 064 times of such an operation on a TelosB mote.

To investigate the efficiency of public key cryptography, we have implemented the 160-bit ECC algorithm of TinyECC library [26] on MicaZ and TelosB motes, it is measured that the signature generation times are 1.9096 and 3.0526 s, respectively. And the signature verification times are 2.4195 and 3.8942 s, which are 658 and 473 times longer than the times of evaluating a polynomial of degree 50 on MicaZ and TelosB motes, respectively. When signature generation is executed on a MicaZ mote, $V_r = 0.149$ V and $P = 22.0353$ mW. For a TelosB mote, $V_r = 0.042$ V and $P = 6.4320$ mW. Thus, the energy consumption of signature generation are 42.0786 and 19.6344 mJ on MicaZ and TelosB motes, respectively. Also, the energy consumption of signature verification are 85.8099 and 53.3144 mJ on MicaZ and TelosB motes, which are 409 and 141 times more than evaluating a polynomial of degree 50, respectively.

TABLE VIII
EXECUTION TIMES OF RC5 FOR MICAZ AND TELOSB MOTES WITH THE PLAINTEXT OF DIFFERENT LENGTHS

	MicaZ						TelosB					
Plaintext length (bytes)	16	32	48	64	80	96	16	32	48	64	80	96
Encryption time (ms)	1.849	3.708	5.567	7.413	9.265	11.116	2.576	5.141	7.708	10.29	12.84	15.312
Decryption time (ms)	1.864	3.703	5.554	7.407	9.255	11.099	2.604	5.18	7.747	10.325	12.927	15.406

TABLE IX
EXECUTION TIMES OF SKIPJACK FOR MICAZ AND TELOSB MOTES WITH THE PLAINTEXT OF DIFFERENT LENGTHS

	MicaZ						TelosB					
Plaintext length (bytes)	16	32	48	64	80	96	16	32	48	64	80	96
Encryption time (ms)	0.671	1.335	2.001	2.67	3.328	3.991	0.957	1.888	2.82	3.758	4.693	5.520
Decryption time (ms)	0.665	1.319	1.974	2.614	3.283	3.935	0.963	1.908	2.840	3.778	4.723	5.683

TABLE X
EXECUTION TIMES OF SOFTWARE AES FOR MICAZ AND TELOSB MOTES WITH THE PLAINTEXT OF DIFFERENT LENGTHS

	MicaZ						TelosB					
Plaintext length (bytes)	16	32	48	64	80	96	16	32	48	64	80	96
Encryption time (ms)	1.147	2.283	3.355	4.555	5.691	6.829	1.838	3.645	5.46	7.254	9.061	10.864
Decryption time (ms)	1.337	2.663	4.001	5.326	6.66	7.99	2.193	4.359	6.517	8.675	10.843	13.006

TABLE XI
EXECUTION TIMES OF STAND-ALONE HARDWARE AES FOR MICAZ AND TELOSB MOTES WITH THE PLAINTEXT OF DIFFERENT LENGTHS

	MicaZ						TelosB					
Plaintext length (bytes)	16	32	48	64	80	96	16	32	48	64	80	96
Encryption time (ms)	0.043	0.074	0.108	0.143	0.175	0.206	0.116	0.195	0.278	0.365	0.434	0.537

The implementation of signature generation and verification on a MicaZ mote occupies 21 972 bytes of ROM and 2310 bytes of RAM, respectively. Also, the implementation occupies 26 196 bytes of ROM and 2178 bytes of RAM on a TelosB mote, respectively. Thus, the implementation of ECC occupies more ROM and comparable RAM than that of evaluating a polynomial of degree 50. According to the above analysis, the proposed protocol is much more efficient than the public key based mechanisms [6], [16] in terms of computation complexity and memory overhead.

In our approach, each node needs to store a t-degree polynomial share of $f(x, y)$, which occupies $(t + 1) \log p$ memory space. In our implementation, p is set to 64 bits long for typical cryptosystems such as RC5. Thus, the security strength and memory overhead of the proposed scheme vary with t. The larger the t is, the more secure the proposed scheme can be. However, a large t incurs large memory overhead on each node. Therefore, in the proposed scheme, it is flexible for the administrator of each BSN to choose the value of t to balance the security strength and memory overhead. Moreover, compared to PKI, the polynomial-based authentication scheme does not require the generation, transmission, and verification of digital certificates, and there is no communication overhead during the common key establishment process between any two nodes. Also, for the random key scheme, each two nodes need to exchange key information in order to establish the pairwise key and, inevitably, higher communication overhead is incurred. According to the above analysis, besides the flexibility, the proposed protocol is more efficient than public-key-based mechanisms [6], [16] and the random key scheme-based software design [19] with respect to communication overhead.

TABLE XII
CODE SIZES OF IMPLEMENTATION OF RC5, SKIPJACK, SOFTWARE AES, AND STAND-ALONE HARDWARE AES

		RC5	Skipjack	Software AES	stand-alone hardware AES
MicaZ	ROM (bytes)	4,092	5,600	7,144	11,366
	RAM (bytes)	131	387	2,262	283
TelosB	ROM (bytes)	3,602	4,760	5,912	11,616
	RAM (bytes)	131	131	2,262	301

RC5, Skipjack, and AES are symmetric-key encryption/decryption algorithms used on sensor nodes. Here, we have ported the RC5 and Skipjack implementations for TinyOS 1.x in the TinySEC library [27] to TinyOS 2.x. For AES, its encryption module is implemented in hardware in one of the most broadly used radios, CC2420. We have also implemented software AES encryption/decryption algorithm [28] on MicaZ and TelosB motes. RC5 is used with 12 rounds (with a 64-bit key and 64-bit block size), Skipjack is used with a 80-bit key and 64-bit block size, software AES and CC2420 radios stand-alone hardware AES encyption modules are used with ten rounds (with a 128-bit key and 128-bit block size). All algorithms are used with ECB mode. Tables VIII–XI show the execution times of RC5 encryption/decryption, Skipjack, Software AES encryption/decryption, and stand-alone hardware AES for MicaZ and TelosB motes with the plaintext of different lengths, respectively. Also, Table XII shows the code sizes of the implementation of these four symmetric-key algorithms. It is clear that hardware implemented AES outperforms all other algorithms for a given security level. Moreover, since hardware AES acceleration is supported in many hardware platforms for BSN, AES is clearly the preferred solution. Thus, compared to the security

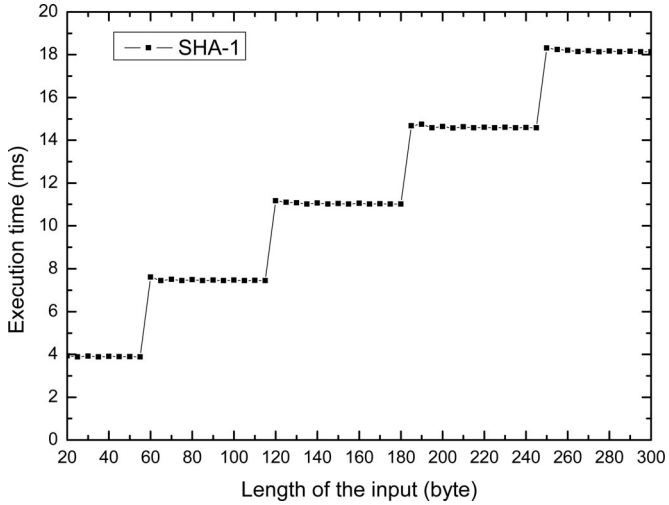


Fig. 4. Execution times of SHA-1 function on TelosB motes.

mechanism based on RC5 [6], the proposed protocol is more efficient due to the use of hardware implemented AES.

Fig. 4 shows the execution times of SHA-1 hash function on a TelosB mote. The length of the input is varied from 20 to 300 bytes in increments of 5 bytes. We perform the same experiment 10 000 times and take an average over them. For example, the execution times on a TelosB mote for inputs of 55, 60, 120, 185, and 250 bytes are 3.8901, 7.612, 11.1717, 14.6703, and 18.3106 ms, respectively. From Fig. 4, it can be seen that the execution time remains very stable when the byte length of the input falls in the interval $[0, 55]$, $[56, 119]$, $[120, 183]$, $[184, 247]$, or $[248, 300]$. According to (2), as the input of a hash function, the longer the length of the PHI in each round (i.e., the better the degree of randomness), the stronger the security strength of the updated individual key. Thus, it is suggested that in our system, the length of the PHI in each round should be chosen according to the above intervals to achieve a balance between the desired security level and computing complexity. For example, when the length of the PHI is a bit longer than 55 bytes, according to the system configuration, only 55 bytes data are picked from the PHI as the input of a hash function.

Next, we have implemented the proposed protocol (including key establishment and update) as a whole. Here, SHA-1 hash function is used. Each node uses CC2420 stand-alone hardware AES encryption module to encrypt the sensed data while the PWH uses software AES module to decrypt the received data. Because the key size of the used AES algorithms is 16 bytes while the output of SHA-1 function is 20 bytes, only the first 16-byte data are used in the individual key update phase. For PWH and each node, the buffer size for the plaintext [i.e., $\{data_j^r, r\}$ in (1)] of AES algorithm can be fixed according to the application scenario. In our implementation, it is set to 32 bytes. Table XIII shows the code sizes of the whole protocol. For example, the implementation of the node side programs on a MicaZ mote occupies 22 566 bytes of ROM and 2404 bytes of RAM, respectively. The resulting size of this implementation corresponds to only 15.64% and 17.68% of the RAM and ROM capacities of MicaZ, respectively. Also, the implementation of PWH side programs on a MicaZ mote occupies 22 566 bytes of ROM and 2404 bytes of RAM, respectively.

TABLE XIII
CODE SIZES OF THE WHOLE PROTOCOL

	MicaZ		TelosB	
	ROM (bytes)	RAM (bytes)	ROM (bytes)	RAM (bytes)
PWH	22,566	2,404	20,296	2,427
Node	20,506	724	18,628	747

VIII. CONCLUSION

In this paper, we have explored the features of a BSN and then presented a novel secure and lightweight network admission and transmission protocol. Further, to reduce the computation and communication overhead, some additional mechanisms such as subkeyed hash function and the hardware-implemented AES algorithm are incorporated into the design of the proposed system. The security analysis and experimental results have shown that our approach is feasible for real applications. Our experiments have also shown that the system overhead of the proposed protocol is affordable on resource-limited motes, which is much more efficient than the well-known approaches.

REFERENCES

[1] K. Lorincz, D. J. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, S. Moulton, and M. Welsh, "Sensor networks for emergency response: Challenges and opportunities," *IEEE Pervasive Comput.*, vol. 3, no. 4, pp. 16–23, Oct./Dec. 2004.

[2] N. Challa, H. Çam, and M. Sikri, "Secure, efficient data transmission over body sensor and wireless networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2008, Article ID 291365, 2008.

[3] The health Information Trust Alliance (HITRUST). [Online]. Available:http://www.hitrustalliance.org

[4] The US Congress. (1996). Health Insurance Portability and Accountability Act. Washington, DC. [Online]. Available:http://www.hhs.gov/ocr/privacy/

[5] The European Parliament and the Council of the European Union, Directive 95/46/EC [Online]. Available: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:HTML

[6] K. Malasri and L. Wang, "Design and implementation of a secure wireless mote-based medical sensor network," *Sensors*, vol. 9, no. 8, pp. 6273–6297, 2009.

[7] C. Cordeiro and M. Patel, "Body area networking standardization: Present and future directions," in *Proc. BodyNets*, 2007.

[8] H. Wang and Q. Li, "Efficient implementation of public key cryptosystems on mote sensors," in *Proc. Int. Conf. Inf. Commun. Security*, 2006, pp. 519–528.

[9] W. Du, J. Deng, Y. Han, P. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks," *ACM Trans. Inf. Syst. Security*, vol. 8, no. 2, pp. 228–258, May 2005.

[10] D. He, C. Chen, S. Chan, and J. Bu, "SDRP: A secure and distributed reprogramming protocol for wireless sensor networks," *IEEE Trans. Ind. Electron.*, vol. 59, no. 11, pp. 4155–4163, Nov. 2012.

[11] W. He, Y. Huang, R. Sathyam, K. Nahrstedt, and W. Lee, "SMOCK: A scalable method of cryptographic key management for mission-critical wireless ad-hoc networks," *IEEE Trans. Inf. Forensics Security*, vol. 4, no. 1, pp. 140–150, Mar. 2009.

[12] S. Zhao, A. Aggarwal, R. Frost, and X. Bai, "A survey of applications of identity-based cryptography in mobile ad-hoc networks," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 2, pp. 380–400, May 2012.

[13] K. Malasri and L. Wang, "Addressing security in medical sensor networks," in *Proc. HealthNet.*, 2007, pp. 7–12.

[14] S. Bao, C. Poon, Y. Zhang, and L. Shen, "Using the timing information of heartbeats as an entity identifier to secure body sensor network," *IEEE Trans. Inf. Technol. Biomed.*, vol. 12, no. 6, pp. 772–779, Nov. 2008.

[15] A. Ali, S. Irum, F. Kausar, and F. Khan, "A cluster-based key agreement scheme using keyed hashing for body area networks," *Multimed. Tools Appl.*, 2011, DOI: 10.1007/s11042-011-0791-4.
[16] C. C. Tan, H. Wang, S. Zhong, and Q. Li, "IBE-Lite: A lightweight identity-based cryptography for body sensor networks," *IEEE Trans. Inf. Technol. Biomed.*, vol. 13, no. 6, pp. 926–932, Nov. 2009.
[17] D. He, C. Chen, S. Chan, J. Bu, and A. Vasilakos, "A distributed trust evaluation model and its application scenarios for medical sensor networks," *IEEE Trans. Inf. Technol. Biomed.*, vol. 16, no. 6, pp. 1164–1175, Nov. 2012.
[18] D. He, C. Chen, S. Chan, J. Bu, and A. Vasilakos, "ReTrust: Attack-resistant and lightweight trust management for medical sensor networks," *IEEE Trans. Inf. Technol. Biomed.*, vol. 16, no. 4, pp. 623–632, Jul. 2012.
[19] S. Mollera, T. Newe, and S. Lochmann, "Prototype of a secure wireless patient monitoring system for the medical community," *Sens. Actuators A: Phys.*, vol. 173, no. 1, pp. 55–65, Jan. 2012.
[20] R. Pickholtz, D. Schilling, and L. Milstein, "Theory of spread spectrum communications-a tutorial," *IEEE Trans. Commun.*, vol. 30, no. 5, pp. 855–884, May 1982.
[21] C. Blundo, A. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Proc. Adv. Cryptol.*, 1993, vol. LNCS-740, pp. 471–486.
[22] K. C. Barr and K. Asanovi, "Energy aware lossless data compression," *ACM Trans. Comput. Syst.*, vol. 24, no. 3, pp. 250–291, Aug. 2006.
[23] Tiny OS. [Online]. Available:http://www.tinyos.net
[24] J. Lee, K. Kapitanova, and S. Son, "The price of security in wireless sensor networks," *Comput. Netw.*, vol. 54, no. 17, pp. 2967–2978, Dec. 2010.
[25] K. Piotrowski, P. Langendoerfer, and S. Peter, "How public key cryptography influences wireless sensor node lifetime," in *Proc. SASN*, 2006, pp. 69–176.
[26] A. Liu and P. Ning, "TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks," in *Proc. Int. Conf. Inf. Process. Sensor Netw.*, 2008, pp. 245–256.
[27] C. Karlof, N. Sastry, and D. Wagner, "TinySec: A link layer security architecture for wireless sensor networks," in *Proc. ACM SenSys*, 2004, pp. 162–175.
[28] [Online]. Available: http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/crypto/index.html

Daojing He received the B.Eng. and M.Eng. degrees in computer science from the Harbin Institute of Technology, Harbin, China, in 2007 and 2009, respectively. He is currently working toward the Ph.D. degree in the Department of Computer Science, Zhejiang University, Zhejiang, China.

His research interests include network and systems security with focuses on wireless security.

Mr. He is an Associate Editor or on the Editorial Board of some international journals such as *Wiley's Wireless Communications and Mobile Computing Journal, Wiley's Security and Communication Networks Journal*, and *KSII Transactions on Internet and Information Systems*. He is a Technical Program Committee Member of many international conferences.

Chun Chen received the Bachelor's degree in mathematics from Xiamen University, Xiamen, China, in 1981, and the Master's and Ph.D. degrees in computer science from Zhejiang University, Zhejiang, China, in 1984 and 1990, respectively.

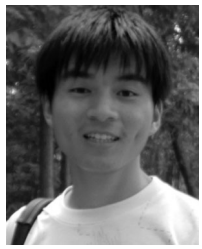
He is currently a Professor in the College of Computer Science, and the Director of the Institute of Computer Software at Zhejiang University. His research interests include image processing, computer vision, and embedded system.

Sammy Chan received the B.E. and M.Eng.Sc. degrees in electrical engineering from the University of Melbourne, Parkville, Vic., Australia, in 1988 and 1990, respectively, and the Ph.D. degree in communication engineering from the Royal Melbourne Institute of Technology, Melbourne, Vic., Australia, in 1995.

From 1989 to 1994, he was with Telecom Australia Research Laboratories, first as a Research Engineer, and between 1992 and 1994 as a Senior Research Engineer and Project Leader. Since December 1994, he has been with the Department of Electronic Engineering, City University of Hong Kong, where he is currently an Associate Professor.

Jiajun Bu received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Zhejiang, China, in 1995 and 2000, respectively.

He is currently a Professor at the College of Computer Science and Deputy Director of the Institute of Computer Software, Zhejiang University. His research interests include embedded systems, mobile multimedia, and data mining.



Pingxin Zhang received the B.S. degree in computer science from Shanghai Jiaotong University, Shanghai, China, in 2011. He is currently working toward the Master's degree at the College of Computer Science, Zhejiang University, Zhejiang, China.

His research interests include network security and wireless sensor networks.