IMDfence: Architecting a Secure Protocol for Implantable Medical Devices

Muhammad Ali Siddiqi  
Department of Neuroscience  
Erasmus Medical Center  
Rotterdam, The Netherlands  
m.siddiqi@erasmusmc.nl

Christian Doerr  
Cyber Threat Intelligence Lab  
Hasso Plattner Institute  
Potsdam, Germany  
christian.doerr@hpi.de

Christos Strydis  
Department of Neuroscience  
Erasmus Medical Center  
Rotterdam, The Netherlands  
c.strydis@erasmusmc.nl

Abstract—Over the past decade, focus on the security and privacy aspects of implantable medical devices (IMDs) has intensified, driven by the multitude of cybersecurity vulnerabilities found in various existing devices. However, due to their strict computational, energy and physical constraints, conventional security protocols are not directly applicable to IMDs. Custom-tailored schemes have been proposed instead which, however, fail to cover the full spectrum of security features that modern IMDs and their ecosystems so critically require. In this paper we propose IMDfence, a security protocol for IMD ecosystems that provides a comprehensive yet practical security portfolio, which includes availability, non-repudiation, access control, emergency access, entity authentication, remote monitoring and system scalability. The performance of the security protocol as well as its feasibility and impact on modern IMDs are extensively analyzed and evaluated. We find that IMDfence achieves the above security requirements at a mere 4.64% increase in total IMD energy consumption, and less than 14 ms and 6 kB increase in system delay and memory footprint respectively.

Index Terms—Authentication protocol, battery-depletion attack, battery DoS, denial-of-service attack, IMD, implantable medical device, non-repudiation, smart card, zero-power defense

I. INTRODUCTION

Modern implantable medical devices (IMDs), such as cardiac pacemakers and defibrillators, neurostimulators, and more, are equipped with wireless connectivity in order to aid in treatment-related reconfiguration, patient-health monitoring, device testing etc. [1], [2]. However, wireless links have made IMDs susceptible to various attacks by malicious entities.

Earlier-generation IMDs had little or no security provisions whatsoever, as confirmed by numerous ethical-hacking incidents over the past decade [3]–[5]. The research community has responded with a wealth of new schemes and, eventually, top IMD manufacturers now claim to have rectified the security weaknesses over the past few years [6], [7]. However, due to the constraints imposed by an IMD’s scant computational, storage and energy resources, most proposed schemes in research have refrained from taking proven security approaches. Moreover, since these schemes have been specifically tailored for IMDs, they have missed the big picture and resulted in limited coverage of the security properties essential to a modern IMD. Specifically, most focus has been drawn on confidentiality, integrity, authentication and emergency access (e.g., [8]–[11] etc.), while availability, non-repudiation, access control and system scalability have been left unaddressed for the most part. In addition to the fact that these additional requirements are difficult to tackle, prior seminal work has not identified or stressed their importance.

In this paper, we debunk the myth that advanced security is impossible in modern IMDs. To this end, we collect both typical and overlooked security requirements and propose IMDfence, a novel, complete security protocol for an IMD ecosystem. We make the following contributions:

• We provide a comprehensive security protocol for IMD ecosystems, IMDfence, which addresses crucial, yet previously ignored requirements, i.e., availability, non-repudiation, access control, remote monitoring and system scalability.

• We propose a complete and realistic solution for accessing the IMD during emergencies without compromising security or patient safety.

• We provide an extensive evaluation of IMDfence, and pay special attention to the protection against battery denial-of-service (DoS) attacks.

The rest of the paper is organized as follows: We enumerate IMD-system requirements in Section II and then identify related works in Section III. Section IV details our proposed security protocol. We evaluate IMDfence in Section V and provide concluding remarks in Section VI.

II. IMD-SECURITY REQUIREMENTS

In this section, we collect and present the necessary security and related functional requirements that should be satisfied in modern IMD systems. These requirements form the basis of the IMD-specific security protocol, to be detailed in Section IV.

We consider an implant that is capable of communicating wirelessly with a reader/programme [1]. In order to evaluate the IMD-system security, we assume an attacker whose aim could be to either (1) modify or sabotage IMD operation in order to prevent patient treatment, (2) manipulate patient-related data, or (3) steal patient data. Furthermore, we assume

1 The term reader will be used for any device that is able to directly communicate with the implant.
that the attacker has full control of the wireless channel between the reader and IMD. This means that he/she can eavesdrop, modify, insert, block or replay messages between these two entities at will. As a result, the IMD-security system is required to provide certain security services and domain-specific requirements:

A. Basic security services (SR1)

Like in other domains, the IMD-security system should provide the fundamental security services: Confidentiality, Integrity and (message) Authentication, which are usually addressed through the use of lightweight block-ciphers and message-authentication codes (MAC) [12]. More specifically, the commands sent from the reader to the IMD and the associated responses (e.g., data logs) should be treated as confidential and it should be ensured that such data is not modified in transit.

B. Availability (SR2)

The IMD should always be available for patient treatment whenever required. This implies that the device should be protected against Denial-of-Service (DoS) attacks. One of the highest-likelihood and lowest-cost attacks is the battery-depletion attack (or battery DoS attack), as indicated in the IMD-specific threat-modeling analysis in [1] and practically demonstrated in [3], [4].

C. Non-repudiation (SR3)

Since there is always a possibility of malpractices, medical mistakes or insider attacks, we require non-repudiation to aid in computer forensics in case a patient experiences medical issues as a direct consequence of such actions. This security service ensures that a physician, paramedic or nurse is not able to repudiate his/her involvement in such scenarios. Non-repudiation has not been given detailed consideration by the research community when it comes to IMD systems. One of the reasons is that true non-repudiation can only be achieved through the use of public-key (or asymmetric) cryptography for computing digital signatures, which has traditionally been considered to be resource-costly for IMDs [12], [13]. Another, very important, reason is that past generations of IMDs could only be accessed by one person, i.e., the physician. Nowadays, the IMDs can be accessed by multiple people, including the patients themselves [14]–[16]. Hence, there is a need to introduce user accountability.

Most of the existing IMD-security works have looked into strict reader-IMD communication (without the involvement of a trusted third party). Even if we assume that the resource-constrained IMD is able to support public-key computations, this (reader-IMD) configuration makes it impossible for the IMD to effectively use public-key cryptography since it cannot keep track of the validity of the reader certificates (due to lack of Internet connectivity). What is more, these devices do not have sufficient memory to store the required certificates [17]. For instance, the IMD must store all possible reader certificates if we want to support access during travels or when the patient is in a foreign location. Hence, a scheme is required that employs the use of additional components (as will be discussed in Section IV) to solve these issues.

Another complication is the legal aspect. Since non-repudiation is there to provide evidence, it should be incorporated based on the assumption that such evidence will be scrutinized by a hostile legal expert [18]. One main limitation of cryptography-based non-repudiation is that there is no formally-verifiable link between the device that signs the digital signature and its user. For example, the user, i.e., the private-key owner, can falsely claim that the signature has been generated by a malware program without his/her consent, or that the private key has been stolen. There is no technical mechanism that can determine whether such a claim is false [19]. In this work, we propose to address this limitation, which we term as the Non-repudiation gap, at the policy level, i.e., as part of standard operating procedure.

D. Emergency Access (SR4)

Patient safety always outweighs device security. Hence, in emergencies the security protocol should not hinder or delay paramedic access to the IMD [2], [20]. Although it seems reasonable to drop security altogether in such situations, this can be a problem if, in a normal scenario, an adversary fools the IMD into entering the emergency-access mode. The security protocol must be capable of allowing the IMD to accurately classify whether a communication attempt is an emergency access or normal access. This ensures that the adversary is unable to trigger and exploit the emergency-access mode. Furthermore, since there is a high likelihood of the patient losing control of his/her actions in emergencies, the emergency-access mode should be independent of patient participation.

E. Multi-manufacturer environment (SR5)

In emergencies, it is unlikely for the paramedic to know the IMD model and manufacturer name beforehand. Moreover, it is not possible to preemptively stock all the readers from all the manufacturers in the ambulance. Hence, the IMD-security system should be vendor-independent, i.e., all the vendors need to agree on a unified standard for secure reader-IMD communication. This way an ambulance can use one generic reader regardless of the IMD vendor and type. It follows that an emergency-access scheme should be adoptable by all IMD types. E.g., an emergency-access solution that requires an IMD measuring the cardiac signal, can be easily incorporated in pacemakers, but it will require significant modifications in neurostimulators.

F. Access control (SR6)

The access privileges of the reader user should be differentiated based on the type of user. For example, nurses, patients or patient relatives may only be allowed to read status data from the implant, whereas a physician and a paramedic may also be allowed to modify the implant configuration for therapy updates or suspend or resume its operation. Similarly,
a technician may be allowed to modify the implant firmware in addition to tasks of the above roles.

**G. User and reader-IMD Authentication (SR7)**

In order to aid in non-repudiation and access control, the IMD system should be able to identify the physician, nurse, paramedic etc. who is using the reader to communicate with the implant. Conversely, the reader should also be able to authenticate the IMD in order to prevent spoofing attacks on the reader. Hence, there is a requirement of performing mutual authentication instead of just authenticating the reader unilaterally \[12]\). Furthermore, said authentication is required to be strong, i.e., it should imply both message and entity authentication, and guarantee message freshness, or in other words replay protection.

**H. Flexibility and Scalability (SR8)**

The IMD should not be limited to communicating with only a fixed amount of readers since this severely limits portability, e.g., during emergencies when a paramedic reader is used, or when there is a need for treatment at some hospital during travels. Hence, there should not be any pre-shared secrets between the reader and IMD.

**I. Bedside-reader operation for remote monitoring (SR9)**

Some of the modern IMD systems also include a bedside reader, which enables remote monitoring \[21\]. It establishes communication with the IMD when the patient is asleep and sends treatment status to a back-end server via an Internet connection. However, this additional connection represents an increase in the attack surface, which imposes additional security requirements. We predict that the use of such readers will become more widespread over time due to their time- and cost-saving features. Hence, this phenomenon should proactively be considered when designing secure IMD systems.

**III. EXISTING SYSTEMS AND RELATED WORK**

IMD manufacturers have typically relied on “security through obscurity”, in which they choose to hide the communication-protocol specifications in order to enhance security. This is not a recommended practice, and as a consequence of using this approach, we have seen several successful blackbox-hacking attempts over the past few years \[3\], \[4\].

Most commercial IMDs communicate with the physician’s programmer through an Inductive Coupling channel. In addition, some of the latest devices, including neurostimulators \[16\], insertable cardiac monitors \[15\] and even pacemakers \[14\] offer a Bluetooth Smart connection between the patient smart-phone and the implant. The initial pairing between these devices is based on the Bluetooth Smart standard in addition to proprietary protocols \[16\]. However, they do not disclose the association models used in these pairings, which makes these devices vulnerable to attacks due to the reasons mentioned above. In most of the cardiac devices, in the absence of an IMD-programmer, a magnet can be used to disable therapy or to switch to a non-programmed behavior \[22\]. This mode, however, can be easily exploited by adversaries through the use of a strong magnet when in close proximity to the patient (e.g., in public transport).

From the perspective of the research community, we observe a steep rise in the number of works proposed over the last few years \[23\]. For data confidentiality, integrity and message authentication, the use of lightweight primitives has been proposed. Early works focused on basic security protocols based on symmetric ciphers, which rely on a common pre-shared key between the reader and the IMD \[12\]. However, such approaches are not scalable in terms of adding new readers that can access the implant. They also do not allow paramedic access during emergencies. Therefore, most of the existing works pertain to the domain-specific requirement of emergency access, in addition to entity authentication and key exchange. For entity authentication, these works rely on the touch-to-access policy, which ensures that only the entities that can physically touch the patient for a prolonged period of time are allowed access to the implant \[1, 20\]. In other words, it is infeasible for an attacker to get in close proximity to the patient, and even if that is the case, the patient would detect this and reject physical contact. Also, the attacker would then have far easier methods to harm the patient than via accessing the implant, e.g., by physically attacking the patient. These works can be broadly categorized as follows \[3\]:

**a) Biometric-based:** These approaches (such as \[20\], \[24\]) rely on both the reader and IMD to measure a physiological signal from different parts of the patient’s body. The devices are paired based on the similarity of these measurements.

**b) Proxy-based:** These works propose to use an additional device in the possession of the patient, such as a smart phone, watch, etc \[25\], \[26\]. The device is paired with the IMD and is used to authenticate the reader that is trying to communicate with the implant. In case of emergency, the device can be physically distanced from the patient in order to grant the reader unsecured access to the IMD.

**c) Distance-based:** These works (e.g., \[27\], \[28\]) employ weak or out-of-band (OOB) signals for reader-IMD communication. These can either involve direct transfer of a session key, which would be hard for an attacker to eavesdrop, or they can require the devices to mutually prove proximity to one another.

**d) Token-based:** This is the simplest approach, which relies on the patients having the IMD-access key or password with them, which is stored e.g., on a bracelet. During an emergency, a paramedic can access the IMD using this token.

We now present a brief overview of the latest works from literature that were specifically tailored for IMDs.

Bu et al. propose a low-energy IMD-security scheme called Bulwark \[8\], which, in addition to providing basic security services (SR1), also allows IMD access in emergencies (SR4). This emergency access scheme is based on Shamir’s secret sharing, which relies on the users (including the paramedics) to register with the manufacturer of the specific IMD in advance in order to retrieve the access key in case of an
emergency. As evident, such a requirement inhibits IMD access in case the patient is out of town (SR8).

Chi et al. [10] propose a protocol that relies on the patient’s smartphone for the reader access. However, requiring the patient to be in possession of this additional device (i.e., the smartphone) all the time, including during emergencies, puts a significant burden on the patient.

Belkhouja et al. [11] propose a symmetric crypto system in which they use a Chaotic key generator that is employed by both the reader and IMD to generate the symmetric key. However, in order for this key generator to work, both entities are required to have similar pre-installed initial conditions/values. Hence, this scheme cannot function in an emergency scenario, or when the patient is traveling, since the IMD and the reader will not be sharing the same initial conditions.

Wazid et al. [29] and Mao et al. [30] propose three-factor protocols, which rely on passwords, smart cards, and biometrics. However, their protocols rely on an offline-user-registration phase before an actual session can take place, which inhibits SR4 and SR8. Rathore et al. [31] propose a scheme in which the identifiers of each user (including the patient) are derived from their cardiac signals and are stored in the implant. Hence, it requires a user-registration phase similar to the above protocols. However, their scheme allows emergency access since the paramedic can measure the patient’s cardiac signal, which is compared by the IMD against the stored identifier in order to grant access. The three-factor protocol from Fu et al. [32] also provides emergency access. However, the patient is required to always be in possession of a personal smart card so that the paramedic is able to use it during an emergency.

In addition, quite a few authentication and emergency-access schemes have been proposed recently that rely on static biometrics (such as fingerprints) [33], dynamic biometrics (such as cardiac signals) [34] and combination of both [9]. The interested reader can refer to [23, 35–38] to get an overview of prior works in this area.

Overall, the above works address only parts of the IMD security requirements (SR1, SR2, SR4 and SR7). For instance, non-repudiation is not considered and the emergency-access schemes do not take into account the (current) multi-manufacturer environment, as discussed in Section II. To the best of our knowledge, there is no protocol that provides all the services highlighted in Section II. As will be seen, IMDfence necessarily addresses the complete IMD ecosystem.

| Notation | Definition |
|----------|------------|
| ID_A     | ID of entity A |
| N_A      | Nonce generated by A |
| K_{pubA}/K_{prA} | Public/private key pair of A |
| K_{AB}   | Pre-shared symmetric key between A and B |
| K'_{AB}  | Short-term symmetric key between A and B |
| K_A      | A secret only known to A |
| P_A      | Privilege information of user/card A |
| CMD      | Command configuration |
| ANS      | Answer to CMD |
| k        | Difficulty of solving a client puzzle |
| x < i >  | i^{th} bit of a bitstring x |
| x < i : j > | Bit sequence x < i >,..., x < j > |
| t        | Time stamp |
| T        | Lifetime of reader-card authentication |
| NAC      | Number of allowed user-authentication cycles |
| {\{K_{AB}\}} | Authenticated encryption using K_{AB} |
| MAC_{K_{AB}}(\cdot) | MAC operation using key K_{AB} |
| sig_{K_{prA}}(\cdot) | Signature (of a hashed message) using K_{prA} |
| Cert_A   | Certificate consisting of ID_A, P_A, K_{pubA} and sig_{K_{prC}}(ID_A, P_A, K_{pubA}) |

* Such as Encrypt-then-MAC (EtM). Separate keys should be used for the encryption and MAC operations to prevent certain attacks and to ease key management [19]. However, these keys are not differentiated here for simplicity.

IV. IMDFENCE: SECURITY PROTOCOL FOR IMD ECOSYSTEMS

The absence of a complete security solution for IMD systems has led us to propose IMDfence, a novel secure-communication protocol that satisfies the requirements enumerated in Section II. As will be seen, IMDfence necessarily addresses the complete IMD ecosystem.
configuration helps in making $R$ independent of the need to pre-share keys with the hospital, which aids in scalability. As a result, a patient can use his/her personal reader from any location, and/or buy a new reader from the manufacturer without the need of registering it first at the hospital.

In our proposed configuration, each smart card also has its own public/private key pair. Technically, $R$ has the capability to maintain a comprehensive certificate-revocation list (CRL) of smart cards due to frequent Internet connectivity. Hence, it is able to verify smart cart certificates. On the other hand, due to the limited on-board memory and less-frequent Internet connectivity, $C$ can only maintain a small CRL that does not change frequently. Hence, $C$ can not verify the authenticity of the multitude of reader certificates. As a result, public-key-based key exchange cannot be used to establish a session key between $R \leftrightarrow C$. However, it will be shown in Section IV-C that the session key between $R \leftrightarrow C$ will be established using $S$ as a TTP. The same will be done for establishing session key between $R \leftrightarrow I$. Lastly, no session key is required between $C \leftrightarrow I$.

2) Centralization and Public-key infrastructure: The public keys of $S$, $R$ and $C$ are signed by a trusted certification authority (CA). The smart-card certificates, in addition, also include the user privileges.

We consider the precise implementation details of public-key infrastructure (PKI) and certificate revocation outside the scope of this paper. In case of a smart card, certificate revocation would be needed when a card is stolen, a user leaves, or if he/she changes roles (e.g., from nurse to paramedic). For a reader, certificate revocation would be required in case $R$ is stolen or deemed as out-of-service. The server is given the responsibility to verify the certificates of $R$ and $C$ and hence, it is assumed that it maintains an up-to-date CRL.

3) Modes of operation: We propose two modes of operating in IMDfence, one for regular (online) operation and the other in the absence of an active Internet connection (offline), e.g., during emergencies (SR4); see Fig. 2. Online mode offers the full security- and functional-requirement portfolio highlighted in Section III whereas offline mode results in the graceful degradation of offered services without compromising security and patient safety. Since $S$ is not available in offline mode, $R$ and $I$ will be required to undergo an out-of-band (OOB) pairing phase in order to securely exchange a short-term session key. These modes and the constituent phases will be elaborated in the coming sections.

B. Threat model

As discussed in Section III, we assume an attacker $A$ that has full control of the wireless channel between the reader and IMD. $R$ is assumed to be untrustworthy by $I$, $C$ and $S$, and vice versa. Moreover, we assume that if $A$ steals a personal smart card or a valid reader, then the user or hospital staff should notify the hospital server so that it is blacklisted. Additionally, we assume that $A$ can hack the reader to read out or modify data at the interface of the inserted smart card. However, $A$ does not have access to the keys stored in $R$ and

---

2The routing details of the messages communicated via the reader have been omitted for brevity.
C. This implies that protection against side-channel attacks is considered outside the scope of this work since such attacks are typically addressed through specialized countermeasures. Nonetheless, it will be shown in Section V-A2 that such attacks will not be of any use to the attacker if \( R \) and/or \( C \) are reported as stolen. We also assume that the hospital personal do not have access to the keys stored in the server since such attacks can be prevented by employing standard practices, such as using hardware security module (HSM) etc.

C. Regular (online) mode

The regular mode of IMDfence is shown in Figures 3 to 6. It starts with the \( R \leftrightarrow C \) mutual authentication phase after the physician (or any other user) inserts his/her smart card into the reader.

1) \( R \leftrightarrow C \) mutual authentication: In this phase, \( R \) first tries to establish a secure connection with \( S \) by sending its identifier and a nonce (which is a freshly generated number that is used only once). In order to deter distributed-denial-of-service (DDoS) attacks against \( S \) (to ensure server availability (SR2)), a basic client-puzzle protocol (CPP) is employed [40]. CPP is a proof-of-work system in which any client (or in this case a reader) that wants to access the server (during high load) is required to correctly solve a cryptographic puzzle. For a regular client the costs of solving this puzzle are negligible. However, in order to launch a successful DDoS by initiating a large number of simultaneous connections, it would be computationally infeasible for the attacker to solve a multitude of such puzzles.

\( S \) initiates CPP if it senses a DDoS attack or it is dealing with an abnormally high number of simultaneous connections. It first calculates \( x \), which is the \( n \)-bit hash of \( ID_R \), the current time stamp \( t \) and its long-term secret \( K_S \). It then computes a second hash \( h(x) \). \( S \) sends \( h(x) \) and \( x \) excluding the first \( k \) bits of \( x \), along with the \( t \). \( R \) computes the solution, i.e., the missing \( k \) bits of \( x \), and sends it along with \( ID_R \) and the received time stamp. \( k \) represents the difficulty of solving the puzzle. \( S \) calculates \( x \) again and verifies that the solution indeed corresponds to the missing bits. It also verifies, with the help of \( t \), that the puzzle has not expired. \( S \) is protected against memory exhaustion since it is not required to store any data for the verification of the puzzle solution. In case these checks are successful, \( S \) sends its nonce to \( R \).

\( R \) then performs a Diffie–Hellman (DH)-based handshake with \( S \) in which a session key is established between them based on their public/private key pairs (see Fig. 3). During this handshake, both verify each other’s certificates and, additionally, \( S \) checks if \( R \) is valid (i.e., it is not reported as stolen or out-of-service).

In order to achieve authentication between \( R \) and \( C \), \( R \) then initiates a five-pass, mutual-authentication protocol borrowed from the ISO/IEC 9798-2 standard [41], with \( S \) acting as a TTP (see Fig. 3). \( R \) generates its nonce and sends it along with its identifier and \( N_S \) to \( C \). \( C \) responds by sending \( N_C \) and sending a cryptogram \( m_{SC_1} \) that includes authenticated encryption of its certificate, \( ID_R \) and nonces, along with \( ID_C \) and \( N_C \) in plaintext. This cryptogram is calculated using \( K_{SC} \) since it is intended for the server. \( R \) stores \( ID_C \) and \( N_C \), and forwards the cryptogram to the server, which establishes that it originated from \( C \) and that it is also tied to \( R \). The server then checks the validity of \( C \), in case it has been reported stolen or has expired. It then calculates tokens for both these entities using the respective symmetric keys. These tokens include the nonces and identifiers of \( R \) and \( C \) and a
fresh symmetric key $K''_{RC}$. Additionally, $token_R$ and $token_C$ also contain $T$ (reader-card-authentication lifetime) and $NAC$ (number of allowed user-authentication cycles), respectively. Based on these tokens, $R$ and $C$ can be sure about each other’s trustworthiness.

$R$ decrypts $token_R$, retrieves $K''_{RC}$, calculates the MAC of the nonces, and forwards it along with $token_C$ to $R$. The smart card similarly decrypts $token_C$ and verifies the received MAC using $K''_{RC}$. $C$ sends a MAC that is calculated over $N_R$ and $N_C$ (including an addition by 1 to protect against replay of the previous message). $R$ verifies the received MAC using $K''_{RC}$. At this point, both $R$ and $C$ have mutually authenticated each other.

$R$ then sets its internal real-time clock to $T$ and starts it to track the period over which the subsequent phases can execute without the need of reader-card authentication. Since it is possible that $R$ is not connected to the Internet during its operation (e.g., in emergencies), this scheme enforces that $R$, by design, shall only be usable for a certain duration until it has first established an Internet connection. This makes sure that $R$ receives critical firmware updates in time, if there are any. This will be discussed further in Sections IV-D and V-A4.

2) User authentication: This phase is shown in Fig. 4, and its objective is to authenticate the card holder. Both $R$ and $C$ first recalculate and exchange the MACs of their nonces (including an addition by 2 and 3, respectively, to protect against replays) in order to verify that this phase is linked to the tokens acquired in the previous phase. This MAC re-exchange is introduced for the offline mode, to be discussed in Section IV-D. $R$ then checks its internal real-time clock to verify the validity of its token.

Then, the physician enters his/her PIN using a keypad on the reader. $R$ encrypts the PIN and the nonces (in order to prevent replays) using $K''_{RC}$. $C$ decrypts the message using the same key, verifies the PIN by comparing it with the stored one and sends back a cryptogram intended for the server, which is encrypted with $K_{SC}$. It contains the confirmation of success in addition to the nonces.

A user can access the reader $NAC$ times after which $token_C$ is considered expired and the card is required to perform a new reader-card authentication. A smart card can keep track of the number of accesses by storing the count in the flash memory of its MCU. It can be argued to have $token_C$ expired after a certain time duration instead of keeping count of the number of user-authentication cycles since it seemingly violates message freshness. The stumbling block is the fact that smart cards are not battery powered, and hence, they do not have a continuously running clock that can keep track of time. However, it will be shown in Section V-A2 that stealing a reader with a valid token will not give $A$ any advantage because of the additional protection provided by the touch-to-access principle.

3) Session-key ($K'_{RI}$) establishment: $R$ then initiates a TTP-based key established protocol with $S$ and $I$ in order to acquire a symmetric session key $K'_{RI}$ for providing basic services (SRI), as shown in Fig. 5. $R$ first exchanges the nonces and identifiers with $I$ and then sends the nonces and identifiers of all parties to $S$ along with $m_{SC2}$. $S$ first verifies $m_{SC2}$, and then determines the required privileges ($P_C$) for the particular user (e.g., physician, paramedic, nurse etc) from $Cert_C$. It generates $K'_{RI}$, encrypts it in two independent messages $m_R$ and $m_I$ intended for $R$ and $I$ respectively, and then sends these to $R$. $R$ decrypts $m_R$ and verifies its contents. It then encrypts $N_R$ and $N_I$ using $K'_{RI}$ (to form $m_{RI}$) and then sends it along with $m_I$ to $I$. $I$ first verifies $K'_{RI}$ by decrypting $m_I$, and then decrypts $m_{RI}$ to verify that $R$ has the knowledge of $K'_{RI}$ and that the nonces are valid. $I$ finally creates a MAC using the new session key for $R$ to
validate. At the end of this protocol, both $R$ and $I$ are mutually authenticated (SR7) and have arrived at a fresh session key in addition to performing key confirmation. This session-key-establishment phase is based on the Kerberos protocol, which employs time-stamps and life-time values to ensure freshness. These values are not employed in our case because of the associated complexity, e.g., the need for clock synchronization between $R$ and $I$ etc. [42]. To ensure message freshness, $I$ additionally sends $N_I$ to $R$ in step 2, and then verifies the existence of $N_I$ in the subsequent messages.

To protect against battery-DoS attacks (which impact availability (SR2)), steps 1 to 4 of the session-key establishment should be as lightweight as possible so that the IMD is able to execute it using harvested RF energy. This will be further discussed in Section V-B.

4) Main phase: After the establishment of session key, $R$ allows the user to enter a command on the reader interface (see Fig. 6). The command is encrypted along with the nonces (to prevent replay attacks) using $K'_{RC}$, and is sent to $C$. The card decrypts the command, digitally signs the message using $K_{prC}$ (to form $sig$) and sends it to $R$. $R$ re-encrypts the command using $K'_{RI}$ and sends it to the implant along with $sig$.

$I$ decrypts the command and verifies if it corresponds to the privileges information received in $m_1$ during the previous phase, hence ensuring access control. $sig$ is stored by the IMD next to $ID_C$, $NC$ and $NR$, which were stored during session-key establishment. This is required to ensure non-repudiation since $sig$ was signed using a personal private key. For example, in the case of a medical mistake (e.g., an incorrect command) that led to patient death, the physician will not be able to deny his/her involvement since this signature can always be retrieved from the IMD and subsequently verified using the associated data. Since the implant trusts the reader at this point, there is no need for $I$ to verify the signature since the associated MAC has already been verified by $R$. This relieves $I$ of the need to employ public-key cryptography and to track user certificates. After processing the command, the implant responds with an answer message encrypted with $K_{RI}$. $R$ displays it on its screen for the convenience of the user. The session keys expire after a finish command and its associated response, or after a period $T$.

5) Addressing the non-repudiation gap: As discussed in Section [11] the use of signature alone is not sufficient to address the legal aspects of non-repudiation. In order to bridge the non-repudiation gap, one option could be to enforce that the user protects $C$ and the associated PIN, or immediately reports in case it is lost. However, due to the possibility of human error in general, this is too much of a legal responsibility for the user.

The realistic way of bridging this gap is by introducing additional checks in the implementation of reader-card-authentication and session-key-establishment phases (see Figures 3 and 5). The server can ensure that the implant write access is requested from within the hospital network and during the working hours of the user. On the other hand, the server can allow read-only accesses from external networks, e.g., in case the access is made by a patient’s bedside reader. The user just has to ensure that $R$ is issued from a certified repository, and that $R$ should only be connected to a trusted Ethernet network (i.e., in a hospital or patient home). With these precautions, which a responsible user can easily follow, protection can be ensured against the malicious replacement of a command by a compromised reader, or against an attacker sending a malicious command him/herself in order to frame the said user. Due to the above risk-based multi-factor authentication, a user cannot falsely deny his/her involvement in a certain implant access because the alternative explanation implies that (1) the attacker stole a valid reader, card and pin, (2) accessed the implant from within the hospital and during the user’s working hours, and (3) $R$ and $C$ were not reported as stolen. The likelihood of all of this happening at once is extremely small, or, in other words, the non-repudiation gap is effectively bridged by the introduction of above checks.

6) Bedside-reader operation: The online mode also facilitates bedside-reader operation (see Fig. 1). Here, only the DH-based handshake between bedside $R$ and $S$ (from reader-card authentication phase), the session-key-establishment phase, and the main phase (with a few differences, as indicated in Fig. 6 and Fig. 5, respectively) need to be executed, since the commands and responses are only sent and read by $S$. Moreover, since the remote monitoring done in practice is only read-only, i.e., with the lowest access privileges, there is no need for non-repudiation if the read-only access control is implemented correctly. This can be done if $sig$ in step 6 is replaced by MAC of $CMD$ from $S$ (i.e., $MAC_{K_{SI}}(CMD, NR, N_I)$). Using this MAC, $I$ is able to verify that the command came from the server, and hence, it can be executed with read-only privileges. Finally, the hospital staff can retrieve the critical treatment data by logging into $S$. It can be argued that this remote-access mode should support
read/write access instead of just read-only in order to enable remote firmware updates. However, we stress that such updates should always occur in the presence of a qualified professional. This is important in case patient health suddenly deteriorates due to the update process. Moreover, in practice it is quite common and acceptable to get the IMD firmware updated at the clinic in the presence of a physician [7]. This mode is also useful for securely retrieving the stored signatures pertaining to previous programming sessions in order to free up limited IMD memory.

7) IMD access from a non-local location: In Section IV-A, we discussed that the smart card and the implant are registered at the local hospital, or in other words, they share their respective symmetric keys with the hospital server. During travels or when the patient is out of town, a situation may arise that requires access to the IMD for status monitoring or treatment updates at a non-local hospital. In this situation, the above scheme (see Fig. 1) will not work straightforwardly since the IMD is not registered at this hospital. Hence minor extensions are required (see Fig. 7), in which the non-local (remote) server $S_R$ establishes a secure connection with the local hospital $S_L$ via an IMD-vendor server $S_V$. $S_V$ maintains a list of all the IMDs in service and the hospitals at which they are registered. Based on $ID_L$ sent by $R$ to $S_R$ (and then $S_R$ to $S_V$) during the session-key establishment phase (see Fig. 5), $S_V$ determines $S_L$ and establishes a secure connection with it. $S_R$ sends $K_{RL}$, the relevant identifiers, nonces and $P_C$ to $S_L$ (via $S_V$) so that $S_L$ is able to construct $m_I$ and send it back to $S_R$. The protocol then proceeds normally and the IMD eventually retrieves $K'_{RI}$ after decrypting $m_I$.

D. Offline mode

In the absence of an active Internet connection and hence, the TTP ($S$), e.g., during emergencies, $R$ and $I$ need to establish a temporary shared key so that they can communicate directly in a secure manner. We propose to employ an OOB-channel-based key exchange while using the principle of touch-to-access (as discussed in Section III). This principle is employed by $I$ to establish trust with $R$ since we assume $R$ to be untrustworthy from the perspective of the IMD. We propose to use galvanic coupling for the OOB channel (between $R$ and $I$) since it results in virtually zero electromagnetic leakage compared to other coupling methods, such as capacitive coupling [43]. Moreover, it has advantage over biometric-based touch-to-access mechanisms (mentioned in Section III) because it does not require any initial RF communication messages before the IMD is sure that the external entity is in close proximity. This provides an additional security layer, which is critical for the pre-deployment configuration that will be discussed Section IV-D.

The paramedic places the galvanic-coupling interface of the reader on the patient skin3 at a point that is nearest to the IMD. The patient is assumed to thwart advances of a stranger trying to place a reader on his/her skin, if there is no emergency or a need for treatment. Hence, the implant assumes that the message received from the OOB interface is from a trustworthy source. In other words, in offline mode, the IMD-system security hinges on this OOB pairing.

The protocol is shown in Fig. 8. The paramedic is required to perform reader-card authentication when starting his/her duty, so that both $R$ and $C$ obtain their respective tokens

3Touching the skin is mandatory for the galvanic channel to function.
from $S$. When IMD access is required in an offline setting, $R$ first initiates user authentication with the paramedic smart card in the same way as in the regular mode. Similar to the online mode, $R$ verifies that its internal real-time-clock value is less than $T$. Moreover, the paramedic is allowed $NAC$ authentication cycles (or attempts) before the card requires a new token. Through the OOB channel, $R$ sends a request for offline access along with its identifier. Upon receiving this request, the implant assumes that this is an offline scenario since this channel is activated only in such extraordinary circumstances. As a result, it generates a random key $K'_{RI}$ and its nonce and sends them along with $ID_I$ to the reader using the same channel.

$R$, then, initiates session-key confirmation with $I$ in which both entities verify each other’s MACs that are generated using $K'_{RI}$. In order to update or inquire about the implant operation, the paramedic enters the command on the reader interface, which is encrypted using $K'_{RC}$ and is sent to $C$. The card digitally signs this command and sends it back to $R$. $R$ encrypts the command using $K'_{RI}$, calculates its MAC and sends it to $I$ along with $\text{sig}_{K'_{RC}}(CMD, N_R, N_C)$. This signature is stored by the IMD and is required to ensure non-repudiation, as discussed in Section IV-C. The IMD responds with an answer encrypted by the same session key, which is subsequently displayed on the reader display. The session key expires in a manner similar to that in the regular mode.

In offline mode, the user is allowed paramedic-level privileges, which has less access rights compared to a technician (see Section II). The use of the OOB channel in the beginning of the protocol makes it straightforward for the implant to conclude on granting only paramedic-role commands.

1) **Offline access with/without non-repudiation and access control**: We also propose a second flavor of the offline mode in which non-repudiation and user authentication are not a requirement. This is suitable for less critical implants, such as neurostimulators. This flavor does not require a smart card, and as a result we do not require the reader-card- and user-authentication phases in addition to signature generation. From the perspective of usability, this helps in removing some of the responsibilities from the paramedic. In this scheme, the touch-to-access principle is deemed to be sufficient in order to ensure trust establishment. It is important to note that for IMDfence, supporting non-repudiation during offline mode must be decided before the IMD-system deployment, so as to avoid exploitation.

2) **Offline access with/without reader-interface standardization**: As indicated in Section II, supporting emergency access in the field requires a standardized reader interface, which demands collaboration between major IMD manufacturers. In order to facilitate this **multi-manufacturer environment (SR5)**, there has to be one agreed-upon root CA that grants certificates to the vendors, who can then act as intermediate CAs that sign public keys of $S$, $R$ and $C$. As things stand, however, **real** emergency access does not exist in commercial IMDs. As long as this remains an open issue, the above standardization is not required, and as a result, IMDfence can be simplified by eliminating the need for a global root CA. Emergency-access support in IMDfence is intended to be there in anticipation of any future changes in this regard.

### E. Summary of protocol configurations

The different configurations of IMDfence are highlighted in Fig. 9. The dotted boxes indicate (fixed) pre-deployment configurations, which cannot be changed at run-time. Such configurations were discussed in Sections IV-D1 and IV-D2.

IMDfence is designed in such a way that an attacker cannot target one mode over another for exploitation. For instance, the offline mode is only triggered after an OOB access, which is protected by the touch-to-access principle. Moreover, the sub-modes of online access only come about by disabling certain IMDfence steps instead of switching to a totally independent behavior.

## V. Evaluation

In this section, we evaluate our system in terms of security feasibility and also look into the handling of battery-DoS protection for IMDs.

A. **Security analysis**

1) **Automatic validation using AVISPA tool**: For the automated and formal validation of IMDfence, we used AVISPA (Automated Validation of Internet Security Protocols and Applications) [44]. Any protocol to be validated using this tool is specified using the High-Level Protocol Specification Language (HLPSL). An HLPSL specification consists of a description of the **principals** (i.e., $R$, $I$, $C$, $S$ and the user in our case), security **goals** of the protocol, and the details of the session(s) to be analyzed. AVISPA integrates four back-end engines that provide different types of automatic analysis of an HLPSL specification [44]. The tool helps in detecting vulnerabilities against Man-in-the-middle and replay attacks. It also detects if the HLPSL specification is executable, i.e., all the specified protocol states are traversable. Using AVISPA, we can also optimize our protocols by removing certain parameters from the messages in order to reduce communication overhead and analyze if this results in a new vulnerability.

The analysis of IMDfence using AVISPA is summarized in Table III. The protocol handshake-specific requirements (SR1, SR3, SR6 and SR7) are satisfied by specifying the appropriate goals. In phase III, $S$ extracts user privileges from $Cert_C$ after
successful authentication of $C$ based on $N_S$ in $m_{SC2}$. $I$ then verifies $S$ based on $N_I$ to complete the chain from the card to the implant in order to ensure access control. In order to ensure non-repudiation, the server retrieves $\text{sig}$ from the IMD via the bedside reader, and verifies that it originated from $C$ during the session corresponding to $N_S$.

2) Attack scenarios: When considering all possible attack scenarios, we define the following reader types:

1) Valid $R$ ($R_{\text{valid}}$): This is a legitimate device, which is not reported as stolen.
2) Stolen $R$ ($R_{\text{stolen}}$): A legitimate device which is reported as stolen.
3) Hacked $R$ ($R_{\text{hacked}}$): A stolen reader which is also modified by $A$ in order to e.g., replace the signature or $\text{CMD}$.
4) Forged $R$ ($R_{\text{forged}}$): A custom-built or software-defined radio used by $A$ in order to communicate with an implant. This reader does not have any pre-shared keys with $S$.

The following scenarios are possible in terms of user-reader combinations:

1) Trusted (malicious/non-malicious) user & $R_{\text{valid}}$: This is the most common scenario, which must be handled by IMDfence. This implies that an insider attack (from a legitimate, malicious user) should be detected by the non-repudiation check.
2) Any user or attacker & $R_{\text{stolen}}$: No individual will be able to use a reader that is reported as stolen because of the checks involved in the reader-card-authentication phase.
3) Trusted user & $R_{\text{hacked}}$/$R_{\text{forged}}$: As a guideline, the user needs to make sure that $R$ is issued from a trusted repository, which rules out the use of $R_{\text{hacked}}$ and $R_{\text{forged}}$.
4) Trusted malicious user & $R_{\text{hacked}}$: A legitimate malicious user can cover his/her tracks by employing a hacked reader that can replace the signature correspond-

5) Trusted malicious user & $R_{\text{forged}}$: Such a user cannot send commands using a forged reader in an online case since $R_{\text{forged}}$ does not share a key with $S$. In the offline case, however, such a user can use a forged reader that is able to create a bogus $\text{sig}'$ and hence does not require any involvement of $C$. Moreover, he/she can use the OOB-pairing interface because of being considered as trusted by the patient. Similar to (4), such a scenario also requires hiring an advanced attacker to develop such a reader, and based on the touch-to-access assumption, the user has significantly easier methods to harm the patient.

6) Attacker & $R_{\text{valid}}$: For online access, the communication protocol will break if $A$ gets hold of a valid reader, card and its associated PIN, accesses the implant from within the hospital and during the user’s working hours, and $C$ is not reported as stolen. It is recommended that the user protects his/her card and PIN, or immediately reports it in case it is lost. Moreover, as a guideline, the user should never lend or sell $R$ to a third party. The protocol will also break if $A$ gets hold of an OOB-paired reader and a card with valid respective tokens, and has the knowledge of the PIN. We assume that the paramedic resets the pairing after treatment.

7) Attacker & $R_{\text{hacked}}$/$R_{\text{forged}}$: For online access, the attacker will not be able to use $R_{\text{hacked}}$ because of the reasons mentioned in (6) above. Similarly, he/she cannot use $R_{\text{forged}}$ since it does not have a shared key with $S$. Moreover, for an offline scenario, getting hold of these readers will not help an attacker $A$ since the main symmetric key ($K_{\text{pair}}$) comes from $I$ in the OOB pairing process. Hence, to gain advantage using these readers, $A$ would still need to get close to $I$ (touch-to-access).

The attacker cannot calculate and insert a false signature remotely since the connection between $R$ and $C$ is protected by MAC-based integrity checks. In order to frame someone, $A$ has to force the legitimate user to use a hacked reader, which
replaces the CMD with an incorrect command. On the other hand, in order to send the incorrect command him/herself, A needs a valid card and its associated PIN, a reader that can execute reader-card authentication (i.e., both R and C are not reported as stolen), and the access has to be made during the card-owner’s working hours and from within the hospital network. Both of the above situations correspond to scenarios (3) and (4). Scenario (3) should be handled by IMDfence as mentioned above, whereas the likelihood of stacking occurrences in (6) is very low. Also, such situations can be prevented if the guidelines of addressing the non-repudiation gap are followed (see Section IV-C5). All of the above scenarios are summarized in Table III.

3) Smart-card-specific attacks: Since IMDfence employs smart cards, it is important to ensure that it is safe from the weaknesses [45], [46] present in another widely used smart-card system: EMV (Europay, Mastercard, and Visa). These vulnerabilities exist due to the availability of less secure options for backward compatibility and due to a problematic threat model, in which the reader (i.e., the POS terminal) is assumed to be uncorrupted.

One major issue is that the terminal and the card do not share a symmetric key, due to which most of the important data is exchanged in plain-text (e.g., account data, amount etc.). Moreover, in the offline use of the cards that do not support public-key cryptography, the PIN is also sent as plain-text. An attacker can modify the unencrypted initialization messages to force the terminal to use this mode [46]. The PIN can be recorded using e.g., a hacked terminal that has additional probes to read data from the smart card interface. In case of an offline-encrypted PIN, the terminal can be hacked to record the keystrokes. Using the account data and PIN, the attacker can create a magnetic-strip card for use in a country that does not support chip-based smart cards [47].

Another issue is that the terminal cannot use MAC to authenticate messages from the card since they do not share a symmetric key. Cards following the Combined-Data-Authentication (CDA) scheme from EMV address this by employing signatures. However, in the schemes prior to CDA, the terminal is unable to verify the authenticity of all the card messages either due to unavailability of signatures (in the case of Static Data Authentication, SDA) or the signature-less transaction messages (in the case of Dynamic Data Authentication, DDA). As a result, an SDA card can be cloned for use in offline transactions [46], and a stolen DDA card can be employed in a two-card attack, in which the attacker uses his/her own card for PIN verification and uses the stolen card in the transaction phase [42]. Moreover, the response of the card at the end of PIN verification is unauthenticated. As a result, this response can be modified to deceive the terminal into assuming that the entered PIN is correct.

All these attacks exist because some of the critical data is left unencrypted or not signed. In both the online and offline modes of IMDfence, all the data between R and C is encrypted and is authenticated using MACs. Additionally, our recommendation to avoid magnetic-strip-based cards rules out cloning. Similarly, avoiding contactless cards removes an additional attack vector.

Another far more advanced type of attack is the relay attack [45], [47], which exploits the fact that the card users cannot know for sure if the display of the terminal is showing correct information. It is a timing-critical attack where two transactions are simultaneously taking place. The victim inserts his/her card in a counterfeit terminal (e.g., at a restaurant), which is connected to a fake card of the attacker that is inserted in a valid terminal (e.g. at a jewelry store). The details of the fraudulent transaction are forwarded to the victim’s terminal. Her screen shows the correct information, but in effect she pays the amount for the other party.

We observe that the relay attack is far less likely in the case of IMDfence since it requires a legitimate user operating a forged reader. This corresponds to scenario (3), above.

4) Selecting T and NAC values: The touch-to-access principle guarantees that unreasonably high T and NAC values do not cause a security vulnerability in IMDfence, as evident from Section IV-A2. However, the careful reader may have noticed that a prolonged offline operation enabled by such large values may result in R’s and/or IMD’s firmwares getting out-of-date. On the other hand, very small values hinder legitimate access, i.e., availability. Therefore, the hospital server should ensure that appropriate values are assigned to T and NAC (within maximum and minimum limits) based on the patient’s location and the reader-IMD usage patterns.

Regarding the patient’s locality, the probability of having stable Internet connectivity is higher when the patient is based in an urban area compared to a rural setting. Moreover, it stands to reason that the chances of attackers’ presence is ought to be higher in an urban environment. Hence, it makes sense to assign lower T and NAC values for urban areas compared to rural environments. When assigning these values, reader-IMD usage patterns should also be taken into consideration, which depend on the patient condition and IMD type, ranging from critical implants, such as cardiac defibrillators, to less critical ones, such as neurostimulators. The IMDs requiring frequent reader access should be granted larger T and NAC values. Further investigation on this topic is interesting but is considered outside the scope of this work.

It should be noted that the modification of these parameters can be performed throughout the operational lifetime of the IMD. The physician is required to manually modify these values (in S) based on the above guidelines, which then ultimately take effect in the reader-card authentication phase (See Fig. 3).

B. Availability - Battery-DoS protection

As highlighted in Section [11] one of the system requirements is to ensure that the IMD is always available for treatment. One high-likelihood and low-cost attack that affects this requirement is the battery-DoS attack, as practically demonstrated in [3], [4]. This attack forces the IMD to continuously run energy-consuming operations, which results in battery depletion and ultimately causes device shutdown.
For example, the attacker can repeatedly try to establish a connection with the implant using incorrect credentials. The IMD will scrutinize each invalid request through energy-consuming authentication operations, which will drain its battery.

The IMD can defend against battery DoS by employing a zero-power defense (ZPD) scheme in which the authentication operation is executed using borrowed energy [3]. This energy can be harvested from the incoming RF communication messages from the external reader. The IMD switches to battery power only after it has successfully authenticated the external entity.

Another type of DoS attack can occur when the attacker sends repeated communication requests to the implant. For an IMD with a single-processor, such requests may block the device from performing its primary medical functionality. To protect against this, a dual-CPU paradigm can be employed, in which the first CPU executes the original medical functionality, while the second CPU is responsible for dealing with the (secure) communication requests. This dual-core organization offers, then, both functional and power decoupling, which effectively shields the IMD main functionality from battery-DoS attacks, as previously showed in [12].

In order to assess the viability of IMDfence under energy-harvesting conditions (be it in single- or dual-CPU configuration), we construct the following experimental setup: (I) For the computational costs of IMDfence, and similarly to [48], we employ an ARM Cortex-M0+ based 32-bit MCU [49]. Due to its ultra-low-power capabilities, this MCU is becoming increasingly employed in IoT and WBAN settings, and hence, is a plausible choice for this evaluation. The security-related computations, i.e., authenticated encryption (AES-128), cipher-based MAC and random-number generation were performed using the MCU’s dedicated peripherals (“CRYPTO” and “TRNG”); thus, in our energy measurements hardware-accelerated primitives are considered. (II) For the wireless-communication costs of IMDfence, the commercial transceiver ZL70103 specifically designed for IMDs has been used [50]. To get reasonable energy costs for (encrypted) data transmission, we chose packet-size lengths similar to the ones used in low-cost RFID tags, due to their similarities with IMDS in terms of computational, energy and memory constraints [12]. Hence N, ID, CMD and ANS are set to 32, 96, 32 and 64 bits, respectively. The chosen \( \text{sig} \) size is 384 bits, which corresponds to an ECDSA (Elliptic Curve Digital Signature Algorithm) signature with a 96-bit security level.

The protocol sequence executed by the IMD is shown as numbered steps in Figures 5 and 6. The energy consumption for these steps is shown in Fig. 10 using a supply voltage of 3.3 V, and the default MCU and transceiver clock frequencies of 19 MHz and 24 MHz, respectively. The transceiver data rate is set at 400 kbps (with an effective rate of 265 kbps). We observe that the energy required for authentication \((E_{\text{auth}})\), i.e., for steps 1 to 4 in Fig. 5, is only 59.6 μJ. Total IMD energy consumption per type of activity is also shown Fig. 11. For such a low harvested-energy requirement \((E_{\text{auth}})\), it has been demonstrated before in [48] that real-time performance is possible in the IMD.

C. Impact on IMD Lifetime

In the previous section, we discussed the feasibility of IMDfence under energy-harvesting conditions to defend against battery-DoS attacks. In this section, we wish to assess the total energy costs that the IMDfence protocol incurs over the whole lifetime of a modern IMD. To do so, we need to consider realistic usage patterns of actual devices, drawn from medical practice. As an example, we consider the communication session between a commercial bedside reader (Merlin@home™) and a pacemaker [21].

We consider different data volumes being transferred between the reader and IMD, ranging from a daily two-minute communication session to a two-minute weekly session. Since
this reader is intended for monitoring the IMD status, it is assumed that most of the communicated data is transferred from the implant to the reader (e.g., in the form of data logs). Hence, the size of ANS is increased from 64 bits (for a basic session) to 1.9 MB in order to form a two-minute session. However, for worst-case analysis, the transceiver is considered to be enabled throughout this session and we do not assume the use of energy harvesting for ZPD. Moreover, without loss of generality and in order to more accurately (and pessimistically) quantify the cost of adding IMDfence to an existing system, we consider a dual-CPU IMD, as discussed in the previous section. In this configuration, the security CPU is assumed to execute the complete IMDfence protocol, while the medical CPU is set to a 5% duty cycle (active vs. sleep mode), based on typical pacemaker usage [51], and consumes 25 µJ per heartbeat to provide electrical-stimulation impulses, based on reported figures of commercial devices [52].

With the above consideration, the impact of IMDfence on IMD-battery lifetime can be visualized using Fig. 12 for different implantable-grade battery sizes [53]. The variability in each data point captures the different volumes of data transfer between the reader and IMD.

Since the majority of the cryptographic operations in the protocol (authenticated encryption and MAC) are based on symmetric block ciphers, as shown in Figures 3 and 5, it is very interesting to investigate the impact of different cipher versions and/or implementations thereof on IMD lifetime, e.g., a pacemaker. More box plots have thus, been added to Fig. 12 where we readily notice that the hardware implementation of AES-128 significantly outperforms the software AES-128 implementation, plus other lightweight software ciphers such as SPECK and MISTY1. It is also interesting to observe that the energy impact of the hardware AES-128-based protocol is not significant when comparing with an unsecured communication.

D. Performance

To study the impact of IMDfence on IMD performance during normal operation, we will only analyze the bottleneck of the reader-IMD system in this regard, i.e., the IMD itself. This is because modern readers, such as tablets [16], have far superior computational resources (and battery autonomy) than implants. As far as the smart card is concerned, the amount of computations performed by it is approximately the same with that in commercial uses (e.g., EMV), which we know to exhibit adequate performance.

As far as the IMD is concerned and given that the protocol primitives are executed in hardware, the performance figure of merit that is crucial to capture here is the delay that IMDfence incurs to the system, both for security computations and data transmission over the air. For unsecured data transfer, the wireless transceiver incurs a delay of 2.2 ms. As shown in Fig. 10, for secure data transfer the time delay incurred by each (numbered) protocol step is no higher than 6 ms, for a total protocol delay of 15.7 ms. For the time scales involved in biological processes, we can safely assume that the IMDfence delay overhead is negligible.

E. Summary of introduced overheads

Table IV summarizes the impact of IMDfence on an IMD in terms of energy, performance and program-memory footprint. For the hardware implementation of IMDfence it can be observed that, although the energy requirements increase by more than 6 times for a basic session, the total daily IMD consumption (that includes a two-minute communication session and electrical-stimulation costs) increases from 16.60 J to just 17.37 J, which amounts to a mere 4.64% increase, as previously shown in Fig. 12. The reason for this small increase is that the basic medical functionality, e.g., the continuous electrical stimulation of a pacemaker, dominates the security provisions since the reader accesses are far less frequent. In the case of software (AES-128) implementation of IMDfence, the total daily IMD consumption increases by 13.01% (as shown in Fig. 12). Moreover, there is a minimal increase in the computational delay and required program-memory size. In the context of current MCU technology, 6–7 kB of additional footprint is negligible. Hence, we conclude that there is no noticeable change in the IMD costs when IMDfence is employed.

VI. Conclusions

In this paper, we have proposed a novel security protocol for IMD ecosystems, IMDfence. We have demonstrated that our approach has a comprehensive coverage of security requirements that are critical to these systems. This is enabled through the use of a personal smart card and a trusted third party, which helps in facilitating access control, non-repudiation, user authentication, bedside-reader operation and system scalability. We have also shown that IMDfence does not introduce any noticeable overheads in the implant, and it has the ability to support zero-power defense against battery-DoS attacks. It is observed that our proposed protocol increases the total IMD energy consumption by just 4.64%, which is minimal in the context of the IMD lifespan. We have also proposed an OOB-channel-based version of IMDfence, which enables offline or emergency access.

REFERENCES

[1] M. A. Siddiqi, R. M. Seepers, M. Hamad, V. Prevelakis, and C. Strydis, “Attack-tree-based threat modeling of medical implants,” in PROOFS 2018. 7th International Workshop on Security Proofs for Embedded Systems.
M. A. Siddiqi and C. Strydis, “IMD Security vs. Energy: Are we tilting...

T. Belkhouja, X. Du, A. Mohamed, A. K. Al-Ali, and M. Guizani, “e-SAFE: Secure, Efficient and Forensics-Enabled Access to Implantable Medical Devices,” in Proceedings of the 32nd Annual Conference on Computer Security Applications. ACM, 2016, pp. 226–236.

E. Marin, D. Singelée, B. Yang, V. Volski, G. A. Vandenbosch, B. Nuttin, and B. Preneel, “Securing wireless neurostimulators,” in Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy. ACM, 2018, pp. 287–298.

Medtronic, SECURITY BULLETIN – CoxemTSTM Telemetry and Monitoring Accessories, 2019.

FDA, Firmware Update to Address Cybersecurity Vulnerabilities Identified in Abbott’s (formerly St. Jude Medical’s) Implantable Pacemakers: FDA Safety Communication, 2019.

L. Bu, M. G. Karpovsky, and M. A. Kinsy, “Bulwark: Securing implantable medical devices communication channels,” Computers & Security, vol. 68, pp. 498–511, 2019.

T. Belkhouja, X. Du, A. Mohamed, A. K. Al-Ali, and M. Guizani, “Biometric-based authentication scheme for implantable medical devices during emergency situations,” Future Generation Computer Systems, vol. 98, pp. 109–119, 2019.

H. Chi, L. Wu, X. Du, Q. Zeng, and P. Ratazzi, “e-SAFE: Secure, Efficient and Forensics-Enabled Access to Implantable Medical Devices,” in 2018 IEEE Conference on Communications and Network Security (CNS), 2018, pp. 1–9.

T. Belkhouja, X. Du, A. Mohamed, A. K. Al-Ali, and M. Guizani, “Symmetric Encryption Relying on Chaotic Henon System for Secure Hardware-Friendly Wireless Communication of Implantable Medical Systems,” Journal of Sensor and Actuator Networks, vol. 7, no. 2, p. 21, 2018.

C. Strydis, R. M. Seepers, P. Peris-Lopez, D. Siskos, and I. Soudris, “A system architecture, processor, and communication protocol for secure implants,” ACM Transactions on Architecture and Code Optimization (TACO), vol. 10, no. 4, p. 57, 2013.

M. A. Siddiqi and C. Strydis, “IMD Security vs. Energy: Are we tilting at windmills?: POSTER,” in Proceedings of the 16th ACM International Conference on Computing Frontiers. ACM, 2019, pp. 283–285.

Medtronic, AzureTSTM S SR MRI SureScanTM W3SR01, 2017.

S. J. Medical, ConfirmRxTSTM Model DM3500 Insertable Cardiac Monitor - User’s Guide, 2016.

St. Jude Medical, ProclaimTSTM Implantable Pulse Generator - Clinician’s Manual, 2017.

E. Marin Fábregas, “Security and Privacy of Implantable Medical Devices,” Ph.D. dissertation, KU Leuven, Belgium, 2018.

R. J. Anderson, “Liability and computer security: Nine principles,” in European Symposium on Research in Computer Security. Springer, 1994, pp. 231–245.

M. Roe, “Cryptography and evidence,” University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-780, May 2010. [Online]. Available: https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-780.pdf

M. Rostami, A. Juels, and F. Koushanfar, “Heart-to-heart (H2H): authentication for implanted medical devices,” in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013, pp. 1099–1112.

St. Jude Medical, FAQs - Merlin.netTSTM Patient Care Network (FCN) 8.0 Q4A, 2015.

S. J. Medical, Ellipse, Fortify Assura ICD, Quadra Assura, Quadra Assura MP, Unify Assura CRT-D User Manual, 2017.

M. Rushanan, A. D. Rubin, D. F. Kune, and C. M. Swanson, “Sok: Security and privacy in implantable medical devices and body area networks,” in 2014 IEEE Symposium on Security and Privacy. IEEE, 2014, pp. 524–539.

R. M. Seepers, J. H. Weber, Z. Sourdis, and C. Strydis, “Secure key-exchange protocol for implants using heartbeats,” in Proceedings of the ACM International Conference on Computing Frontiers. ACM, 2016, pp. 119–126.

V. Pournaghband, M. Sarrafzadeh, and P. Reiher, “Securing legacy mobile medical devices,” in International Conference on Wireless Mobile Communication and Healthcare. Springer, 2012, pp. 163–172.

J. Sorber, M. Shin, R. Peterson, C. Cornelius, S. Mare, A. Prasad, Z. Marois, E. Smithayer, and D. Koz, “An amulet for trustworthy wearable mHealth,” in Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications. ACM, 2012, p. 7.

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. ACM, 2009, pp. 410–419.

Y. Kim, W. S. Lee, V. Raghunathan, N. K. Iha, and A. Raghunathan, “Vibration-based secure side channel for medical devices,” in Proceedings of the 52nd Annual Design Automation Conference. ACM, 2015, p. 32.

M. Wazid, A. K. Das, N. Kumar, M. Conti, and A. V. Vasilakos, “A novel authentication and key agreement scheme for implantable medical devices deployment,” IEEE journal of biomedical and health informatics, vol. 22, no. 4, pp. 1299–1309, 2018.

D. Mao, L. Zhang, X. Li, and D. Mu, “ Trusted authority assisted three-factor authentication and key agreement protocol for the implantable medical system,” Wireless Communications and Mobile Computing, vol. 2018, 2018.

A. Rathore, C. Fu, A. Mohamed, A. Al-Ali, X. Du, M. Guizani, and Z. Yu, “Multi-layer security scheme for implantable medical devices,” Neural Computing and Applications, pp. 1–14, 2018.

C. Fu, X. Du, L. Wu, Q. Zeng, A. Mohamed, and M. Guizani, “POKs Based Secure and Energy-Efficient Access Control for Implantable Medical Devices,” in Security and Privacy in Communication Networks, 2019, pp. 105–125.

G. Zheng, W. Yang, C. Valli, L. Qiao, R. Shankaran, M. A. Orgun, and S. C. Mukhopadhyay, “Finger-to-heart (2h): Authentication for wireless implantable medical devices,” IEEE journal of biomedical and health informatics, 2018.

N. Ellouze, S. Rekhis, N. Boudriga, and M. Allouche, “Powerless security for cardiac implantable medical devices: Use of wireless identification and sensing platform,” Journal of Network and Computer Applications, vol. 107, pp. 1–21, 2018.

L. Wu, X. Du, M. Guizani, and A. Mohamed, “Access control schemes for implantable medical devices: A survey,” IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1272–1283, 2017.

G. Zheng, R. Shankaran, M. A. Orgun, L. Qiao, and K. Saleem, “Ideas and challenges for secured wireless implantable medical devices: A review,” IEEE Sensors Journal, vol. 17, no. 3, pp. 562–576, 2016.

R. AllTawy and A. M. Youssef, “Security tradeoffs in cyber physical systems: A case study survey on implantable medical devices,” IEEE Access, vol. 4, pp. 959–979, 2016.

C. Camara, P. Peris-Lopez, and J. E. Tapiaidor, “Security and privacy issues in implantable medical devices: A comprehensive survey,” Journal of biomedical informatics, vol. 55, pp. 272–289, 2015.

C.-S. Park, “Security mechanism based on hospital authentication server for secure application of implantable medical devices,” BioMed research international, vol. 2014, 2014.

A. Juels and J. Brainard, “Client puzzles: A cryptographic countermeasure against connection depletion attacks,” in Proceedings of the 1999 Networks and distributed system security symposium. 1999.

ISO, “Information Technology - Security techniques - Entity authentication - Part 2: Mechanisms using symmetric encryption algorithms,” ISO/IEC 9798-2:2008, 2008.

R. Anderson, Security Engineering. Wiley, 2008.

M. Roeschlin, I. Martinovic, and K. B. Rasmussen, “Device pairing at the touch of an electrode,” in Proceedings of NDSS, vol. 18, 2018.

The AVISPA Team, AVISPA v1.1 User Manual, 2006.

J. van den Breekel, “A security evaluation and proof-of-concept relay attack on dutch emv contactless transactions,” Master’s thesis, 2014.

J. van den Breekel, D. A. Ortiz-Yepes, E. Poll, and J. de Ruiter, “Emv in a nutshell,” KPMG, Tech. Rep., 2016.

B. Adida, M. Bond, J. Chlouw, A. Lin, S. Murdoch, R. Anderson, and R. Rivest, “Phish and chips” in International Workshop on Security Protocols. Springer, 2006, pp. 40–48.
[48] M. A. Siddiqi and C. Strydis, “Towards Realistic Battery-DoS Protection of Implantable Medical Devices,” in Proceedings of the 16th ACM International Conference on Computing Frontiers. ACM, 2019, pp. 42–49.

[49] S. Laboratories. (2018) EFM32™ Tiny Gecko 11 32-bit Microcontroller. [Online]. Available: https://www.silabs.com/products/mcu/32-bit/efm32-tiny-gecko-tg11

[50] Microsemi. (2018) ZL70103 Wireless for Implantable Medical Devices. [Online]. Available: https://www.microsemi.com/product-directory/implantable-medical-transceivers/3915-zl70103

[51] P. Lindqvist et al., “Compression and storage of medical data in pacemakers,” Master’s Thesis, Royal Institute of Technology Stockholm, Sweden, 2005.

[52] M. Deterre, “Toward an energy harvester for leadless pacemakers,” Theses, Université Paris Sud – Paris XI, Jul. 2013. [Online]. Available: https://tel.archives-ouvertes.fr/tel-00868838

[53] E. Technologies. (2018) Medical Power. [Online]. Available: https://www.eaglepicher.com/markets/medical-power