Hindawi Publishing Corporation
BioMed Research International
Volume 2014, Article ID 543051, 12 pages
http://dx.doi.org/10.1155/2014/543051

Research Article

Security Mechanism Based on Hospital Authentication Server for Secure Application of Implantable Medical Devices

Chang-Seop Park

Department of Computer Science, Dankook University, Cheonan 330-714, Republic of Korea

Correspondence should be addressed to Chang-Seop Park; csp0@dankook.ac.kr

Received 31 December 2013; Revised 1 June 2014; Accepted 20 June 2014; Published 24 July 2014

Academic Editor: Hesham H. Ali

After two recent security attacks against implantable medical devices (IMDs) have been reported, the privacy and security risks of IMDs have been widely recognized in the medical device market and research community, since the malfunctioning of IMDs might endanger the patient's life. During the last few years, a lot of researches have been carried out to address the security-related issues of IMDs, including privacy, safety, and accessibility issues. A physician accesses IMD through an external device called a programmer, for diagnosis and treatment. Hence, cryptographic key management between IMD and programmer is important to enforce a strict access control. In this paper, a new security architecture for the security of IMDs is proposed, based on a 3-Tier security model, where the programmer interacts with a Hospital Authentication Server, to get permissions to access IMDs. The proposed security architecture greatly simplifies the key management between IMDs and programmers. Also proposed is a security mechanism to guarantee the authenticity of the patient data collected from IMD and the nonrepudiation of the physician's treatment based on it. The proposed architecture and mechanism are analyzed and compared with several previous works, in terms of security and performance.

1. Introduction

Implantable medical devices (IMDs), such as implantable cardiac defibrillators, insulin pumps, and neurostimuli, monitor chronic disorder within the body and perform life-critical functions to treat cardiac arrhythmia, diabetes, and Parkinson's disease. For instance, implantable cardioverter defibrillators sensing cardiac events can store measurements such as electrocardiograms and administer an electrical shock to restore a normal heart rhythm, when a rapid heartbeat is sensed. Healthcare practitioners can extract data from IMD or modify its settings wirelessly, using an external device called a "programmer." Modern IMDs also support delivery of telemetry, for remote monitoring over long-range, high-bandwidth wireless links, so that chronic patients can be continuously monitored at home or work, without visiting a hospital [1, 2].

Recently, the privacy and security risks of IMDs have been widely recognized in the medical device market and research community [3], after two experimental security attacks have been reported, against a commercial implantable cardiac defibrillator and an insulin pump [4, 5]. As well as securing the wireless link between IMD and programmer, access to IMD should be carefully controlled to prevent illegal exposure of the patient data and life-threatening modification of its settings. Unfortunately, up-to-date commercial IMDs do not employ security mechanisms.

During the last few years, a lot of researches have been carried out addressing the privacy and security issues of IMD. Most of them have concentrated on the authenticated key establishment problems for setting up a secure channel between IMD and programmer. Even though security mechanisms [6] based on public-key cryptography have been proposed to establish a common secret between them, they are not suitable for resource-constraint IMDs, in terms of both computational complexity and energy consumption. Hence, the IMD security should either be based on lighter-weight symmetric encryption and authentication schemes [4, 7, 8] or employ a resource-rich personal device (e.g., smart phone) to mediate communication between an IMD and an external programmer [9–11]. Another approach for the IMD security is associated with the accessibility issues of IMDs, when an

emergency situation occurs. Suppose an unconscious patient with IMD enters an emergency room (ER) of a non-primary-care hospital. In order for ER personnel to access the IMD the patient has, some backdoors should be integrated for the programmer. Even though several techniques [10–13] have been proposed, each of them has its own inherent security weaknesses.

In this paper, a new security architecture for the security of IMDs is proposed, based on a 3-Tier security model, where the programmer interacts with a Hospital Authentication Server (HAS) to get permissions to access IMDs. Regardless of the type of cryptography for the authenticated key establishment, it is assumed in the research carried out so far that there is a single programmer to take care of several IMDs and that the keying material of each IMD is stored in the programmer. However, there are many IMDs and programmers, where any IMD should be able to be accessed from any programmer. In this case, there are two problems, in terms of security and scalability. The keying material of each IMD should be preinstalled into each programmer, so that the scalability problem occurs. Furthermore, if one of the programmers falls into an adversary's hands and the keying materials of IMDs are exposed, the patient's privacy can be compromised by eavesdropping, and the patient's health might even be endangered, by allowing an unauthorized access to IMDs. To address these problems, HAS is introduced, which plays the role of distributing access keys to IMD, according to the physician's privilege. Instead of installing the keying materials of IMDs into several programmers, HAS both maintains the access keys to several IMDs and distributes them to each authorized physician, via the programmer.

A security mechanism is also proposed in this paper to guarantee the authenticity of the patient data collected from IMD and the nonrepudiation of the physician's treatment based on it. An authorized physician usually modifies IMD's settings according to the patient data read out from the patient's IMD, for the purpose of increasing the efficacy of the treatment. However, if the physician's treatment associated with IMD is not correct or appropriate, a medical accident might occur. It could be due to a medical malpractice or a medical error. In order to clearly resolve the medical dispute, a record of the physician's treatments based on the observed patient data will be valuable corroborated facts. Hence, the treatment record should be securely maintained, with nonrepudiation. Regarding this issue, another security mechanism is introduced to ensure that the patient data extracted from its IMD is not modified by the physician illegally, and a record of the physician's treatments based on the patient data cannot be modified, either, even by HAS.

After an introduction of the current commercial IMDs, together with their usage for pervasive patient monitoring in Section 2, related research works to improve the IMD security are presented in Section 3. Security requirements and models for IMDs are investigated in Section 4 to help understand the security mechanism proposed in Section 5. Finally, various features of the security mechanism proposed for the IMD security are analyzed in Section 6 and compared with other research works in terms of security and performance.

2. Current Commercial IMDs and Pervasive Monitoring Systems

IMDs are capable of measuring and altering the physiological characteristics of a patient. IMDs receive commands from an external device (programmer) that enable the adjustment of settings on the implanted device. They can also send information on their current status (as well as sensed data) to an external device. Modern IMDs contain the high-capacity lithium batteries that last five to seven years and the ultralow-power microprocessor with about 128 Kbytes of RAM for telemetry storage. Radio frequency (RF) telemetry, which operates within the Medical Implant Communications Service (MICS) Band, is used for wireless communication between IMD and programmer in the hospital or clinic. Using the MICS Band prevents interference with home electronics such as microwaves, cell phones, and baby monitors. The MICS Band (402 to 405 MHz) allows for 250 Kbps and read range of up to 5 meters.

To open a wireless communication channel between IMD and programmer, a magnet switch inside IMD should be activated by RF command. Then, the programmer searches for activated IMDs within telemetry range, by sending an *ID-Request* command. As shown in Figure 1(a), when receiving an *ID-Response* command from a target IMD, the programmer starts a patient session. The programmer interrogates IMD to gather measured data since the last patient session, through exchanging the *Read-Request* and *Read-Response* commands. IMD can also store patient-related information, such as patient name and date-of-birth, which can be viewed during a patient session. This information is typically programmed into IMD at the time of implant. The device settings for treatment can be reprogrammed into IMD, through the *Write-Request* and *Write-Response* commands. The *Close* command from the programmer ends the current patient session with IMD.

On the other hand, IMD manufacturers, such as Medtronics [14] and Biotronik [15], operate comprehensive Internet-based remote monitoring services to patients with their IMDs and their clinics. The home monitoring device is programmed to wake up IMD, per a schedule defined by physicians. Applicable data are then extracted from IMD and sent to the home monitoring device that communicates them to an Internet site that can be accessed by physicians as shown in Figure 1(b) [16]. Contrary to the programmer that can exchange both *Read* and *Write* commands, the home monitoring device can exchange the *Read* command only with IMD. Namely, the home monitoring device cannot be used to change the settings on IMDs for patient treatment. From now on, the security mechanisms applicable to Figure 1(a) are investigated.

3. Related Works

Since current commercial IMDs and programmers do not employ any security mechanisms for access control and transmission protection, a lot of research papers have been recently proposed to address the security issues associated

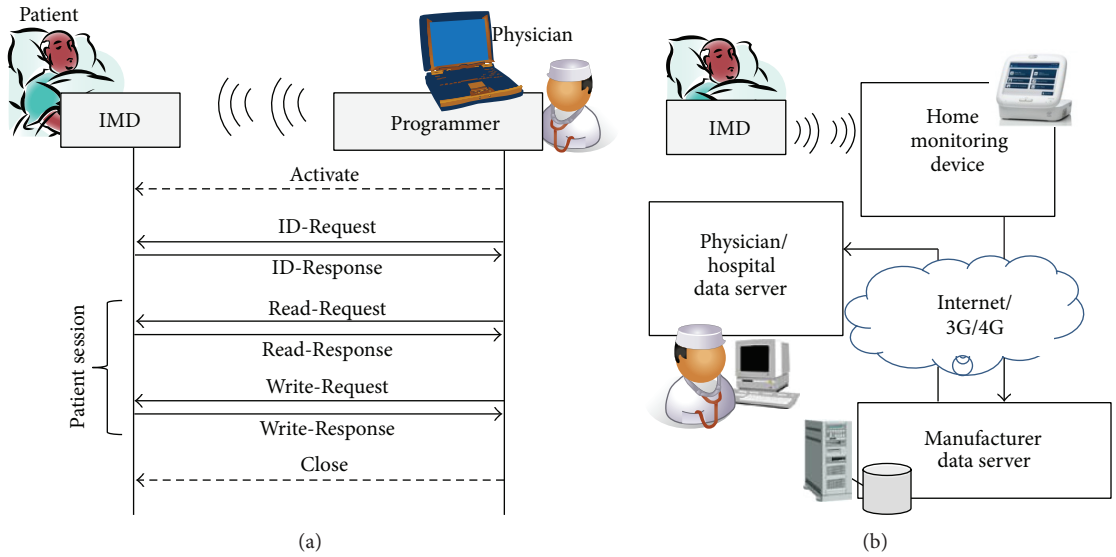

Figure 1: (a) Generic patient session protocol and (b) remote monitoring service for IMD patients.

with IMDs and programmers. Most of them are based on the secret information (e.g., password, secret key) preshared between IMDs and programmers, much like in other network security environments. Basically, the IMD security can be approached through access control at the programmer level. Only authorized personnel can access the programmer capable of communicating with IMD. However, if an adversary possesses his own programmer, this approach cannot be effective. The study in [7] proposed a password-based access control to IMD, where an adversary without knowledge of the password would be unable to access IMD. However, this does not provide any method to secure the transmission of the password to IMD. A challenge-response authentication has been proposed, based on the secret key shared between IMD and programmer [4, 8]. In particular, [4] is specific about the key management for the IMDs: the programmer keeps a master key K_M, and each IMD has an IMD-specific key $K_{\text{dID}} = f(K_M, \text{dID})$, where dID is a serial number or identity of IMD and $f(\cdot)$ is a pseudorandom function. So, given dID, the programmer can share K_{dID} with IMD, which can be used to gain access to IMD, either to send it commands or to read out medical data. Interestingly, a proximity-based access control scheme has been designed by [6]. Assuming that no secret information is shared between IMD and programmer, a session key is derived using a Diffie-Hellman key exchange. The authenticity of the Diffie-Hellman parameters exchanged is guaranteed by executing a distance bounding protocol [17, 18]: namely, wireless interaction with IMD is denied, unless the proximity of IMD is verified.

Personal devices, such as *Guardian* [9], *Communication Cloaker* [10], and *Shield* [11], have been introduced for the purpose of offloading long distance communication and intensive processing on the personal devices. In particular, *Guardian* plays the role of the key distribution center for IMD and programmer, assuming that *Guardian* has the public key of the programmer and a secret key is preshared between IMD and *Guardian*.

Some research efforts [10–13] on solving the tension between the security and the safety of IMD have been carried out, too. In emergency situations, where an unconscious patient with IMD enters an unfamiliar emergency room, the emergency medical staff would not have access to IMD, if a strict access control mechanism based on the preshared secret is employed, and the secret information is not available to them. In this situation, access to the IMD would be protected by a password that is engraved on the back of the medical bracelet the patient always wears or encoded as a 2D barcode and tattooed onto the patient's skin. In addition to regular tattoos with black or colored inks [12], it is now possible to get specialty tattoos that are only visible under UV lights [13]. *Communication Cloaker* and *Shield* can be used for emergency situations. Namely, if they are removed from the patient, the system changes its access policy to allow any programmer to access IMD.

4. Security Requirements and Security Models for IMD

Since patient data should be available only to the authorized personnel, strict access control to IMD is indispensable, and the wireless link between IMD and programmer should be cryptographically protected. In particular, due to the wireless reprogramming capability, an adversary could conceivably cause direct physical harm to a patient, by sending commands to modify the treatment parameters of IMD. First, the wireless link should be protected, for both confidentiality and authenticity of the patient data. Second, the *Read-Request* and *Write-Request* commands in Figure 1 should be sent only by the authorized programmer (physician). Third, the

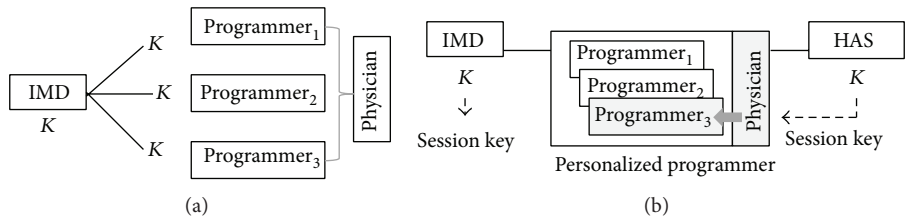

Figure 2: (a) 2-Tier security model and (b) 3-Tier security model.

authenticity of the patient data collected from IMD should be guaranteed, wherever it is stored. Fourth, the physician's treatment based on the collected patient data should be correct and careful to prevent medical accidents. Therefore, all the actions the physician takes for treatment should be recorded with nonrepudiation capability to solve the dispute when a medical accident occurs.

Figure 2 shows two kinds of security models for IMD, the 2-Tier security model and 3-Tier security model. In order to use multiple programmers for a single IMD, a long-term key K should be stored into each programmer, as in Figure 2(a). Previous works for the IMD security are based on this 2-Tier security model. However, it is not easy to manage the long-term keys of IMDs in multiple programmers, when a lot of IMDs are present. In particular, if a programmer is stolen, the long-term keys of several IMDs can be exposed to the adversaries. In this paper, we propose a 3-Tier security model with the Hospital Authentication Server (HAS), as in Figure 2(b). There are multiple programmers handled by the physician. A basic function of each programmer is to send commands to IMD and to receive patient data from IMD. Even though each programmer does not store the long-term keys of IMDs, the common cryptographic modules (such as encryption and decryption algorithm) to secure the transmission between them are embedded into the programmer. On the other hand, HAS maintains the long-term keys of IMDs. The programmer is personalized, when the physician inserts his/her smartcard into it. The smartcard contains the physician's credentials, based on which session keys, derived from the long-term key, are passed securely to the personalized programmer. The session keys are eventually used to retrieve the patient data and to change the treatment parameters of IMD.

5. Proposed Security Mechanisms for the IMD

5.1. Assumptions and Design Principles. First, the security mechanism proposed here is based on the 3-Tier security model in Figure 2(b). There are many programmers available to physicians and patients. However, both patient- and physician-centric information, such as credentials, are not stored in the programmers. Second, the programmer is personalized, when the physician inserts his/her smartcard into it. The smartcard contains the long-term key K_{pID} preshared with HAS, as well as the physician's private key for signature SK_{pID}. The physician's corresponding public key PK_{pID} for signature verification is stored in HAS. Third, a long-term key K_{dID} is preshared between IMD and HAS, when IMD is initialized for the first time. The IMD initialization process is shown in Section 5.2. Fourth, in order to access IMD, the physician should obtain access keys from HAS after successfully authenticating to HAS. There are two kinds of access keys: Read-Key and Write-Key. One is for retrieving the patient data from IMD, and the other is for granting a privilege to the physician to send a treatment command, such as device parameter changes. In particular, the Write-Key is generated by HAS based on the patient data retrieved from IMD, as well as corresponding treatment, so that the physician's treatment can be reactively verified, in the case of medical accident. Fifth, since HAS has all the security-related information of the patients, IMDs, and physicians, it is assumed that they are securely protected and maintained. The notations to be used in Sections 5 and 6 are summarized in Notations section.

5.2. IMD Initialization. Prior to implanting IMD, it is initialized, by installing security-related information. As in Figure 3, the following security-related information $\{\text{dID}, \text{SN}_0, K_{\text{dID}}, \text{PatientInfo}\}$ is securely stored into the IMD and HAS, where K_{dID} is the IMD's long-term key and SN_0 is an initial patient session number.

During the IMD initialization phase, a list of primary care physicians, pID(dID), for the patient with IMD whose identification number is dID is determined. For each $\text{pID} \in \text{pID(dID)}$, the corresponding physician's long-term key K_{pID} and public key PK_{pID} for signature verification are also stored in the HAS. The physician's smartcard contains pID, K_{pID}, and private key SK_{pID} for signature generation.

5.3. Proposed Secure Patient Session. In order to initiate a new patient session through a programmer, the physician first personalizes the programmer, by inserting his/her smartcard containing both the long-term key K_{pID} and the physician's private key SK_{pID} into it. After exchanging the *ID-Request* and *ID-Response* commands during the IDENTIFICATION session, the personalized programmer obtains the device identification number dID of the target IMD, as well as the current session number SN_j. To read from or write to IMD, the programmer requests permissions from HAS. The IDENTIFICATION session is performed based on the RFID to save the battery power on IMD. The current IMD is equipped with a RFID tag, and the programmer also has a RFID reader with it [19].

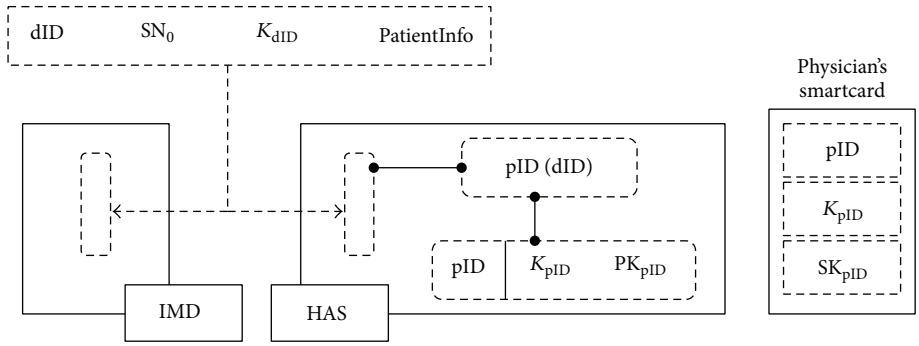

Figure 3: IMD initialization.

5.3.1. READ Session. The READ session consisting of the four commands is for the personalized programmer, namely, physician, to read out the patient data stored in the target IMD. The *Read-Auth-Request* and *Read-Auth-Response* commands are protected by the session key derived from the physician's long-term key K_{pID} preshared with HAS, while both *Read-Request* and *Read-Response* commands are protected by the session key derived from the IMD's long-term key K_{dID}, preshared with HAS.

(Step-①②) The personalized programmer first computes the session key K_j using the key derivation function $kdf(\cdot)$:

$$K_j = kdf\left(K_{pID}, SN_j, dID, pID\right), \tag{1}$$

where SN_j is the jth patient session number, dID is IMD's device identification number, and pID is physician's identification number. Then, the *Read-Auth-Request* $\{SN_j, dID, pID, MAC(K_j)\}$ command is sent to HAS, where $MAC(K_j)$ is the message authentication code computed over all preceding fields of the command using K_j. When receiving it, HAS also computes K_j and verifies $MAC(K_j)$. If the verification is successful, HAS computes the Read-Key, Kr_j, based on K_{dID}:

$$Kr_j = kdf\left(K_{dID}, SN_j, dID, pID\right), \tag{2}$$

which is sent back to the personalized programmer through the *Read-Auth-Response* $\{SN_j, [Kr_j]K_j, MAC(K_j)\}$ command. The Read-Key is a kind of permission from HAS for the personalized programmer to access the patient data in IMD. Now, using the Read-Key, the *Read-Request* command is sent to IMD.

(Step-③④) When receiving the *Read-Request* command, IMD verifies the MAC value, after deriving the Read-Key, Kr_j, in the same way as the HAS did. If the verification is successful, the $PatientData_j$ and its integrity-check value $Auth_j$ are sent to the personalized programmer, through the *Read-Response* command. Otherwise, IMD just drops the command and takes no further action. The integrity-check value $Auth_j = h(K_{dID}, SN_j, PatientData_j)$ guarantees that the $PatientData_j$ is really sent from IMD. However, the personalized programmer cannot verify the integrity-check value, since it does not have the long-term key K_{dID}. When receiving the *Read-Response* command, the programmer also checks the MAC value and obtains the $PatientData_j$. If the purpose of initiating this patient session is only to retrieve the current patient data, the patient session is terminated here.

5.3.2. WRITE Session. The WRITE session also consists of the four commands whose purpose is for the patient treatment, by adjusting the running parameters of IMD. In order for the personalized programmer to obtain permission for the write operation to IMD, the Write-Key is provided to the personalized programmer by HAS. In particular, since the physician's treatment, $Conf_j$, should be based on the $PatientData_j$ collected from IMD during the READ session, both $PatientData_j$ and $Conf_j$ should be applied, when deriving the Write-Key.

(Step-❶❷) If a subsequent action, such as changing parameters of IMD, should be taken for the treatment, based on the retrieved patient data, the personalized programmer should obtain a Write-Key from HAS. The personalized programmer generates the treatment command $Conf_j$ based on $PatientData_j$ and computes Z_j and Sig_j, where

$$\begin{aligned} Z_j &= \left[PatientData_j, Conf_j\right] K_j, \\ Sig_j &= Sig\left(SK_{pID}\right) \text{ is obtained from } W_j, \\ W_j &= \left[SN_j, dID, pID, PatientData_j, \right. \\ &\qquad \left. Conf_j, Auth_j, Sig\left(SK_{pID}\right)\right]. \end{aligned} \tag{3}$$

Z_j is the encryption of the confidential data ($PatientData_j$, $Conf_j$) and $Sig_j = Sig(SK_{pID})$ is the signature computed over $[SN_j, dID, pID, PatientData_j, Conf_j, Auth_j]$ using the physician's private key SK_{pID}. Since W_j contains the physician's treatment, $Conf_j$, based on the patient physiological data, $PatientData_j$, it can be used for evidence data when a medical dispute occurs. When receiving the *Write-Auth-Request* $\{SN_j, dID, pID, Z_j, Auth_j, Sig_j, MAC(K_j)\}$ command, HAS first verifies both the MAC value and the signature value and checks if the received $Auth_j$ is identical to the computed $Auth_j$. If the verifications are successful, HAS stores W_j in its database after obtaining $Data_j$ and $Conf_j$ from Z_j. Finally, the Write-Key, Kw_j, is computed and sent to the

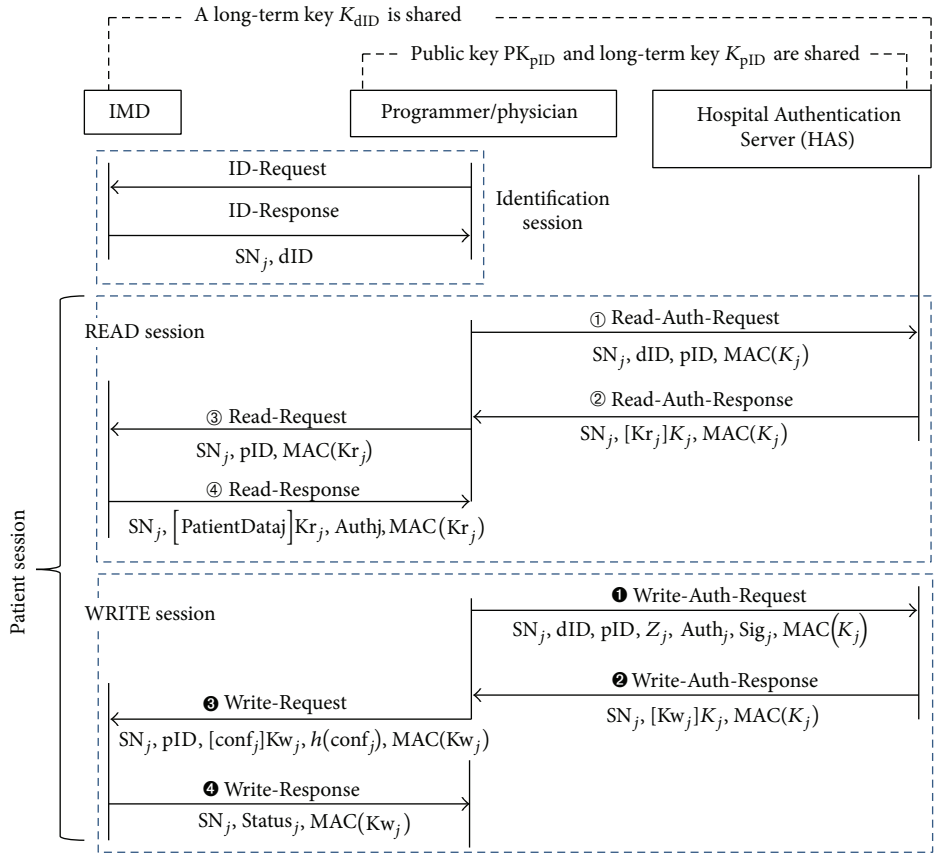

Figure 4: Proposed protocol for secure patient session.

personalized programmer, through the *Write-Auth-Response* $\{SN_j, [Kw_j]K_j, MAC(K_j)\}$ command:

$$Kw_j = kdf\left(K_{dID}, SN_j, dID, pID, PatientData_j, h\left(Config_j\right)\right). \quad (4)$$

(Step-❸❹) After receiving the *Write-Request* $\{SN_j, pID, [Conf_j]Kw_j, h(Conf_j), MAC(Kw_j)\}$ command, IMD first computes its own Kw_j and verifies the MAC value. A reason to use $h(Conf_j)$ instead of $Conf_j$ to generate the Write-Key is for confidentiality of the treatment parameters. If the verification is successful, IMD changes its parameters according to $Conf_j$ and responds with the *Write-Response* $\{SN_j, Status_j, MAC(Kw_j)\}$ command, where $Status_j$ is "*successful*."

6. Analysis and Comparisons

6.1. Threat Model and Security Assumptions. We consider two types of adversaries with polynomially bounded computational power: an outside adversary and an inside adversary as in Figure 5. The outside adversary monitors and manipulates the wireless links among IMD, the personalized programmer, and HAS, for the purpose of accessing the patient data and modifying the physician's treatment to the patient. On the other hand, the inside adversary, including the physician, tries to falsify the patient data retrieved from IMD and to give unsuitable treatment to the patient. When a medical accident occurs, it might be due to unintentional medical malpractice or intentional error. Whether or not it is intentional, the effect of the medical malpractice and error is harmful to the patient. Therefore, all the actions the physician takes for treatment should be recorded with nonrepudiation capability to solve the dispute when a medical accident occurs. Denial-of-service attacks, such as an energy depletion attack, are excluded from our adversary model.

As mentioned in Section 5.1, it is assumed that two long-term keys, K_{dID} of IMD and K_{pID} of the physician, are preestablished. The physician's public key PK_{pID} is also enrolled in HAS, and the corresponding private key SK_{pID} is kept by the physician. It is assumed here that they are securely protected.

6.2. Formal Verification of the Proposed Protocol. When designing a cryptographic protocol such as the one in Figure 4, one often uses arguments such as "since this message was digitally signed by one principal A, a second principal B can be sure it came from A" in informal proofs justifying how the cryptographic protocol works. However, in such informal proofs, it is easy to overlook an essential assumption, such as a

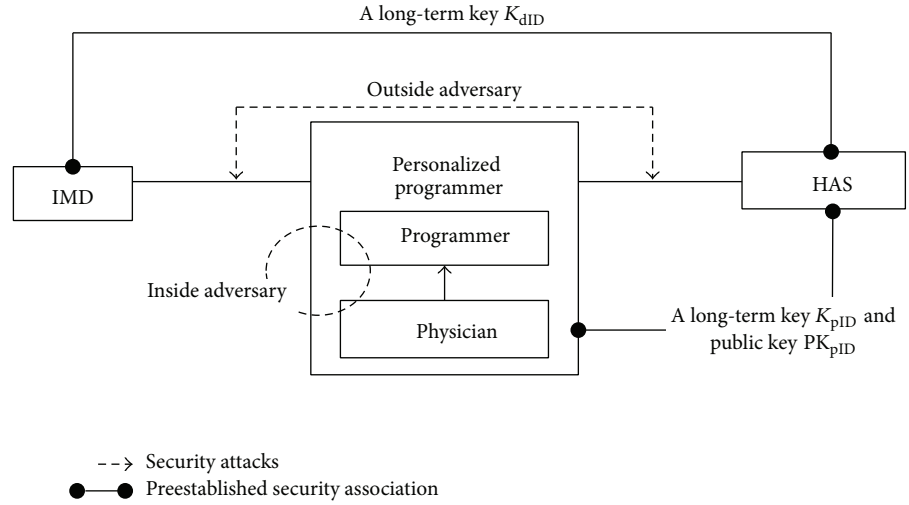

Figure 5: Threat model and preestablished security associations.

$A \models X$	A believes X
$A \lhd X$	A sees X
$A \mid\sim X$	A once said X
$A \Rightarrow X$	A has jurisdiction over X
$\#(X)$	X is fresh
$\{X\}_K$	X is encrypted using key K
$(A \overset{K}{\leftrightarrow} B)$	A shares a key K with B

$$\frac{A \models B \mid\sim X}{A \models (A \overset{K}{\leftrightarrow} B),\ A \lhd \{X\}_K}\ (\text{Message meaning}) \qquad \frac{A \models X}{A \models B \Rightarrow X,\ A \models B \models X}\ (\text{Jurisdiction})$$

$$\frac{A \models B \models X}{A \models \#(X),\ A \models B \mid\sim X}\ (\text{Nonce verification}) \qquad \frac{A \models \#(X, Y)}{A \models \#(X)}\ (\text{Fresh}) \qquad \frac{A \models B \models X}{A \models B \models (X, Y)}\ (\text{Select})$$

Figure 6: Notations and inference rules of BAN logic.

trust relation or the belief that a message is not a replay from a previous session. Therefore, it is desirable to write such proofs in a formal method to help in pointing the weaknesses of the proposed cryptographic protocol.

There are three principals (programmer, IMD, and HAS) in the protocol proposed in Figure 4. What should be proven through the formal method is that the session key shared between any two principals is not exposed to the attackers (key authentication) and one principal is assured that a second principal actually has possession of the session key (key confirmation).

In this section, the correctness of the proposed cryptographic protocol is formally validated, based on BAN logic. The notations and five inference rules of the BAN logic are shown in Figure 6. The symbols A and B denote specific principals involved in a particular protocol, while X and Y are formulas. The details on notations and logical postulates of the BAN logic can be found in [20]. In particular, the definition of $\{X\}_K$ is extended to denote both encryption of X with K and that$X$ is integrity protected with K, since the purpose of the rule $\{A \models B \mid\sim X$ derived from $A \models (A \overset{k}{\leftrightarrow} b),\ A \lhd \{X\}_K\}$ is to verify that X is sent from B.

We verify the correctness of the proposed protocols, the READ session and WRITE session, of Figure 4. However, since the two protocols are identical in terms of the way of deriving and using the session keys, the correctness of the READ session protocol is verified. For simplicity of notations, P, H, and I denote programmer/physician, HAS, and IMD, respectively. The idealized "READ session" protocol of Figure 4 is formulated as follows:

$$\begin{aligned}
P &\longrightarrow H : \left\{\mathrm{SN}_j, P \overset{K_j}{\longleftrightarrow} H\right\}_{K_j},\\
P &\longleftarrow H : \left\{\mathrm{SN}_j, \left\{P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I\right\}_{K_j}\right\}_{K_j},\\
I &\longleftarrow P : \left\{\mathrm{SN}_j, \left\{P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I\right\}_{\mathrm{Kr}_j}\right\}_{\mathrm{Kr}_j},\\
I &\longrightarrow P : \left\{\mathrm{SN}_j, \left\{P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I\right\}_{\mathrm{Kr}_j}\right\}_{\mathrm{Kr}_j}.
\end{aligned} \tag{5}$$

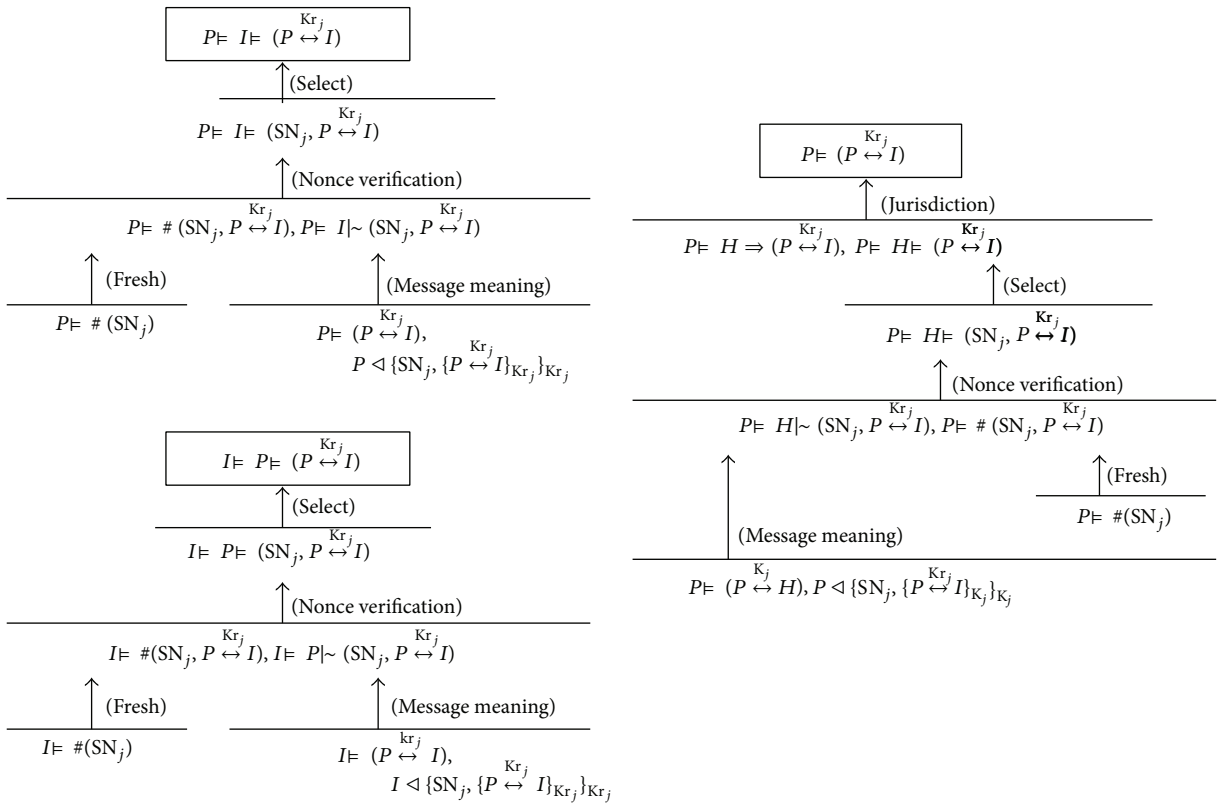

Figure 7: Formal verification for correctness of the proposed protocol.

The following are the initial assumptions for analysis, while some assumptions unused for analysis are omitted here:

$$\begin{aligned} P \models H &\Longrightarrow \left(P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I\right), \\ I &\models \left(P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I\right), \\ P &\models \#\left(\mathrm{SN}_j\right), \\ I &\models \#\left(\mathrm{SN}_j\right), \end{aligned} \tag{6}$$

$$P \models \left(P \overset{K_j}{\longleftrightarrow} H\right) \quad /* \text{ since } K_j \text{ is derived from } K_{\mathrm{pID}} \text{ shared between } P \text{ and } H */, \tag{7}$$

$$H \models \left(P \overset{K_j}{\longleftrightarrow} H\right) \quad /* \text{ since } K_j \text{ is derived from } K_{\mathrm{pID}} \text{ shared between } P \text{ and } H */. \tag{8}$$

The final goal of the protocol is $P \models (P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I), I \models (P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I), P \models I \models (P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I)$, and $I \models P \models (P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I)$, where $I \models (P \overset{\mathrm{Kr}_j}{\longleftrightarrow} I)$ is an initial assumption, and no proof is needed. The first two goals guarantee the mutually authenticated key establishment (*Key Authentication*) between P and I, while the other two goals are for *Key Confirmation*, which is the property whereby one principal is assured that a second principal actually has possession of the key Kr_j. As shown in Figure 7, both *Key Authentication* and *Key Confirmation* properties are embedded into the READ session protocol of Figure 4.

6.3. Forgery and Replay Attacks by an Outside Adzversary. All the commands between IMD and personalized programmer are secured using the Read-Key, $\mathrm{Kr}_j = kdf(K_{\mathrm{dID}}, \mathrm{SN}_j, \mathrm{dID}, \mathrm{pID})$, and Write-Key, $\mathrm{Kw}_j = kdf(K_{\mathrm{dID}}, \mathrm{SN}_j, \mathrm{dID}, \mathrm{pID}, \mathrm{PatientData}_j, h(\mathrm{Config}_j))$, which are a kind of session keys for the current patient session. Since IMD/physician identification numbers, dID and pID, as well as the current session number, SN_j, are used to derive the session keys, the freshness of each command can be guaranteed to protect from a replay attack. In particular, the patient data $\mathrm{PatientData}_j$ and physician's treatment Conf_j are symmetrically protected against eavesdropping, for patient privacy. On the other hand, the session key, $K_j = kdf(K_{\mathrm{pID}}, \mathrm{SN}_j, \mathrm{dID}, \mathrm{pID})$, derived from the physician's long-term key, K_{pID}, is employed to secure commands exchanged between the HAS and personalized programmer. Hence, without knowing K_{dID} and K_{pID}, it is not feasible to forge or replay the commands.

6.4. Mutual Authentication between Physician and HAS. The *Read-Auth-Request* command (①) plays the roles of both authenticating the personalized programmer (namely, physician) to the HAS and sharing the session key, $K_j = kdf(\mathrm{K_{pID}}, \mathrm{SN}_j, \mathrm{dID}, \mathrm{pID})$, with HAS. The physician authentication is based on his/her long-term key, $\mathrm{K_{pID}}$, shared with the HAS:

$$\begin{aligned} &\textit{Read-Auth-Request}\left\{\mathrm{SN}_j, \mathrm{dID}, \mathrm{pID}, \mathrm{MAC}\left(K_j\right)\right\}, \\ &\textit{Read-Auth-Response}\left\{\mathrm{SN}_j, \left[\mathrm{Kr}_j\right] K_j, \mathrm{MAC}\left(K_j\right)\right\}. \end{aligned} \tag{9}$$

The HAS maintains a list of primary care physicians PID(dID) authorized to access each dID registered. If pID $\in$ PID(dID), then a Read-Key, Kr_j, is granted to the physician through the *Read-Auth-Response* command (②). Otherwise, the command is silently dropped, or an error message is sent back to the physician. In particular, the successful verification on the MAC value of the *Read-Auth-Response* command corresponds to the successful authentication on the HAS.

6.5. Protection against Patient Data Falsification. A reason to include pID in the *Read-Request* command is for IMD to compute a session key, $\mathrm{Kr}_j = kdf(K_{\mathrm{dID}}, \mathrm{SN}_j, \mathrm{dID}, \mathrm{pID})$, to be shared with the personalized programmer. The *Read-Response* command contains two fields: $[\mathrm{PatientData}_j]\mathrm{Kr}_j$ and Auth_j, where $\mathrm{PatientData}_j$ is encrypted for the patient privacy and Auth_j is the origin-check value for $\mathrm{PatientData}_j$:

$$\begin{aligned} &\textit{Read-Request}\left\{\mathrm{SN}_j, \mathrm{pID}, \mathrm{MAC}\left(\mathrm{Kr}_j\right)\right\}, \\ &\textit{Read-Response}\left\{\mathrm{SN}_j, \left[\mathrm{PatientData}_j\right] \mathrm{Kr}_j, \right. \\ &\qquad \left. \mathrm{Auth}_j, \mathrm{MAC}\left(\mathrm{Kr}_j\right)\right\}. \end{aligned} \tag{10}$$

The current programmer stores the patient data in plaintext form retrieved from IMD into its internal or external memory. Since the patient data is not protected at all, as long as it remains in the programmer, there is a possibility that it can be modified intentionally or accidentally, before it is sent to HAS. Even though the physician is authorized to retrieve the patient data, its origin and integrity should be guaranteed, so that even the physician cannot modify it. Hence, the patient data is accompanied by the origin-check value $\mathrm{Auth}_j = h(K_{\mathrm{dID}}, \mathrm{SN}_j, \mathrm{PatientData}_j)$. Since the origin-check value is computed based on the IMD's long-term key, K_{dID}, shared between IMD and HAS, it can be verified by HAS when it is finally stored in the HAS.

6.6. Authenticity of the Physician's Treatment with Nonrepudiation. After examining the patient data, $\mathrm{PatientData}_j$, retrieved from IMD, the physician generates a treatment, Conf_j, based on which IMD's running parameters are changed. However, if the physician's treatment associated with IMD is not correct or appropriate, a medical accident might occur. In order to clearly resolve the medical dispute, a record of the physician's treatments based on the observed patient data will be valuable corroborated facts. Hence, the treatment record should be securely maintained with nonrepudiation. In order for the physician to obtain the Write-Key, Kw_j, from HAS, he/she must digitally sign both patient data and the corresponding treatment as follows:

$$\begin{aligned} &\textit{Write-Auth-Request}\left\{\mathrm{SN}_j, \mathrm{dID}, \mathrm{pID}, Z_j, \mathrm{Auth}_j, \right. \\ &\qquad \left. \mathrm{Sig}_j, \mathrm{MAC}\left(K_j\right)\right\}, \quad \text{where} \\ &\mathrm{Sig}_j = \mathrm{Sig}\left(\mathrm{SK_{pID}}\right) \text{ is obtained from } W_j, \\ &W_j = \left[\mathrm{SN}_j, \mathrm{dID}, \mathrm{pID}, \mathrm{PatientData}_j, \right. \\ &\qquad \left. \mathrm{Conf}_j, \mathrm{Auth}_j, \mathrm{Sig}\left(\mathrm{SK_{pID}}\right)\right], \\ &\textit{Write-Auth-Response}\left\{\mathrm{SN}_j, \left[\mathrm{Kw}_j\right] K_j, \right. \\ &\qquad \left. \mathrm{MAC}\left(K_j\right)\right\}, \quad \text{where} \\ &\mathrm{Kw}_j = kdf\left(K_{\mathrm{dID}}, \mathrm{SN}_j, \mathrm{dID}, \mathrm{pID}, \mathrm{PatientData}_j, \right. \\ &\qquad \left. h\left(\mathrm{Config}_j\right)\right). \end{aligned} \tag{11}$$

Suppose the physician is an inside adversary. Then, the following two security attacks can be imagined: the first is to generate an abnormal treatment, given the correct patient data. The second is to modify the patient data and generate a normal treatment corresponding to the modified patient data. In the first case, if a medical accident occurs, the liability of the physician is proven, due to the physician's signature. In the second case, it is not possible to modify the patient data arbitrarily by the physician, since it is protected with $\mathrm{Auth}_j = h(K_{\mathrm{dID}}, \mathrm{SN}_j, \mathrm{PatientData}_j)$, as described in Section 6.5.

6.7. Emergency Care Issues. In emergency situations, where an unconscious patient with IMD should be taken care of in either an unfamiliar emergency room of a different hospital or an emergency vehicle, IMD should first be disabled for emergency care. However, the emergency medical staff would not have access to IMD, if the access key is not available. Several methods [10–13], such as wearing personal devices or medical bracelets with the password/secret key, have been proposed to access IMD under such emergency situations. They can be effective methods, as long as the personal device or medical bracelet is not lost. Here, the medical bracelet method is employed for emergency care:

$$\begin{aligned} &\mathrm{IMD} \longleftarrow \text{programmer:} \\ &\quad \textit{Emergency-Request}\left\{\mathrm{SN}_j, \mathrm{dID}, \mathrm{MAC}\left(\mathrm{Ke}_j\right)\right\}, \\ &\mathrm{IMD} \longrightarrow \text{programmer:} \\ &\quad \textit{Emergency-Response}\left\{\mathrm{SN}_j, \mathrm{Status}_j, \mathrm{MAC}\left(\mathrm{Ke}_j\right)\right\} \end{aligned} \tag{12}$$

Suppose a patient wears a medical bracelet with K_{dID}. Then, the emergency key, $\mathrm{Ke}_j = kdf(K_{\mathrm{dID}}, \mathrm{SN}_j, \mathrm{dID})$, is derived by the programmer, after obtaining both SN_j and dID during the IDENTIFICATION session, and the *Emergency-Request* command is sent to IMD to disable it.

Table 1: Security comparisons.

	[8]	[4]	[6]	[9]	Proposed
Security model	2-Tier	2-Tier	2-Tier	3-Tier w/PD	3-Tier w/HAS
Cryptographic method for authentication	Password	CR with shared secret	CR with shared secret	CR with shared secret and signature	CR with shared secret
Authentication type	Unilateral	Unilateral	Mutual	Unilateral	Mutual
SK generation	None	None	Diffie-Hellman	PD generates SK	HAS generates SK
Key management	N/A	IMD-specific key from MK	N/A	PD maintains multiple PKs	RK and WK from LK
Multiple programmers	N/A	Insecure support	Secure support	Insecure support	Secure support
Patient data protection	N/A	N/A	N/A	N/A	Secure with Auth_j
Treatment protection	N/A	N/A	N/A	N/A	Possible with Auth_j and Sig_j

CR: challenge-response; PD: personal device; MK: master key; SK: session key; LK: long-term key; PK: public key; RK: Read-Key; WK: Write-Key.

6.8. Security Comparisons. In this section, various security features of the proposed protocol are compared with those in [4, 6, 8, 9]. Security comparisons are summarized in Table 1. The proposed one and the one in [9] are based on the 3-Tier security model. However, [9] employs a personal device with more power and computational resources than IMD. It works as a proxy for IMD and performs the authentication on its behalf.

Except [8], cryptographic methods for the authentication of IMD and programmer are challenge-response authentication based on a shared secret. In particular, the challenge-response authentication in [9] is based on a shared secret between IMD and personal device and digital signature between the personal device and programmer. The proposed one and those in [6, 9] generate session keys for confidentiality, integrity, and authentication. The session key between IMD and programmer is generated based on the Diffie-Hellman key exchange in [6], by the personal device in [9], and by HAS in the proposed one.

The key management scheme is employed in the proposed one and in [4, 9]. In [9], the personal device maintains the public keys of multiple programmers. The public key is used for the personal device to send the session key to the programmer and to verify the signature generated by the programmer. The IMD-specific keys, $K_{\text{dID}} = f(K_M, \text{dID})$, are derived from a master key, K_M, which is common to all the programmers in [4]. When receiving the IMD's identification number, dID, from the IMD, the programmer can share the IMD-specific key with IMD.

In the case that a programmer is stolen by an adversary, the security of IMD can be endangered. Since the master key and private key of the programmer are stored in the programmer in [4, 9], respectively; they are insecure. On the other hand, [6] employs the Diffie-Hellman key exchange to generate the session key, so that the stolen programmer does not induce the security problem of IMD. Likewise, the proposed one is also secure, since secret information is not stored in the programmer. The last two features, Patient Data Falsification and Medical Dispute Resolution, are provided only by the proposed one, which has been discussed in Sections 6.5 and 6.6, respectively.

The proposed protocol in Figure 4 has a similarity to the Kerberos protocol [21] in terms that session keys are shared between IMD and programmer through HAS. However, there are several differences between them, so that the Kerberos protocol cannot be applied directly to the proposed protocol. First, unlike the Kerberos protocol that is distributing only one session key, two distinct session keys (Read-Key, Write-Key) are employed in the proposed protocol in order to differentiate between *Read* permission and *Write* permission. Second, the session key is encrypted and transported to IMD from HAS when employing the Kerberos protocol, while only the keying materials to compute the session key are transported to IMD in the proposed protocol. Third, to prevent from the replay attack, the timestamps are used in the Kerberos protocol, while the session numbers are employed in the proposed protocol. Basically, the timestamp for freshness is not appropriate for the IMD-programmer environment since timestamp-based protocols require that time clocks be both synchronized and secured. Namely, a supplementary protocol for the time synchronization between IMD and programmer should be provided. Furthermore, it is not difficult to modify the local time clock of the programmers.

6.9. Performance Comparisons. Since each of [4, 6, 8, 9] is devoted to its own security feature that is not adequate for direct comparison, the proposed one and those in [6, 9] are compared in terms of computational complexity of the authentication/integrity, session key generation/distribution,

Table 2: Performance comparisons.

2009	[6]		[9] w/PD			Proposed w/HAS		
	IMD	Prog.	IMD	PD	Prog.	IMD	Prog.	HAS
Authentication/integrity	1 mac (UniL)	1 mac (UniL)	1 enc 2 dec	1 enc 1 ver	2 enc/1 dec 1 sig	2 mac	4 mac	2 mac
SK generation/distribution	1 DH w/RBE	1 DH w/RBE	1 dec	1 Penc 1 enc	1 Pdec	1 kdf	1 dec	1 kdf 1 enc
No. msg	$4 \cdot \lvert p \rvert$		IMP-PD: 1/PD-Prog.: 3 IMD-Prog.: 2			IMD-Prog.: 2/Prog.-HAS: 2		
Patient data protection	N/A		N/A			1 enc 1 hash	1 enc 1 dec	1 hash 1 dec
Treatment protection	N/A		N/A			N/A	1 sig	1 ver

PD: personal device; Prog.: programmer; No. msg: number of messages exchanged; mac: MAC function; UniL: unilateral authentication; RBE: rapid bit exchange; DH: Diffie-Hellman; hash: hash function; Penc: public-key encryption; Pdec: public-key decryption; enc: symmetric encryption; dec: symmetric decryption; kdf: key derivation function; sig: signature generation; ver: signature verification.

and the number of messages exchanged. Performance comparisons are summarized in Table 2.

In order to generate a session key, g^{xy} mod p, between IMD and programmer, [6] uses a DH (Diffie-Hellman) key exchange, where g^x mod p and g^y mod p are DH contributions generated by each of them and exchanged via rapid bit exchange. The rapid bit exchange (RBE) means that each bit of the DH contributions is transmitted on a bit-by-bit basis to estimate the distance between them. So, a total of $4 \cdot |p|$ messages, each of which consists of one bit, are exchanged, where $2 \cdot |p|$ are for the bit-by-bit exchanges of two additional nonces. Then, the session key is used for the programmer to authenticate itself to the IMD. Only one MAC (message authentication code) computation is required for unilateral authentication.

The proposed one and that in [9] are based on the 3-Tier security model with PD (personal device) and HAS (Hospital Authentication Server), respectively. In the case of [9], PD generates and encrypts a session key with the public key of the programmer and the shared key with IMD, respectively, and sends it to them. The session key is used to protect the command and response exchanged between IMD and the programmer. The programmer is unilaterally authenticated to PD, based on the signature, while IMD unilaterally authenticates PD, based on the preshared secret. A total of 6 messages are exchanged among IMD, PD, and programmer.

In the proposed one, there are two distinct sessions, the READ and WRITE sessions, for a patient session. Since they are identical in terms of computational complexity, only the READ session is considered for equivalent comparisons with [6, 9]. The (personalized) programmer exchanges two messages with HAS for the purpose of obtaining the Read-Key and exchanges another two messages with IMD to obtain the current patient data. A total of 4 messages are exchanged among IMD, programmer, and HAS. On the other hand, during the WRITE session, the signature is used for both the patient data protection and medical dispute resolution, which are not supported in [6, 9].

7. Conclusions

Recently, the privacy and security issues of IMDs have been widely recognized in the medical device market and research community, and a lot of researches have been carried out to address the privacy and security issues of IMDs. Most of them have concentrated on the authenticated key establishment between the IMD and programmer, as well as solving the tension between the security and the safety of IMDs in emergency situations. By introducing a new security architecture in this paper, we have addressed three security-related issues associated with the IMDs: support for multiple programmers, prevention of patient data falsification, and the physician's treatment protection. In particular, by introducing HAS, multiple programmers can be supported for the IMDs, without embedding the keying materials inside the programmers. Furthermore, even the physician cannot modify the patient data arbitrarily, and the physician's treatment history based on the observed patient data can be monitored securely.

Notations

dID:	IMD's device identification number
pID:	Physician's identification number
PatientInfo:	Patient information such as patient name, date-of-birth, and gender
SN_j:	The jth patient session number
$PatientData_j$:	Patient data collected from IMD during the jth patient session
$Conf_j$:	Physician's treatment to change parameters of IMD during the jth patient session
$MAC(K)$:	Message authentication code computed over all preceding fields of a command using symmetric key K; namely, $\{m, MAC(K) = h(K, m)\}$, where m is the command and $h(\cdot)$ is one-way hash function
$[m]K$:	Encryption of m using symmetric key K

$kdf(\cdot)$: Key derivation function
$h(\cdot)$: One-way hash function
K_{dID}: A long-term key shared between IMD and HAS
PK_{pID}, SK_{pID}: Physician's public key and private key for signature
K_{pID}: A long-term key shared between physician and HAS
Kr_j, Kw_j: Read-Key and Write-Key for the jth patient session

K_j: Session key derived from K_{pID} for the jth patient session
$Sig(SK_{pID})$: Signature computed over all preceding fields of a command using the physician's private key.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2012R1A1A2000677). This research was also supported by the MSIP (Ministry of Science, ICT and Future Planning), Republic of Korea, under the CPRC (Communications Policy Research Center) Support Program supervised by the KCA (Korea Communications Agency) (KCA-2013-003) and under the Employment Contract based Masters Degree Program for Information Security supervised by the KISA (Korea Internet Security Agency)(H2101-14-1001).

References

[1] U. Lakshmanadoss, A. Shah, and J. P. Daubert, "Telemonitoring of the pacemakers," in *Modern Pacemakers—Present and Future*, M. R. Das, Ed., pp. 129–146, InTech, 2011.

[2] D. Halperin, T. Kohno, T. S. Heydt-Benjamin, K. Fu, and W. H. Maisel, "Security and privacy for implantable medical devices," *IEEE Pervasive Computing*, vol. 7, no. 1, pp. 30–39, 2008.

[3] FDA, "FDA Safety Communication: Cybersecurity for Medical Devices and Hospital Networks," June 2013, http://www.fda.gov/MedicalDevices/Safety/AlertsandNotices/ucm356423.htm.

[4] D. Halperin, T. S. Heydt-Benjamin, B. Ransford et al., "Pacemakers and implantable cardiac defibrillators: software radio attacks and zero-power defenses," in *Proceedings of the IEEE Symposium on Security and Privacy (SP '08)*, pp. 129–142, Oakland, Calif, USA, May 2008.

[5] C. Li, A. Raghunathan, and N. K. Jha, "Hijacking an insulin pump: security attacks and defenses for a diabetes therapy system," in *Proceeding of the IEEE 13th International Conference on e-Health Networking, Applications and Services (HEALTHCOM '11)*, pp. 150–156, Columbia, MO, USA, June 2011.

[6] K. B. Rasmussen, C. Castelluccia, T. S. Heydt-Benjamin, and S. Capkun, "Proximity-based access control for implantable medical devices," in *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS '09)*, pp. 410–419, Chicago, Ill, USA, November 2009.

[7] R. A. Balczewski and K. Lent, "Security system for implantable medical devices," U.S. Patent 6,880,085, 2005.

[8] J. A. von Arx, A. T. Koshiol, and J. E. Bange, "Secure long-range telemetry for implantable medical device," U.S. Patent 7,155,290, 2006.

[9] F. Xu, Z. Qin, C. C. Tan, B. Wang, and Q. Li, "IMDGuard: securing implantable medical devices with the external wearable guardian," in *Proceedings of the 30th IEEE International Conference on Computer Communications*, pp. 1862–1870, Shanghai, China, April 2011.

[10] T. Denning, K. Fu, and T. Kohno, "Absence makes the heart grow fonder: new directions for implantable medical device security," in *Proceedings of the 3rd Conference on Hot Topics in Security*, pp. 1–7, Boston, Mass, USA, 2008.

[11] S. Gollakota, H. Hassanieh, B. Ransford, D. Katabi, and K. Fu, "They can hear your heartbeats: non-invasive security for implantable medical devices," *ACM Computer Communication Review*, vol. 41, no. 4, pp. 2–13, 2011.

[12] T. Denning, A. Borning, B. Friedman, B. T. Gill, T. Kohno, and W. H. Maisel, "Patients, pacemakers, and implantable defibrillators: human values and security for wireless implantable medical devices," in *Proceedings of the 28th Annual SIGCHI Conference on Human Factors in Computing Systems*, pp. 917–926, Atlanta, Ga, USA, April 2010.

[13] S. Schechter, "Security that is meant to be skin deep: using ultraviolet micropigmentation to store emergency-access keys for implantable medical devices," in *Proceedings of the 1st USENIX Workshop on Health Security and Privacy*, pp. 1–2, Washington, DC, USA, August 2010.

[14] Medtronic CareLink Network, https://world.medtroniccarelink.net/.

[15] Biotronik, Home Monitoring Service Center, http://www.biotronik.com/.

[16] D. Panescu, "Emerging technologies: wireless communication systems for implantable medical devices," *IEEE Engineering in Medicine and Biology Magazine*, vol. 27, no. 2, pp. 96–101, 2008.

[17] S. Brands and D. Chaum, "Distance-bounding protocols," in *Advances in Cryptology—EUROCRYPT '93: Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques Lofthus, Norway, May 23–27, 1993*, vol. 765 of *Lecture Notes in Computer Science*, pp. 344–359, Springer, New York, NY, USA, 1994.

[18] N. Sastry, U. Shankar, and D. Wagner, "Secure Verification of Location Claims," in *Proceedings of the ACM Workshop on Wireless Security*, pp. 1–10, San Diego, Calif, USA, September 2003.

[19] K. Malasri and L. Wang, "Securing wireless implantable devices for healthcare: ideas and challenges," *IEEE Communications Magazine*, vol. 47, no. 7, pp. 74–80, 2009.

[20] M. Burrow, M. Abadi, and R. Needham, "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, 1990.

[21] C. Neuman, T. Yu, S. Hartman, and K. Raeburn, "The Kerberos Network Authentication Service," RFC 4120, July 2005.