Securing Body Sensor Networks: Sensor Association and Key Management

Sye Loong Keoh
Dept. of Info. & System Security, Philips Research
High Tech Campus 34, 5656 AE, Eindhoven
Email: sye.loong.keoh@philips.com

Emil Lupu and Morris Sloman
Dept. of Computing, Imperial College London
London SW7 2RH
Email: {e.c.lupu, m.sloman}@imperial.ac.uk

***Abstract*—Body Sensor Networks can be used to continuously monitor patients' health. However, secure association of sensors with the patient and key management for providing integrity and confidentiality to the sensor readings is essential. We propose a secure discovery protocol based on the synchronised LED blinking pattern, to enable healthcare workers to authorise the sensor-to-patient association. We also propose a novel key distribution and management scheme that uses keychains to establish group keys for body sensor networks and caters for group key update and re-keying to adapt to membership changes. These protocols have been implemented to demonstrate their feasibility and initial performance evaluation is presented.**

I. Introduction

Wireless Body Sensor Networks (BSNs) with portable devices such as smart phones enable continuous efficient monitoring and management of post-operative and chronically-ill patients to enable early release from hospital. Healthcare personnel can be automatically alerted if the patient's condition deteriorates. BSNs must ensure confidentiality, integrity and availability of the physiological data, as wireless sensor networks are susceptible to passive eavesdropping, packet injection, and vulnerable to many other security attacks. Furthermore, the design of the security protocols must carefully balance satisfying the security requirements with limiting the power and computational requirements. To securely establish BSNs we must first ensure that only designated sensors are associated with the patient and only by an authorised party. For example, a nurse is permitted to attach a specific approved ECG sensor to monitor the heart-rate of a post-operative patient while arbitrary nearby ECG sensors on other patients must not be associated.

The second security goal is to preserve the confidentiality and integrity of the medical data, which is susceptible to eavesdropping when transmitted wirelessly. Sensor readings must therefore be encrypted and given the sensors' limited capabilities, the use of a shared group key is desirable. A lightweight key management scheme is needed to use less computationally intensive cryptographic operations and reduce power consumption by minimising messages exchanges.

Thirdly, managing group membership is required to facilitate addition and removal of sensors from the BSN, and renewing the group key when necessary. Existing sensors may fail permanently or temporarily become disconnected from the BSN whereas new sensors may be added according to medical needs. There is a need to distinguish between temporary and permanent disconnection, to establish persistent pairings that survive transient disconnection and to renew the group key when a sensor is permanently detached. We propose a novel secure sensor discovery protocol that integrates the BLIG [1] sensor association scheme, with an efficient key distribution and key management scheme using a keychain to establish a shared secret group key.

The paper is organised as follows: Section II discusses related work and its limitations. Section III discusses the threat model. Section IV, describes sensor discovery and association, while Section V describes the key distribution and management. We present the prototype implementation and results in Section VI and discuss trade-offs in Section VII. Conclusions and future work are presented in Section VIII.

II. Related Work

BLIG [1] uses synchronised blinking of LEDs enabling a healthcare worker to visually verify the correct grouping of sensors on a patient. It uses short range communication, (i.e., < 0.5m), to discover new sensors, establish the patient's identity and map between the patient's true id and the sensor group. We have enhanced BLIG for more robust sensor discovery and association as well as sensor authentication and key management for BSNs.

Pre-shared symmetric key protocols are used in large scale sensor networks for environmental monitoring [2], [3]. A key-share is loaded in each node and used to derive a common secret key. In a hospital setting, this is not sufficient as groups overlap in their wireless range and only the correct sensors must be associated with the patient. Additionally, these protocols require intensive computation to replace the group key when membership changes.

Balfanz [4] uses a secure-limited channel, e.g., infrared to exchange public-keys between parties in a pre-authentication phase before authentication. However, this requires dedicated hardware and/or hosting the pre-authentication phase in a confined area, which would be difficult in a hospital ward. Human confirmation of correct association is also difficult when based on public-keys and without visual cues.

This research was supported by UK EPSRC BiosensorNet Grant EP/C547586/1

The resurrecting duckling protocol [5], [6] establishes a master-slave relationship between devices where the first device in contact with a sensor becomes its master and can upload policies to the sensor that permit interactions with other devices. Sensors from previous patients have to be explicitly dis-associated by the master before reuse by other patients which may not always be practical in hospitals.

Jiang [7] uses self-certified keys (SCK) and Elliptic Curve Cryptography (ECC) to establish pair-wise keys for authentication. Each sensor establishes a secret with the user based on the secret information pre-loaded by a key distribution centre (KDC). Authentication is achieved if the user demonstrates knowledge of the shared secret-key with at least t sensors. To achieve sensor to patient association each patient's BSN would require different ECC curve parameters (as each BSN is a domain)which would be impractical for hundreds of BSNs in a hospital. SNAP [8] also uses ECC to establish pair-wise keys between sensors and the base station. It requires each sensor to be equipped with a biometric device to authenticate the patient and uses the shared secret to communicate with the base station. However, it does not establish a group keys. Other studies have shown that ECC-based public-key cryptography [9], [10], [11], is viable for resource constrained wireless sensor networks to provide better key distribution, management and authentication.

III. Threat Model

It is not easy to confine short range wireless interactions in hospital wards. Data confidentiality may be compromised through eavesdropping. Data integrity may be compromised by injecting data into the wireless channel or simply interfering with the wireless transmission. Sensors may be associated with the wrong patient either mistakenly or maliciously (e.g., through impersonation). Medical data is confidential and would interest many attackers including employers, insurance companies, personal enemies and unscrupulous media for newsworthy persons.

Passive Eavesdropping. Wireless channels are susceptible to passive eavesdropping and message interception. Whilst traffic can be encrypted, sensor data is vulnerable to cryptanalysis as measurements fall within known ranges, and chosen ciphertext attacks become possible. Therefore, the encryption key must be regularly renewed and used for short time periods.

Active Attacks. Active attacks include injecting messages into the wireless channel, replaying or altering messages. Spurious messages injected into the network, e.g., sending false sensor readings will disrupt the accuracy of the measurements. Message replays or modification of messages, to cause a medication overdose, pose a serious risk to the patient. Thus, authentication, authorisation and message freshness are necessary to ensure the confidentiality and integrity of the transmitted data.

Impersonation and Masquerading. Impersonation would typically occur when an attacker in the guise of a sensor joins or claims to be part of a patient's BSN to eavesdrop on data and/or report false data. Authentication is essential to prove the sensor's membership in the BSN

Denial of Service Attack. Our work does not address Physical Denial-of-Service (DoS) attacks by frequency jamming [12]. We are solely concerned with detecting malicious sensors that repeatedly attempt to join a BSN, thus aiming to deplete resources and prevent genuine sensors from joining.

IV. Secure Sensor Association

Post-operative patients typically need continuous monitoring whilst in hospital or at home e.g. ECG sensors monitor heart condition, SpO_2 sensors monitor oxygen saturation, and other sensors observe blood pressure and body temperature. A device such PDA or mobile phone is often used as a *controller* to co-ordinate communication and manage the sensors in the BSN. When attaching a new sensor, the healthcare worker switches on the sensor, enabling discovery by the controller. However, nearby sensors will also be discovered and solely based on sensor type and credentials, the BSN controller cannot decide if the sensor should be associated with its patient. An explicit action is thus required to identify the correct association. Instead of using short-range confined communication or physical contact to distinguish the sensor, we use the BLIG approach [1] where sensor and controller synchronise their LED blinking in an agreed pattern. This provides an authorised healthcare worker with the visual representation of the association, that she can then explicitly authorise. The discovery protocol also establishes a pairwise key between each sensor admitted to the BSN and the patient controller device, used to distribute the group key. The next section presents our assumptions and introduces the cryptographic notation whilst the protocol is described in Section IV-B.

A. Assumptions and Notation

We consider the hospital as the root Certification Authority (CA) issuing attribute certificates to its staff, patients, devices and sensors certifying their roles and attributes. Appropriate certificates and private keys are pre-loaded into devices and sensors before release for use to enable attribute-based authentication.

Sensors attached to the patient are discovered sequentially, by the patient's controller and a complete iteration of the protocol is necessary for each sensor. Otherwise, an attacker sensor could "lie" and copy the neighbour's blinking at the same time as the neighbour is being associated.

Medical sensors are trusted to generate good pseudo random numbers usingaccelerometers or input from other noisy sensors to determine a seed for the generator.

Authenticated healthcare workers and patients i.e. who present valid attribute certificates issued by the hospital are assumed well-behaved. The protocols do not attempt to deal with malicious insiders. More specifically, healthcare workers are trusted to correctly authorise sensor-to-patient associations, and patients are trusted to not interfere with other patients BSN, e.g., launching man-in-the middle attacks. Further, sensors that have become detached from the BSN, are trusted to

why is this needed

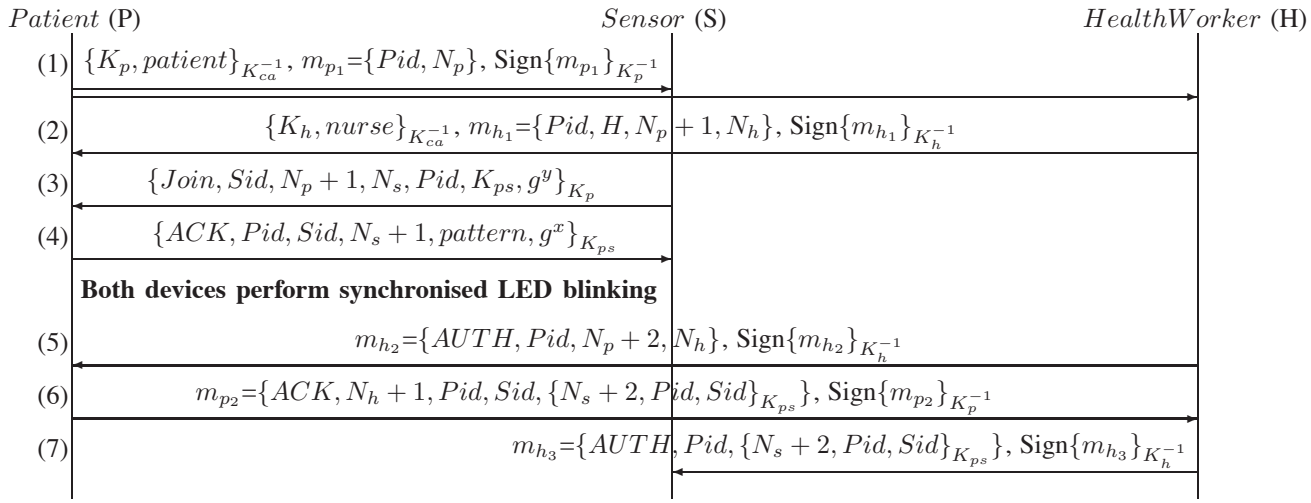

Fig. 1. The secure discovery protocol.

detect this and wipe state information such as encryption keys and received messages from their RAM. The notation used in our protocols is given in Table I

TABLE I
NOTATION USED

H	Healthcare worker's device
P	Patient's device
S	Wireless medical sensor
CA	Root Certification Authority Hospital
K_x, K_x^{-1}	Public and private key of x
$\{K_x, att\}_{K_y^{-1}}$	Attribute certificate bound to K_x, signed by K_y^{-1}
$\{m\}_{K_x}$	Encryption of m using public-key of x
$\{m\}_{K_{ab}}$	Encryption of m using secret key K_{ab}
Sign$\{m\}_{K_x^{-1}}$	Signing hash of m using private-key of x
N_x	Nonce of x
Pid	Unique identifier of BSN
Sid	Unique identifier of Sensor

B. Secure Discovery Protocol

Figure 1 summarises the secure discovery protocol using a sequence diagram. Broadly the protocol is divided in three phases: mutual authentication, information exchange and sensor association.

1) Mutual Authentication Phase: The patient's controller and the healthcare worker's device mutually authenticate each other to establish a session context in which the healthcare worker becomes the sole entity permitted to associate sensors to the patient's controller. The patient's controller periodically broadcasts message (*Msg 1*) to discover sensors and devices nearby. *Msg 1* consists of a certificate binding the attribute '*patient*' to the patient's public-key, $\{K_p, patient\}_{K_{ca}^{-1}}$ as well as a signed message m_{p_1} containing the BSN id, Pid and a nonce N_p. This enables the healthcare worker to verify the signature and the attribute (role) of the patient's controller. The healthcare worker then authenticates itself to the patient's controller by sending *Msg 2* which consists of the healthcare worker's attribute certificate, $\{K_h, nurse\}_{K_{ca}^{-1}}$ and a signed message m_{h_1}. In m_{h_1}, the nonce N_p+1 guarantees message freshness and N_h+1 is used to detect replay of *Msg 2*. Note that, the healthcare worker could also authenticate itself to the sensor, but this is not necessary as no interaction between them is needed at this stage. Furthermore, the sensor will need to authenticate the healthcare worker when the association between the sensor and the patient's device (*Msg 7*), is authorised.

contradict prev. sentence

2) Information Exchange Phase: The sensor and the patient's device perform a DH key exchange of their respective keyshares. The LED blinking pattern is sent to the sensor encrypted with a symmetric key. *Msg 1* enables the sensor to verify the patient's attribute certificate and public-key, K_p. The sensor then sends a join request (*Msg 3*) encrypted with the patient's public-key to the patient. *Msg 3* contains N_s that guarantees freshness of the message, and Sid as the sensor's ID. The sensor also generates a secret key K_{ps} and a Diffie-Hellman (DH) keyshare g^y. The patient's controller decrypts the receive join request using the private key and retrieves the secret-key and the sensor's DH keyshare. Instead of using public-key encryption, the patient uses the secret-key, K_{ps} to encrypt a blinking pattern together with its DH keyshare g^x and sends them back to the sensor in *Msg 4*. We rely on the sensor to generate a secret key K_{ps} instead of the patient's device to reduce message exchanges. Furthermore, only the sensor needs to authenticate the patient, but not vice versa as we rely on the healthcare worker to pair the correct devices together. As shown in Figure 2, if the patient's device generates K_{ps}, the sensor must convey its public-key to the patient before it can encrypt K_{ps}. This would require four messages instead of the original three; the number of cryptographic operations also increases as the sensor needs to generate and verify signatures in *Msg 1, 2, 3* as well as decrypting *Msg 3* using public-key cryptography before the secret key, K_{ps} is first used.

3) Sensor Association Phase: Only the patient's device and the sensor know the chosen blinking pattern to display for a specified period of time. After observing the synchonised blinking, the healthcare worker authorises the association by

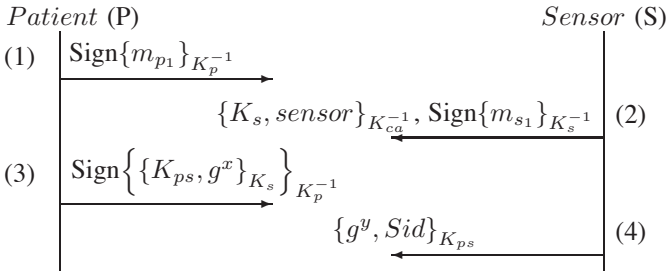

Fig. 2. Requiring the patient's device to generate K_{ps} incurs additional costs.

an $AUTH$ message (*Msg 5*) to the patient's device, which acknowledges it in *Msg 6*. N_p+2 and N_h in *Msg 5* guarantee the freshness of the authorisation message. The healthcare worker's device needs the sensorID for the $AUTH$ message so the ACK message (*Msg 6*) from the patient contains Sid, as well as a commitment encrypted with K_{ps} that only the sensor can decrypt and containing N_s+2 to prevent replay. *Msg 7* is an $AUTH$ message sent by the healthcare worker to the sensor confirming its association with the patient. The sensor can decrypt the commitment and verify freshness *Msg 7* by checking N_s+2. With both the sensor's and the patient's Diffie-Hellman keyshares exchanged, they derive g^{xy} as their secret key. K_{ps} would have been sufficient for the purpose of this paper, and the DH key exchange could have been removed. However, the DH secret key allows future extensions to cater for direct sensor-to-sensor authentication. In [13] we have proved that both the sensor and the patient's controller believe the secret-key they have derived, using BAN logic.

V. Key Management

Communication between the sensor and the patient's device can be encrypted using the DH secret key. However, no keys are shared between sensor pairs and routing all communication via the controller device would be inefficient. Moreover, medical events and network changes would typically be conveyed to all devices in the BSN network. Point-to-point notifications have redundancy and overhead as the same message must be encrypted n times and then sent to n parties. Encrypted broadcast provides an effective scheme for constrained environments that consist of mostly sensors with scarce resources. A shared group key, G can be established to enable all parties in the BSN to communicate with each other directly. This is based on the assumption that only authenticated, hospital approved sensors have been included in the BSN, so they are well behaved and do not impersonate other sensors in the same BSN. The key distribution and management scheme we propose does not rely on public-key cryptography to distribute initial keying materials, but uses symmetric-key cryptography and computation of hashes to significantly reduce computation on sensors.

Two one-way hash chains are used to generate shared group keys in which a key from each hash chain is concatenated and then hashed to produce the group key. The group key can be renewed by advancing both keychains forward to obtain the previous key from the corresponding hash chain. The next group key is then generated by hashing the concatenated keys from the chains. This scheme provides forward secrecy as it does not reveal any information about the hash chains and their keys for generating the group key if the group key is compromised.

A. Establishing a Group Key, G_i

The patient's controller in the BSN is responsible for the key distribution and management as it typically has higher computational capability and already shares a DH secret key with each sensor. It generates two key chains [14] each consisting of n keys using a one-way hash function, e.g., SHA-1 using a random number, generated as the initial key and the hash function is applied to the key to generate the next key. The next key is hashed repeatedly for $n-1$ times to produce the keychain. The first k *chain* contains keys K_1, K_2,..., K_n and the second x *chain* contains X_1, X_2,..., X_n.

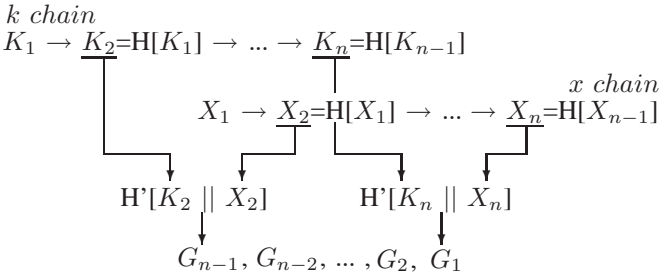

Fig. 3. Generating group keys.

As shown in Figure 3, both keychains are used in reverse order when K_n and X_n are disseminated, encrypted using the secret-key g^{xy}, shared between the patient's device and the sensor. On receipt, the sensor computes the hash of K_n and X_n concatenated as a group secret key, G_i. This key can be changed whenever a new sensor or device is added, but it must be renewed when an existing sensor is removed from the BSN or when the group key has been used for an extended period of time (c.f. Section V-B). Our key management scheme provides forward secrecy as there is no dependency between G_1, G_2, ..., and G_n. Compromising any of them does not reveal information about keys on k and x $chains$, so attackers cannot compute the keys to decrypt the data.

B. Re-Keying or Key Update

When a new sensor is discovered, the patient's device conveys the current K_i and X_i, encrypted with the shared secret-key, g^{xy} to enables the new sensor to compute the current group key, G_i.

When a sensor leaves the BSN, the group key must be renewed. The patient's device advances the keychains to obtain K_{i-1} and X_{i-1} from their respective chain. As shown in Figure 4, both keys are encrypted with the current X_i and then broadcast to all sensors/devices in a key update message. Upon receipt, the sensors compute H[K_{i-1}] and H[X_{i-1}] and ensure that the hash values match the current K_i and X_i respectively. This authenticates the source and contents of the

key update message as only the patient's controller knows the keychains. The sensors then derive the new group key, G_{i-1} using H'[$(K_{i-1} \mid\mid X_{i-1})$].

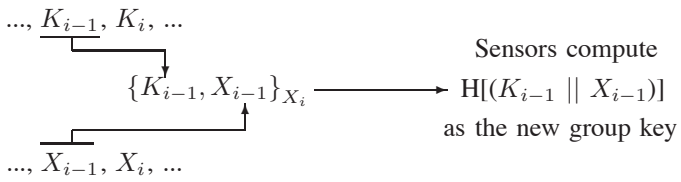

Fig. 4. Renewing the group key.

If a sensor is detected to have been compromised, a new group key must be derived. The patient's controller broadcasts the following:
$\{s_1, \{K_{i-1}, X_{i-1}, N_p\}_{g^{ax}}\}$, $\{s_2, \{K_{i-1}, X_{i-1}, N_p\}_{g^{bx}}\}$,..., $\{s_n, \{K_{i-1}, X_{i-1}, N_p\}_{g^{nx}}\}$

The compromised sensor's key is not used so it will not be able to obtain K_{i-1} and X_{i-1}, but other sensors can use their respective secret-key to decrypt the message and compute the new group key. N_p is used to detect replay of the message. Detection that a node is compromised, is very difficult – usually based on anomalous behavior and is not covered in this paper.

When all keys in the keychains have been used up, the patient's device generates two new keychains and similarly conveys the first key of both keychains (K_n and X_n) to all sensors encrypted with each individual sensor's secret-key to ensure that K_n and X_n are from the patient's device as encrypting them with the group key does not prove source authenticity. Unlike TESLA [15], [16] which uses keychains for authenticated broadcast, we use keychains to generate group keys for encryption. This has the advantage of efficient and effective authentication of the source of key update messages when re-keying or distributing key updates without relying on public-key cryptography.

C. Managing Sensor's Intermittent Connectivity

Transient failures occur if a sensor temporarily moves out of communication range and leaves the BSN for a short period. When rejoining the BSN they must still be able to communicate with other devices in the BSN so they have to obtain the latest K_i and X_i if they have missed the key update messages. We have devised a simple protocol to rejoin a BSN by requiring the patient's device to maintain the DH secret-key with the sensor for an extended period of time after the sensor has left. As shown in Figure 5, the group key could have been renewed multiple times if the BSN's membership changes after the sensor has left, so it will not have received any key updates from the patient's device to renew its group key. When it rejoins, it proves membership using its secret-key, g^{xy}. It sends a $challenge$ containing its Sid and a nonce N_s encrypted with g^{xy} to the patient's device, which $responds$ by encrypting the current K_i and X_i together with N_s + 1 using g^{xy}. The sensor can authenticate the message as it only shares the secret-key with the patient's device. This scheme avoids the high computation overheads of the full discovery protocol.

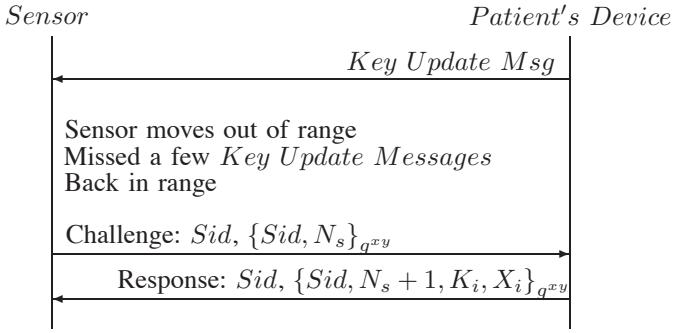

Fig. 5. A temporarily disconnected sensor regains its membership in BSN.

VI. Implementation and Evaluation

A. Elliptic Curve Cryptography Parameters

Both the secure discovery and key management protocols have been implemented on Tmote Sky[1] using the elliptic curve cryptography (ECC) library, TinyECC version 0.3 [9]. We use the recommended 160-bit Elliptic Curve domain parameters over F_p associated with a verifiably random parameters, i.e., *secp160r1*. A wide range of parameters can be selected from [17] and a base point G is chosen.

B. ElGamal Implementation

The public key encryption used to encrypt *Msg 3* in the secure discovery protocol is implemented using the ElGamal scheme [18]. The plaintext message is first embedded onto the elliptic curve E as a point, P_m. The sender then chooses a random bit pattern, r and computes two points, P_r = rG where G is the base point, and P_h = P_m + rP_B where P_B is the public key of the receiver.

The sensor sends both points, P_r and P_h to the patient device, which extracts the message point by computing P_s = $k_B P_r$ where k_B is the private key of the receiver. It then substracts this from P_h to get P_m = P_h - P_s. By expanding this equation, we show that P_m can be recovered as follows:

P_m = P_h - P_s
since P_h = P_m + $r(k_B G)$ where rP_B = $r(k_B G)$
we derive that P_m = P_m + $r(k_B G)$ - $k_B(rG)$

Based on the BigInteger library in TinyECC, we have implemented ElGamal public key encryption for TinyOS version 1.0x.

C. Diffie-Hellman Implementation

The ECC Diffie-Hellman (DH) key agreement requires the two parties to first compute a public point by choosing a random bit pattern k_i as the private key share and multipling it with the base point, G which is public, i.e., P_i = $k_i G$. This public point is then exchanged between two parties and the shared DH key can be computed by multiplying their respective private key k_i with the received public point, i.e.,

[1]Runs TinyOS and has 16-bit, 8 MHz Texas Instruments MSP430 processor with 48 KB of ROM and 10 KB of RAM

$P_{ab} = k_A(k_BG) = k_B(k_AG)$. Although a point on an elliptic curve is represented by (x, y), only the x value is used as the shared secret. x value is hashed to produce a 160-bit key for encryption. Using TinyECC, the DH key exchange protocol has been implemented using simple scalar point multiplication.

D. Skipjack Symmetric Key Encryption

We initially considered using Tmote Sky's on-chip Advance Encryption Standard (AES) library for symmetric-key encryption of data transfers. However, it only performs stand-alone encryption and does not support stand-alone decryption, presumably as the manufacturer assumed that all the data from the motes will be sent to a sink node. Although it supports inline AES encryption at the MAC layer, this is not suitable as we require application layer to be able to select a specific key based on the destination and software AES implementation on the mote results in unsatisfactory performance [19]. Consequently, we modified the MicaZ specific Skipjack algorithm implemented in TinySec [19] for Tmote Sky. All messages are encrypted using the Skipjack algorithm with a a symmetric key. The hash function SHA-1 produces 160-bit output which fits nicely into the key size of Skipjack. We used the Skipjack Cipher Block Chaining (CBC) mode with a block size of 8 bytes and non-repeating Initialisation Vector (IV). The battery level, or accelerometer reading of the mote has been used as the seed to generate the initial IV to produce different ciphertext from the same plaintext.

E. Measurements and Evaluation

Table II shows the execution time of various security operations on Tmote Sky. ElGamal takes 9.53s (variance is 0.167) to encrypt a 52 bytes message, while decryption takes 5.28s (variance is 0.00004), as the sender needs to compute two points, $P_r = rG$ and rP_B. Note that, according to [9], digital signature generation and verification take 4.361s and 5.448s respectively on TelosB which has the same design as same design as Tmote Sky. These results are much slower compared to MicaZ's implementation because hybrid multiplication has not been implemented on TelosB/Tmote Sky in TinyECC version 0.3. Symmetric-key encryption using Skipjack, is significantly faster than public-key encryption for the same plaintext length. Encryption takes 150μs and decryption takes 90μs. Consequently, the secure discovery protocol tries to minimise the use of signature generation/verification, as well as public-key encryption/decryption by using a symmetric secret-key to convey the blinking pattern to the sensor. The key management scheme also uses Skipjack for key distribution, key renewal and key updates.

Table III shows the codesize of security components. SHA-1 uses 2,442 bytes [9] for TelosB. The wireless communication, LEDs indication and basic standard library uses 16.43 Kb of ROM and 0.57 Kb of RAM with an overall codesize of 36.20 Kb (ROM) and 3.56 Kb (RAM) for the secure discovery protocol and key management. This could be further optimised by using TinyECC version 1.0 [20] which implements Barret reduction, repeated point doubling and affine coordinate point addition and doubling.

TABLE II
LATENCY FOR VARIOUS SECURITY OPERATIONS ON TMOTE SKY

Security Operations	Time
Public-key Encryption	9.53 s
Public-key Decryption	5.28 s
Signature Generation	4.26 s
Signature Verification	5.45 s
Diffie-Hellman Key Generation	5.97 s
Skipjack Encryption	150 μs
Skipjack Decryption	90 μs

TABLE III
CODESIZE FOR SECURITY OPERATIONS ON TMOTE SKY

Security Operations	ROM	RAM
Elgamal public-key encryption	10.42 Kb	1.54 Kb
Diffie-Hellman	6.92 Kb	1.12 Kb
Skipjack	4.41 KB	0.45 Kb

VII. DISCUSSION

In contrast to wireless sensor networks for environmental monitoring where sensors relay data to a sink node, BSNs for healthcare usually have a patient controller that manages its sensors in terms of configuration changes, i.e., modifying monitoring thresholds, managing addition and removal of sensors and actuators. Delegating the computational intensive security functions to the patient's controller can prolong battery life on sensors and support better security mechanisms such as signature generation and verification. However, this results in a single point of failure and hence it is important to ensure that the patient protects the device from malicious tampering and theft. In case of theft, the patient can manually turn off the sensors attached to his BSN to avoid information leakage.

Human intervention is needed to replace a failed battery and restart the sensor, but results in loss of the state information regarding which patient the sensor is associated with. Consequently, we advocate that upon discovery, the sensor stores the patient's public-key in its flash and when restarted, it authenticates and re-associates with the patient device without requiring a healthcare worker's intervention. The patient's device must keep a record of the sensor's information, e.g., the DH shared secret-key or the sensor's public-key to facilitate re-association. The sensor can be explicitly disassociated by erasing the public-key from the flash through a management action by the healthcare worker.

The use of a shared group key has a drawback in that it does not distinguish the sender of a message from other group members. Hence, it does not guarantee non-repudiation and implies that anyone in possession of the group key can modify the message content. As a result, the group key is purely used for ensuring message confidentiality and integrity in BSNs. Information sources which need to be authenticated can sign messages using the DH shared secret-key.

VIII. Conclusions and Future Work

We have presented an approach to securely discover sensors and associate them to the patient's controller to join the patient's BSN. The design takes account of the limitations of mobile devices/sensors – lack of screen display and constrained computational capability. Our approach to discovering sensors enables the devices to exchange security information such as public-keys, symmetric key and DH keyshares. More importantly, we provide a practical scheme of associating sensors to the patient's device which can only be done by an authorised healthcare worker.

We presented a novel key distribution and management scheme based on a keychain which only uses hash functions and symmetric-key encryption and does not rely on public-key cryptography, so is very efficient with low computational overheads. Distribution of group keys is simplified and key update messages can be authenticated easily because of the one-way property of the secure hash function. This facilitates efficient renewal of group keys to cater for membership changes.

The future work includes investigating the possibility of providing access control for the BSN to control interactions between individual sensors, e.g., invoking actions and sending event notifications directly to each other. This will require extension to the current key management scheme to enable sensors to authenticate each other and enforce access control policies.

References

[1] J. Andersen and J. E. Bardram, "BLIG: A New Approach for Sensor Identification, Grouping, and Authorisation in Body Sensor Networks ," in *Proc. 4th Int. Workshop on Wearable and Implantable Body Sensor Networks, Aachen, Germany, March 26 - 28*, 2007.

[2] D. Liu, P. Ning, and R. Li, "Establishing Pairwise Keys in Distributed Sensor Networks," *ACM Trans. on Information and System Security*, vol. 8, no. 1, pp. 41 – 77, Feb 2005.

[3] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proc. 9th ACM Conf. on Computer and communications security*. Washington, DC, USA: ACM, 2002.

[4] D. Balfanz, D. Smetters, P. Stewart, and H. Wong, "Talking to Strangers: Authentication in Ad-hoc Wireless Networks," in *Proc. Network and Distributed System Security Symp., San Diego*, 2002.

[5] F. Stajano and R. Anderson, "The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks," in *Proc. 7th Int. Workshop on Security Protocols*, ser. LCNS. Springer-Verlag, 1999.

[6] F. Stajano, "The Resurrecting Duckling – What Next?" in *Proc. 8th Int. Workshop on Security Protocols*, ser. LCNS, 2000.

[7] C. Jiang, B. Li, and H. Xu, "An Efficient Scheme for User Authentication in Wireless Sensor Networks," in *Proc. 21st Int. Conf. on Advanced Information Networking and Applications Workshops*. Washington, DC, USA: IEEE Computer Society, 2007.

[8] K. Malasri and L. Wang, "Addressing security in medical sensor networks," in *Proc. 1st ACM Int. Workshop on Systems and Networking Support for Healthcare and Assisted Living Environments*. San Juan, Puerto Rico: ACM, 2007.

[9] A. Liu, P. Kampanakis, and P. Ning, "TinyECC: Elliptic Curve Cryptography for Sensor Networks (Version 0.3)," Feb 2007. [Online]. Available: http://discovery.csc.ncsu.edu/software/TinyECC/

[10] R. Watro, D. Kong, S. fen Cuti, C. Gardiner, C. Lynn, and P. Kruus, "TinyPK: securing sensor networks with public key technology," in *Proc. 2nd ACM workshop on Security of ad hoc and sensor networks*. Washington DC, USA: ACM, 2004, pp. 59–64.

[11] H. Wang, B. Sheng, and Q. Li, "TelosB Implementation of Elliptic Curve Cryptography over Primary Field," Dept. of Computer Science, College of William and Mary, Tech. Rep. WM-CS-2005-12, October 2005.

[12] C. He and J. Mitchell, "Security Analysis and Improvements for IEEE 802.11i," in *Proc. Network and Distributed System Security Symp.*, 2005.

[13] S. L. Keoh, "Security proof of secure discovery protocol for body sensor networks," Aug 2008. [Online]. Available: http://www.doc.ic.ac.uk/slk/files/bsn-proof.pdf

[14] L. Lamport, "Password Authentication with Insecure Communication," *Communications of the ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[15] A. Perrig, R. Canetti, J. Tygar, and D. Song, "The TESLA Broadcast Authentication Protocol," *RSA Cryptobytes*, 2002.

[16] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar, "SPINS: Security Protocols for Sensor Networks," in *Proc. of Mobile Computing and Networking*, 2001, pp. 189–199.

[17] "SEC 2: Recommended Elliptic Curve Domain Parameters," Standards for Efficient Cryptography (SEC), Certicom Research, Tech. Rep., September 2000. [Online]. Available: http://www.secg.org/collateral/sec2_final.pdf

[18] M. Rosing, *Implementing Elliptic Curve Cryptography*. Manning Publications Co., 1999.

[19] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: a link layer security architecture for wireless sensor networks," in *Proc. 2nd Int. Conf. on Embedded Networked Sensor Systems*. Baltimore, MD, USA: ACM, 2004, pp. 162–175.

[20] A. Liu and P. Ning, "TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks," North Carolina State Univ., Dept of Computer Science, Tech. Rep. TR-2007-36, Nov 2007.