Enhanced end-to-end security through symmetric-key cryptography in wearable medical sensor networks

Winderickx, J.; Braeken, A.; Mentens, N.

Health and Technology (2021) 11:511–523
https://doi.org/10.1007/s12553-021-00527-9

ORIGINAL PAPER

Enhanced end-to-end security through symmetric-key cryptography in wearable medical sensor networks

Jori Winderickx[1] · An Braeken[2] · Nele Mentens[1,3]

Received: 23 October 2020 / Accepted: 26 January 2021 / Published online: 30 March 2021

Abstract
This paper describes a security protocol and proof-of-concept implementation for wearable medical sensor devices that are deployed in hospitals. The sensor device measures the patient's vital sign parameters and sends them to the hospital server, such that the data can be processed and stored in the EMR (Electronic Medical Record) of the patient. The proposed security protocol is based on symmetric-key cryptography and addresses the challenges of anonymity, unlinkability, mutual authentication and perfect forward secrecy. Moreover, it relies on decentralised authentication, avoiding an authentication server to be the single point of attack. Besides offering strong security features, the proposed protocol and implementation take into account that sensor devices are typically constrained with respect to communication bandwidth and computation power. Therefore, these parameters are evaluated in addition to the security analysis of the presented protocol. Our solution gives stronger security guarantees than related work, while featuring a comparable computation overhead and the lowest communication overhead.

Keywords Medical sensor data · Symmetric-key security protocol · Anonymity · Perfect forward secrecy · Mutual authentication

1 Introduction

Just like smart homes and smart cities, smart hospitals consist of a network of interconnected devices. One of the main functionalities of these devices is to collect medical sensor data on patients and to send these data to the Electronic Medical Record (EMR) through a central server. This way, part of the monitoring tasks of hospital staff can be taken over by the sensors. Because the sensors and network devices should have a limited influence on the mobility of hospitalised patients, wearable sensors are used that communicate the measured data wirelessly in the network. However, the wireless communication channel introduces a threat with respect to the security of the patients' data. Especially with the European General Data Protection Regulation (GDPR) [1] in place, it is indispensable to protect the privacy of the patient.

In a practical setting, a device carried by the patient, e.g. a smartphone, can be used to collect data wirelessly from the wearable sensors. In the first place, it is important to guarantee end-to-end security between the wearable sensor device and the patient's device. In addition, wirelessly associating a wearable sensor device to a patient's device, which is typically done by hospital staff, should not violate the anonymity of the patient. With these security requirements in mind, this paper proposes a symmetric-key security protocol. Besides the focus on end-to-end security and anonymous association, the protocol also concentrates on the minimisation of the communication bandwidth and the required computation power of the wearable device. The paper explains the different protocol steps for both anonymous association and end-to-end secured communication. Further, a thorough security analysis and evaluation of the required communication and computation

✉ Jori Winderickx
jori.winderickx@kuleuven.be

An Braeken
abraeken@vub.be

Nele Mentens
nele.mentens@kuleuven.be

[1] ES&S and imec-COSIC, KU Leuven, Leuven, Belgium

[2] Department of Industrial Engineering (INDI), Vrije Universiteit Brussel, Brussels, Belgium

[3] LIACS Leiden University , Leiden, The Netherlands

cost are performed. A comparison to previously proposed solutions is shown to be favourable for our protocol.

The outline of the paper is as follows. Section 2 gives an overview of related work in the field of security protocols for healthcare. Section 3 describes the envisioned system and the security requirements. The different steps in our proposed solution are elaborated in Section 4. An evaluation of the security on the one hand, and the computation and communication requirements of the protocol on the other hand, are given in Sections 5 and 6. Finally, Section 7 concludes the paper and points out to future work.

2 Related work

A lot of key agreement schemes for healthcare systems or similar applications have been described in literature. In this overview of related work, we focus on the most recent schemes and explain the differences with respect to our proposed scheme.

In a recent study of Kumar et al. [2], a symmetric-key based key agreement scheme between smart sensor nodes and a home gateway in a Home Area Network (HAN) is proposed. The scheme is designed to provide anonymity, unlinkability, mutual authentication, integrity, perfect forward secrecy, and resistance to general attacks like replay attacks, impersonation attacks and man-in-the-middle attacks. Furthermore, the authors argue that a key agreement scheme in a HAN network should be automatic without human intervention. This leaves the gateway the most interesting attack target as it sits at the centre of the communication model of a HAN. In our setting, the patient is hospitalised and accesses the sensor nodes via a smartphone. The data are highly sensitive and should, therefore, only be accessible via human intervention, e.g. via an authentication step.

The communication model of Shuai et al. [3] is similar to ours as they provide a key agreement scheme for remote patient monitoring. In this scheme, the sensor nodes are connected in a Wireless Body Area Network (WBAN) to a gateway which translates the packets to the remote user. All three entity types (sensor node, gateway, and user) are involved in the key agreement scheme, thus, end-to-end security is not provided. Via this approach, the gateway will be an interesting attack vector, since, it could provide access to all data that are sent from all sensor nodes connected to it. We argue that end-to-end security is essential, because, the sensor nodes generate highly sensitive data.

Another approach to an authenticated key agreement scheme in WBANs is presented by Gupta et al. [4]. The scheme assumes three entities: sensor nodes, gateway/user mobile, and authentication server. The scheme requires five steps to securely establish a shared key. However, the authentication server actively participates in the process. This may cause a slow or a potentially impossible connection setup due to the amount of messages and the remote authentication server which may or may not be reachable. In our proposed scheme, we use a distributed authentication approach to enable the patient/user and sensor node to authenticate each other without the involvement of an authentication server during key establishment.

The work of Chen et al. [5] proposes an anonymous mutual authenticated key agreement scheme for wearable sensor nodes in WBANs. The authors first highlight the shortcomings of another scheme which was proven to be susceptible to the following attacks: offline identity guessing, sensor node impersonation and hub node spoofing attack. Then, they propose an updated scheme that addresses these problems. In their communication model, three entities are considered: second level nodes/sensor nodes, first level nodes/user, and hub nodes/data centre. The sensor nodes use this scheme to establish connection with both the user and the data centre. However, their proposal is not complete in how the devices are provisioned or updated.

The work of Winderickx et al. [6] proposes a security protocol and proof-of-concept implementation to be used in a hospital scenario. The authors conclude that the use of the TLS protocol in DHE_RSA_AES_GCM configuration is the most favourable from four analysed configurations, which include two pre-shared-key based configurations. This configuration relies on public-key based algorithms to provide secure end-to-end communication. Using our proposal, we can provide a more lightweight key establishment protocol, as we are relying on symmetric-key cryptography and hash functions, while still meeting all security requirements needed in wireless medical sensor networks.

3 System assumptions, threat model and security requirements

3.1 System assumptions

The system model is presented in Fig. 1. The entities that are considered are the following: Sensor nodes (S), Patient's smartphone (P), and Remote Centre (RC). All entities can directly or indirectly communicate with each other, e.g. via an access point or a gateway. The sensor nodes are constrained devices with a wireless interface that are placed on or close to the patient, and, they measure vitals like heart rate and blood pressure. These nodes are divided into groups that represent the selection of sensor nodes required per patient. Furthermore, the data generated by the sensor nodes in one group are collected by the patient on his/her personal device such as a smartphone. During or after data

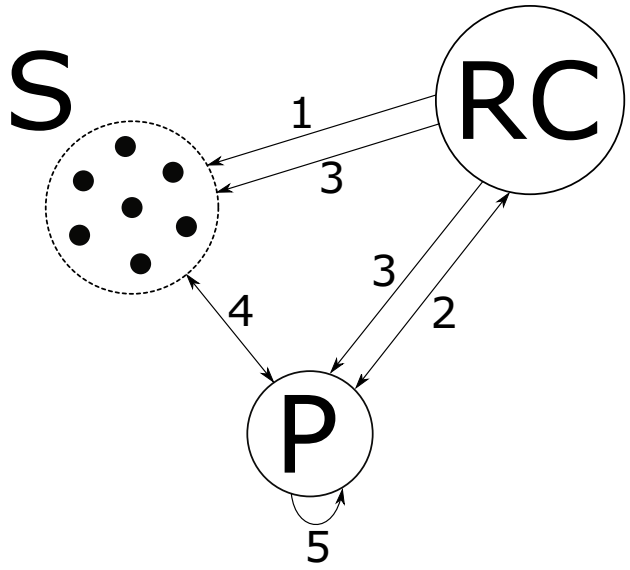

Fig. 1 System architecture and communication actions used in the proposed scheme, where S denotes the sensor nodes, RC is the remote centre, and P is the patient's smartphone

collection, the patient's smartphone can share the data with the hospital's EMR server over a secured channel. The RC is an authorisation server that can validate and authenticate the patient and is able to manage and link the sensor nodes to a patient. Both the patient's device and RC are not considered to be constrained in terms of storage, computation, and energy.

The proof-of-concept implementation that is used for the validation of our proposed scheme consists of one wearable health sensor that is placed on the chest of a patient and monitors vitals such as heart rate and blood pressure variation, and communicates via a WiFi wireless interface. This sensor transfers its data to the patient, which is emulated via a Python program on a laptop that is connected to the network.

The consecutive actions between the entities, that will be explained in Section 4, are indicated in Fig. 1:

1. Offline installation of the sensor nodes by the remote centre;
2. Patient registration with the remote centre;
3. Linking of the patient to a group of sensors by the remote centre;
4. Key agreement between the sensor nodes and the patient's smartphone;
5. Update of the patient's pin.

3.2 Threat model

We assume the Dolev-Yao threat model [7]. An attacker knows the protocol used and may eavesdrop, modify, corrupt, delete, redirect, or replay all the messages that are transmitted. Additionally, this attacker can capture a sensor node and extract all stored parameters from its memory using side-channel attacks. Even in this case, it should not be possible to derive the previously constructed session keys and decrypt the already sent data. The patient's device and RC are assumed to be monitored closely by either the patient or IT staff. The attacker should, therefore, not be able to capture these entities. Furthermore, it is assumed that the RC can communicate securely using classic techniques during the set-up and registration phase. The main focus of our protocol is on providing end-to-end security between the sensor node and the patient's device. However, the goal of the attacker might be to illegitimately obtain the data that are measured by the sensor node, to control the access to the sensor node, or to perform service degradation.

3.3 Security requirements

The required cryptographic properties are in line with related work and can be summarised as follows:

R.1 **Anonymity and unlinkability** The identity of the patient for whom the data is collected should not be accessible by an adversary. Also, an adversary should not be able to link multiple sessions to each other that are established between a sensor node and the patient's smartphone.
R.2 **Mutual authentication** Both the sensor node and the patient's smartphone should be able to authenticate each other's identity.
R.3 **Perfect forward secrecy** Previously established sessions keys should not be computable if an adversary acquires the long-term keys of both the sensor node and the patient's smartphone.
R.4 **Protection against general attacks** We consider replay, man-in-the-middle, impersonation, synchronisation, and password guessing attacks as potential attacks.
R.5 **Distributed authentication** The sensor node and patient's smartphone should be able to establish session keys without the intervention of the remote centre.
R.6 **Two-factor authentication** The key establishment should take into account the authentication of the user and the device.

Distributed authentication Our goal of distributed authentication is to reduce the amount of single point attack vectors that impact all data channels, e.g. the authentication server. While an additional attack vector is added by involving the patient's smartphone, powerful cryptographic techniques like asymmetric cryptography can be used on this device to secure it. Furthermore, only one or at maximum a small number of sessions of a specific patient can be compromised

Fig. 2 Action 1: Offline installation of sensor nodes by RC

Action 1: Offline installation of sensor nodes by RC

RC		**Sensor**(S_s)
Generates secret master keys: x_g, y_g, z_g for a certain group of sensor nodes.		
Computes for each sensor:		
$A_s = \mathsf{H}(ID_s \| ID_g \| x_g)$		
$B_s = A_s \oplus \mathsf{H}(y_g)$		
$\mathsf{hk}_s = \mathsf{H}(A_s \| y_g)$		
$\mathsf{H}(A_s)$		
$\mathsf{H}(z_g)$		
	$\xrightarrow[\text{physical channel}]{\langle B_s, \mathsf{H}(A_s), \mathsf{H}(z_g), \mathsf{hk}_s \rangle}$	
		Stores B_s
		Stores $\mathsf{H}(A_s)$
		Stores $\mathsf{H}(z_g)$
		Stores hk_s

on the patient's smartphone before the compromise is detected by the smartphone, the patient or the system. It is therefore a smaller risk that has to be taken into account than involving a single point of attack that could impact all data channels.

4 Proposed solution

Our proposed scheme exists of the following five actions: (1) the remote centre (RC) installs the sensor nodes (S) offline, (2) the patient (P) is registered with the RC, (3) the RC links P to S, (4) S and P agree on a key, and (5) P updates the PIN. The overall flow of these actions is presented in Fig. 1, while more details are provided in Figs. 2, 3, 4, and 5. Table 1 presents the symbols and their descriptions used throughout the paper. First a brief description of the five actions will be given, after which a more detailed explanation is presented.

– Setup phase: Action 1 consists of the initialisation of the sensor nodes and the RC. This action is done, for example, by the IT staff in a controlled and secure environment.

Fig. 3 Patient registration with RC

Action 2: Patient registration with RC

Patient		**RC**
Smartphone generates random value b		
Patient enters ID_p and PIN_p		
$RPW_p = \mathsf{H}(PIN_p \oplus b)$		
$y_g^* = \mathsf{H}(RPW_p \| ID_p)$		
	$\xrightarrow[\text{physical channel}]{\langle ID_p, y_g^* \rangle}$	
		Validates ID_p and stores $\langle ID_p, y_g^* \rangle$

Fig. 4 RC links patient to a group of sensor nodes

Action 3: RC links the patient to a group of sensor nodes

RC		**Sensor**(S_s)
Stores $x_g, y_g, z_g, ID_g,$ $[ID_s], ID_p, y_g^*$		Stores $ID_s, B_s,$ $\mathsf{H}(A_s), \mathsf{H}(z_g), \mathsf{hk}_s$
$\mathsf{H}(ID_p)$ $\mathsf{H}_{yg}^* = \mathsf{H}(y_g) \oplus y_g^*$ For each sensor: $A_s = \mathsf{H}(ID_s \Vert ID_g \Vert x_g)$ $HK_s = \mathsf{H}(A_s \Vert y_g) \oplus \mathsf{H}(A_s \Vert y_g^*)$		
	$\xrightarrow{\mathsf{Enc}_{\{\mathsf{H}(z_g)\}}(\mathsf{H}(ID_p), \mathsf{H}_{yg}^*, HK_s)}$	
		$B_s^* = B_s \oplus \mathsf{H}_{yg}^*$ Updates B_s with B_s^* $HREG = \mathsf{H}(\mathsf{H}(ID_p) \Vert \mathsf{H}(z_g))$ $\mathsf{hk}_s^* = \mathsf{hk}_s \oplus HK_s$ Updates hk_s with hk_s^* Creates counter $N_s = 0$ Stores $HREG, \mathsf{H}(ID_p)$
		Patient
		Stores b
$HREG = \mathsf{H}(\mathsf{H}(ID_p) \Vert \mathsf{H}(z_g))$ Replaces y_g with y_g^*		
	$\xrightarrow{\mathsf{Enc}_{y_g^*}\{HREG\}}$	
		Stores $HREG$

- Registration phase: At first, the patient needs to register himself/herself to the RC (Action 2). Then, the RC authenticates the patient, remotely configures the sensor nodes, and sends the key that links the patient to the sensor nodes to the patient's smartphone (Action 3).
- Run-time phase: When the sensor nodes are enabled, for example by a nurse that puts the sensor nodes on the patient, they will initiate the key agreement protocol between the sensor node and the smartphone (Action 4).
- Update phase: During the run-time phase, the patient can also choose to update its PIN number using Action 5.

The core of the solution is contained in Action 4 and needs to be continuously repeated for each new session. The Actions 1, 2, and 3 are only required during initialisation, and Action 5 needs to be performed after a significant amount of sessions.

4.1 Action 1: offline installation of sensor nodes by RC

In Action 1 (shown in Fig. 2), let x_g, y_g, z_g be three secret master keys chosen by the RC for a group of sensor nodes. Denote ID_s, ID_g the identifiers of respectively a sensor node and a group. The RC computes for each sensor node their secret identity $A_s = \mathsf{H}(ID_s \Vert ID_g \Vert x_g)$, public identity $B_s = A_s \oplus \mathsf{H}(y_g)$ and hash key $\mathsf{hk}_s = H(A_s \Vert y_g)$. Then, the RC shares the parameters $B_s, \mathsf{H}(A_s), \mathsf{H}(z_g), hk_s$ with each sensor node in the group via a controlled and secure environment, e.g. a physical channel. Note that the parameter $\mathsf{H}(z_g)$ is the encryption key shared by all sensor nodes in a specific group.

4.2 Action 2: patient registration with RC

In Action 2 (shown in Fig. 3), the patient registers himself/herself using a smartphone via an interface of RC. For example, an application on the smartphone that wirelessly communicates with RC via the available Wi-Fi network using a secured data channel. Let b be a random number chosen by the smartphone. The patient enters/scans his/her identifier ID_p and chooses a personal PIN number PIN_p. The smartphone, then, computes the patient's password $RPW_p = \mathsf{H}(PIN_p \oplus b)$ and key $y_g^* = \mathsf{H}(RPW_p \Vert ID_p)$. Finally, the smartphone shares ID_p, y_g^* with RC. The RC uses the identifier to validate and authenticate the patient. Upon positive authentication, the RC stores the tuple $\langle ID_p, y_g^* \rangle$.

Fig. 5 Key agreement between Sensor and Patient

Action 4: Key agreement between Sensor and Patient

Sensor(S_s)	**Patient**
Stores $ID_s, B_s, \mathsf{H}(A_s), \mathsf{H}(z_g)$ $HREG, \mathsf{H}(ID_p), \mathsf{hk}_s, N_s$	Stores $b, HREG$
Generates random R_1 and timestamp T_1 $M_1 = R_1 \oplus \mathsf{H}(A_s)$ $M_2 = \mathsf{H}(HREG \Vert \mathsf{H}(A_s) \Vert R_1 \Vert T_1 \Vert N_s)$ $M_3 = B_s \oplus \mathsf{H}(HREG \Vert T_1)$	
$\langle M_1, M_2, M_3, T_1, N_s \rangle \longrightarrow$	
	Generates timestamp T_2 Checks $\lvert T_2 - T_1 \rvert < \Delta T$ $B_s = M_3 \oplus \mathsf{H}(HREG \Vert T_1)$ Patient enters ID_p and PIN_p $y_g^* = \mathsf{H}(\mathsf{H}(PIN_p \oplus b) \Vert ID_p)$ $A_s = B_s \oplus y_g^*$ $R_1 = M_1 \oplus \mathsf{H}(A_s)$ $M_2^* = \mathsf{H}(HREG \Vert \mathsf{H}(A_s) \Vert R_1 \Vert T_1 \Vert N_s)$ Checks $M_2^* \stackrel{!}{=} M_2$ Generates random R_2 If hk_s is not known: Stores $\mathsf{hk}_s = \mathsf{H}^{N_s}(\mathsf{H}(A_s \Vert y_g^*))$ $\mathsf{sk} = \mathsf{H}(R_1 \Vert R_2 \Vert T_1 \Vert T_2 \Vert B_s \Vert \mathsf{H}(ID_p) \Vert \mathsf{hk}_s)$ Replaces hk_s with $\mathsf{H}(\mathsf{hk}_s)$ $M_4 = R_2 \oplus \mathsf{H}(A_s)$ $M_5 = \mathsf{H}(\mathsf{sk} \Vert HREG \Vert T_1 \Vert T_2)$
	$\langle T_2, M_4, M_5 \rangle \longleftarrow$
Checks $\lvert T_2 - T_1 \rvert < \Delta T$ $R_2 = M_4 \oplus \mathsf{H}(A_s)$ $\mathsf{sk} = \mathsf{H}(R_1 \Vert R_2 \Vert T_1 \Vert T_2 \Vert B_s \Vert \mathsf{H}(ID_p) \Vert \mathsf{hk}_s)$ $M_5^* = \mathsf{H}(\mathsf{sk} \Vert HREG \Vert T_1 \Vert T_2)$ Checks $M_5^* \stackrel{!}{=} M_5$ Replaces hk_s with $\mathsf{H}(\mathsf{hk}_s)$ Increments N_s	

4.3 Action 3: sensor - patient grouping

Before Action 3 (shown in Fig. 4), the RC decides which group of sensor nodes will be linked to the patient. Then, the RC computes the hash of the patients identifier $\mathsf{H}(ID_p)$ and the key modifier $\mathsf{H}_{yg}^* = \mathsf{H}(y_g) \oplus y_g^*$. For each sensor, the RC also computes the respective hash key modifier $HK_s = \mathsf{H}(A_s \Vert y_g) \oplus \mathsf{H}(A_s \Vert y_g^*)$. Next, the RC shares $\mathsf{H}(ID_p), \mathsf{H}_{yg}^*, HK_s$ securely by encrypting the data with the encryption key $\mathsf{H}(z_g)$. Using these values, the sensor

Table 1 Description of the symbols used throughout the paper

Symbol	Description
RC	Remote Centre
S_s	Sensor s
ID_i	Identifier of entity i
A_s	Sensor's secret identity
B_s	Sensor's public identity
x_g, y_g, z_g	Secret master keys
PIN_p	Patient's PIN
RPW_p	Patient's password
N	Hash key counter
T_i	Current timestamp
R_i, b	Random value
M_i	Message value
ΔT	Maximum delay
$HREG$	Patient's link
hk	Hash key
sk	Secret key
H	Cryptographic hash function
$\oplus$	XOR operation
$\|\|$	Concatenation operation

updates its public identity B_s with $B_s^* = B_s \oplus \mathsf{H}_{yg}^*$ and hash key hk_s with $\mathsf{hk}_s^* = \mathsf{hk}_s \oplus \mathsf{hk}_s$, and computes and stores the key that links the sensor node group to the patient $HREG = \mathsf{H}(\mathsf{H}(ID_p)\|\mathsf{H}(z_g))$. Furthermore, a counter N_s is created that represents the amount of times the hash key is hashed. After configuring the sensor nodes, the RC also calculates the key $HREG$ and shares it with the patient's smartphone using a secured communication channel.

4.4 Action 4: key agreement between sensor and patient

The run-time phase is defined by Action 4. Let R_1, T_1 be a random value and a timestamp generated by the sensor node. Three message parameters are computed: $M_1 = R_1 \oplus \mathsf{H}(A_s)$, $M_2 = \mathsf{H}(HREG\|\mathsf{H}(A_s)\|R_1\|T_1\|N_s)$, and $M_3 = B_s \oplus \mathsf{H}(HREG\|T_1)$. The sensor shares M_1, M_2, M_3, T_1, N_s with the smartphone. Next, the smartphone verifies that T_1 lies within a maximum delay interval ($|T_2 - T_1| < \Delta T$) by generating its own timestamp T_2. Using M_3 and key $HREG$, the public identity of the sensor B_s can be determined. To calculate the secret identity of the sensor node, the smartphone requests the identifier ID_p and PIN PIN_p of the patient. Using these values, the key $y_g^* = \mathsf{H}(\mathsf{H}(PIN_p \oplus b)\|ID_p)$ and the secret identity $A_s = B_s \oplus y_g^*$ can be calculated. Via the secret identity and parameter M_1, the smartphone can compute random value $R_1 = M_1 \oplus \mathsf{H}(A_s)$. Then, the correctness of A_s, R_1, T_1, N_s is validated using parameter $M_2 \stackrel{!}{=} M_2^* = \mathsf{H}(HREG\|H(A_s)\|R_1\|T_1\|N_s)$. As a response, the patient's smartphone executes the following actions. If the hash key of this sensor is not yet known by the patient, it can calculate the current hash key using $\mathsf{hk}_s = \mathsf{H}^{N_s}(\mathsf{H}(A_s\|y_g^*))$. Now, with a randomly chosen parameter R_2, the secret encryption key that is to be used for securing the communication between the sensor node and smartphone can be computed via $\mathsf{sk} = \mathsf{H}(R_1\|R_2\|T_1\|T_2\|B_s\|\mathsf{H}(ID_p)\|\mathsf{hk}_s)$. After the key is generated, the smartphone updates the hash key $\mathsf{hk}_s = \mathsf{H}(\mathsf{hk}_s)$ and generates message parameters $M_4 = R_2 \oplus \mathsf{H}(A_s)$ and $M_5 = \mathsf{H}(\mathsf{sk}\|HREG\|T_1\|T_2)$. Then, the smartphone shares T_2, M_4, M_5 with the sensor. The timestamp T_2 can be verified using Timestamp T_1 by comparing it to the maximum delay interval ($|T_2 - T_1| < \Delta T$). Random value R_2 can be computed via $R_2 = M_4 \oplus \mathsf{H}(A_s)$. Next, the sensor can also compute the secret key sk. The correctness of the secret key sk and timestamp T_2 are verified via message parameter $M_5 \stackrel{!}{=} M_5^* = \mathsf{H}(\mathsf{sk}\|HREG\|T_1\|T_2)$. Finally, the sensor updates the hash key $\mathsf{hk}_s = \mathsf{H}(\mathsf{hk}_s)$ and increments counter N_s.

4.5 Action 5: patient's PIN update

The patient can also choose to update the PIN during the run-time phase using Action 5, see Fig. 6. The action's procedure is as follows. Via the smartphone, the patient first enters the old PIN_p and the new PIN_p^*. Then, the smartphone calculates a new value $b^* = PIN_p^* \oplus PIN_p \oplus b$ and replaces the old value with the new value $b = b^*$.

5 Security analysis

This section provides the detailed security analysis of our proposed scheme according to the security requirements defined in Section 3. First, the RUBIN formal analysis technique is used to prove that a secure session key is established between the patient's smartphone and a sensor node. Secondly, an informal analysis is given of different popular attacks. Finally, this section compares the offered security features to those of related work.

Action 5: Patient's PIN update

Patient(P)
Stores $b, HREG, (\mathsf{sk}, \mathsf{hk}_s, N_s)$

Patient enters PIN_p and PIN_p^*
$b^* = PIN_p^* \oplus PIN_p \oplus b$
Updates b with b^*

Fig. 6 Procedure to update the PIN of a patient

5.1 Formal verification

Our scheme is verified by using the RUBIN technique that can provide a non-monotonic logic-based verification proof. With RUBIN we can analyse the protocol using a specification that is close to the implementation. Its specification and other examples can be found in [8–10].

5.1.1 Specification

Our scheme assumes that both the patient's smartphone (P) and all sensor nodes (S) trust the Remote Centre (RC). Therefore, we focus the proof on the key agreement phase (Action 4). In terms of global sets, the principal set is $PS = \{RC, P, S\}$ where S is the list of sensor nodes S_s in a group. Without loss of generality, we will provide the proof with one of the sensor nodes S_s in the group. The rule set contains the inference rules as defined by Rubin et al. [9]. The secret set is $SS = \{ID_p, PIN_p, z_g, x_g, y_g, HREG, HK_s\}$ at the start of Action 4. The observer set is as follows:

- Observer $(ID_p) : \{RC, P\}$
- Observer $(\mathsf{H}(ID_p)) : \{RC, S, P\}$
- Observer $(PIN_p) : \{P\}$
- Observer $(z_g) : \{RC\}$
- Observer $(\mathsf{H}(z_g)) : \{RC, P\}$
- Observer $(x_g) : \{RC\}$
- Observer $(y_g) : \{RC\}$
- Observer $(y_g^*) : \{RC, P\}$
- Observer $(HREG) : \{RC, S, P\}$
- Observer $(A_s) : \{RC\}$
- Observer $(\mathsf{H}(A_s)) : \{RC, S\}$
- Observer $(B_s) : \{RC, S\}$

Now we define the local set for each entity, starting with S_s. Note that the local set of an entity P exists of the Possession Set $POSS(P)$, Belief Set $BEL(P)$, and Behaviour list $BL(P)$.

Principal S_s
$POSS(S_s) = ID_s, B_s, \mathsf{H}(A_s), \mathsf{H}(z_g), \mathsf{H}(ID_p), HREG, \mathsf{hk}_s, N_s$
$BEL(S_s) = \#(HREG), \#(\mathsf{hk}_s), \#(\mathsf{H}(A_s)), \#(\mathsf{H}(z_g)), \#(\mathsf{H}(ID_p)), \#(B_s)$
$BL(S_s) =$

SA1 Generate-timestamp(T_1)
SA2 Generate-secret(R_1)
SA3 $M_1 \leftarrow R_1 \oplus \mathsf{H}(A_s)$
SA4 $M_2 \leftarrow$ SHA256(Concat($HREG, \mathsf{H}(A_s), R_1, T_1, N_s$))
SA5 $M_3 \leftarrow B_s \oplus$ SHA256(Concat($HREG, T_1$))
SA6 Concat(M_1, M_2, M_3, T_1, N_s)
SA7 Send($P, \{M_1 \cdot M_2 \cdot M_3 \cdot T_1 \cdot N_s\}$)
SA8 Update($\{M_1 \cdot M_2 \cdot M_3 \cdot T_1 \cdot N_s\}, \mathcal{W}$)
SA9 Forget(M_1, M_2, M_3)
SA10 Receive($P, \{T_2 \cdot M_4 \cdot M_5\}$)
SA11 Split($\{T_2 \cdot M_4 \cdot M_5\}$)
SA12 Check-freshness(T_2)
SA13 $R_2 \leftarrow M_4 \oplus \mathsf{H}(A_s)$
SA14 $sk \leftarrow$ SHA256(Concat($R_1, R_2, T_1, T_2, B_s, \mathsf{H}(ID_p), \mathsf{hk}_s$))
SA15 $M_5^* \leftarrow$ SHA256(Concat(sk, $HREG, T_1, T_2$))
SA16 Check(M_5, M_5^*)
SA17 $\mathsf{hk}_s \leftarrow$ SHA256(hk_s)
SA18 Forget-secret(R_1, R_2)
SA19 Forget(T_1, T_2, M_4, M_5)

Principal P
$POSS(P) = b, HREG$
$BEL(P) = \#(HREG)$
$BL(P) =$

PA1 Receive($S_s, \{M_1 \cdot M_2 \cdot M_3 \cdot T_1 \cdot N_s\}$)
PA2 Split($\{M_1 \cdot M_2 \cdot M_3 \cdot T_1 \cdot N_s\}$)
PA3 Check-freshness(T_1)
PA4 $B_s \leftarrow$ SHA256(Concat($HREG, T_1$))
PA5 Input(ID_p, PIN_p)
PA6 $y_g^* \leftarrow$ SHA256(Concat(SHA256($PIN_p \oplus b$), ID_p))
PA7 $A_s \leftarrow B_s \oplus y_g^*$
PA8 $\mathsf{H}(A_s) \leftarrow$ SHA256(A_s)
PA9 $R_1 \leftarrow M_1 \oplus \mathsf{H}(A_s)$
PA10 $M_2^* \leftarrow$ SHA256(Concat($HREG, \mathsf{H}(A_s), R_1, T_1, N_s$))
PA11 Check(M_2, M_2^*)
PA12 Forget(M_1, M_2, M_3)
PA13 Generate-secret(R_2)
PA14 Generate-timestamp(T_2)
PA15 If $\mathsf{hk}_s \notin POSS(P)$ then $hk_s \leftarrow$ SHA256^{N_s} (SHA256(Concat(A_s, y_g^*)))
PA16 sk $\leftarrow$ SHA256(Concat($R_1, R_2, T_1, T_2, B_s, H(ID_p), \mathsf{hk}_s$))
PA17 $\mathsf{hk}_s \leftarrow$ SHA256(hk_s)
PA18 $M_4 \leftarrow R_2 \oplus \mathsf{H}(A_s)$
PA19 $M_5 \leftarrow$ SHA256(Concat(sk, $HREG, T_1, T_2$))
PA20 Concat(T_2, M_4, M_5)
PA21 Send($S_s, \{T_2 \cdot M_4 \cdot M_5\}$)
PA22 Update($\{T_2 \cdot M_4 \cdot M_5\}, \mathcal{W}$)
PA23 Forget-secret($R_1, R_2, ID_p, PIN_p, A_s, \mathsf{H}(A_s)$)
PA24 Forget(T_1, T_2, M_4, M_5, B_s)

5.1.2 The analysis

The scheme starts with the execution of the SA1-SA9 RUBIN actions of $BL(S_s)$, resulting in the updated local set of principal S_s described below. Also, the Update action results in Observers(M_1, M_2, M_3, T_1, N_s) = $\mathcal{W}$. Principal S_s generates message parameters M_{1-3} that are believed to be fresh because of the freshness believes of terms R_1, T_1, and $HREG$. After sending the message, principal S_s discards M_1, M_2, M_3 in SA9, since, they are no longer needed.

$POSS(S_s) = ID_s, B_s, \mathsf{H}(A_s), \mathsf{H}(z_g), HREG, \mathsf{H}(ID_p), \mathsf{hk}_s, N_s, T_1, R_1, M_1, M_2, M_3$
$BEL(S_s) = \#(HREG), \#(\mathsf{hk}_s), \#(H(A_s)), \#(T_1), \#(R_1), \#(M_1), \#(M_2), \#(M_3)$
$BL(S_s) = SA1, ..., SA8$

The next four RUBIN actions to be executed are those of principal P, because, the terms $\{M_1 \cdot M_2 \cdot M_3 \cdot T_1 \cdot N_s\}$ are sent to it. The new local set described below is the result of this. The terms M_{1-3}, N_s are not believed to be fresh, because, they cannot directly be verified. The freshness of timestamp T_1, however, is ensured using the Check-freshness action.

$POSS(P) = b, HREG, N_s, T_1, M_1, M_2, M_3, B_s$
$BEL(P) = \#(HREG), \#(T_1),$
$\{\{S_s, P, RC\} \subseteq \text{Observer}(HREG)\}$
$BL(P) = PA1, \ldots, PA4$

The next RUBIN action collects the input ID_p and PIN_p of the patient, but, it can not be verified to be fresh. Then, the following six RUBIN actions of principal P are executed. They results in the new local set described below. By checking $M_2 \stackrel{!}{=} M_2^*$, $\text{Observer}(H(A_s), R_1) = \{S_s, P\}$ is proven, since, $\{S, P, RC\} \subseteq \text{Observer}(HREG)$ and $\text{Observer}(H(A_s)) = \{RC, S_s, P\}$.

$POSS(P) = b, HREG, N_s, T_1, M_1, M_2, M_3, B_s, y_g^*, A_s, H(A_s), R_1, ID_p, PIN_p$
$BEL(P) = \#(HREG), \#(T_1), \#(A_s), \#(R_1)$
$BL(P) = PA1, \ldots, PA11$

The PA12-PA22 RUBIN actions of principal P are performed if the verification of M_2 is successful. The local set described below is the result. The secret and hash key are generated and are believed to be fresh via respectively $\#(A_s)$ and $\#(R_1, R_2, T_1, T_2, hk_s)$. Then, principal P calculates message parameters M_4 and M_5.

$POSS(P) = b, HREG, (\mathsf{hk}_s, N_s), T_1, B_s, y_g^*, A_s, \mathsf{H}(A_s), R_1, ID_p, PIN_p, R_2, T_2, M_4, M_5, \mathsf{sk}$
$BEL(P) = \#(HREG), \#(T_1), \#(A_s), \#(R_1), \#(R_2), \#(T_2), \#(\mathsf{sk}), \#(\mathsf{hk}_s), \#(M_4), \#(M_5)$
$BL(P) = PA1, \ldots, PA22$

The last RUBIN action of P is to forget all terms computed or generated in this process except for (hk_s, N_s) and sk via RUBIN actions $PA23$ and $PA24$.

Principal P sends $\{T_2 \cdot M_4 \cdot M_5\}$ to principal S_s which enables the sensor node to continue executing RUBIN actions SA10-SA17. The resulting local set is as provided below. After splitting the received terms, the freshness of timestamp T_2 is verified. Via $\text{Check}(M_5, M_5^*)$, S_s can believe $\text{Observer}(\mathsf{sk}) = \{S_s, P\}$. Furthermore, the freshness of sk is ensured via the random values and the time stamps. Next, the hash key hk_s^i is updated with hk_s^{i+1}, thus, $\text{POSS}(S_s) := \text{POSS}(S_s) - \mathsf{hk}_s^i + \mathsf{hk}_s^{i+1}$.

$POSS(S_s) = ID_s, B_s, \mathsf{H}(A_s), \mathsf{H}(z_g), HREG, \mathsf{H}(ID_p), \mathsf{hk}_s, N_s, T_1, R_1, T_2, R_2, M_4, M_5, \mathsf{sk}$
$BEL(S_s) = \#(HREG), \#(hk_s), \#(H(A_s)), \#(T_1), \#(R_1), \#(T_2), \#(sk)$
$BL(S_s) = SA1, \ldots, SA17$

Finally, principal S_s forgets all terms computed or generated in this process except for (hk_s, N_s) and sk via RUBIN actions $SA18$ and $SA19$. The observer set is as follows at the end of Action 4:

- Observer $(ID_p) : \{RC, P\}$
- Observer $(\mathsf{H}(ID_p)) : \{RC, S, P\}$
- Observer $(PIN_p) : \{P\}$
- Observer $(z_g) : \{RC\}$
- Observer $(\mathsf{H}(z_g)) : \{RC, P\}$
- Observer $(x_g) : \{RC\}$
- Observer $(y_g) : \{RC\}$
- Observer $(y_g^*) : \{RC, P\}$
- Observer $(HREG) : \{RC, S, P\}$
- Observer $(A_s) : \{RC\}$
- Observer $(\mathsf{H}(A_s)) : \{RC, S\}$
- Observer $(B_s) : \{RC, S\}$
- Observer $(\mathsf{sk}) : \{S, P\}$
- Observer $(\mathsf{hk}) : \{S, P\}$

This analysis implies that:

- R_1, T_1, R_2, T_2 are fresh for each session and are known by the legitimate S_s and P. This ensures resilience against replay attacks and linkability.
- S_s and P are mutually authenticated, protecting against man-in-the-middle and impersonation attacks.
- ID_p and A_s are only possessed by P and RC, providing anonymous communication.
- The secret key sk is only known to S_s and P and is an independent key for each session as it is computed over random values and the hash key hk_s. Note that all consecutive hash keys can only be known at any time by P and RC, thus, the scheme achieves perfect forward secrecy if a sensor node is compromised.
- The random value b and a scan based ID_p, which is a random identifier generated by the hospital, are used to protect against password guessing attacks.

5.2 Informal verification

The security verification is continued in this subsection with an informal discussion of the provided security properties

and the protection against potential security attacks. Note that we assume the threat model described in Section 3.

Patient impersonation attack Suppose an attacker $\mathcal{A}$ tries to create valid input credentials. $\mathcal{A}$ needs to enter an identifier and the PIN code (ID_p^a, PIN_p^a) on the patient's smartphone, assuming that $\mathcal{A}$ can get access to the smartphone. $\mathcal{A}$ cannot determine the correct values via eavesdropping the communication, because, they are not shared in plain text nor over an insecure channel. Furthermore, ID_p cannot be computed from $\mathsf{H}(ID_p)$ or $HREG = \mathsf{H}(\mathsf{H}(ID_p)\|\mathsf{H}(z_g))$, which are stored on the sensor nodes, since, we consider the hash function a one-way function.

Sensor node impersonation attack An attacker $\mathcal{A}$ intercepts the first transmission $\langle M_1, M_2, M_3, T_1, N_s\rangle$ of sensor node S_s and tries to generate a valid message $\langle M_1^a, M_2^a, M_3^a, T_1^a, N_s^a\rangle$ by impersonating this node. $\mathcal{A}$ can generate a random value R_1^a, timestamp T_1^a, and hash key counter N_s^a. However, $\mathcal{A}$ cannot generate valid message parameters M_1, M_2, or M_3, because, this requires the knowledge of secret values $\mathsf{H}(A_s)$ and $HREG$. Moreover, these secret values cannot be guessed in polynomial time. Message parameter M_1 protects $\mathsf{H}(A_s)$ via the one-time pad technique, and, message parameters M_2 and M_3 protect $HREG$ via the one-way hash function.

Replay attack Assume an attacker $\mathcal{A}$ has eavesdropped on the messages that are sent during the key establishment procedure (Action 4) and tries to replay these messages to either the patient's smartphone or sensor node. The freshness of both messages are easily verifiable via the timestamps T_1 and T_2, also, the integrity of the timestamp is guaranteed via M_2 and M_5.

Eavesdropping attack Messages $\langle M_1, M_2, M_3, T_1, N_s\rangle$ and $\langle T_2, M_4, M_5\rangle$ are, therefore, seen by $\mathcal{A}$. The formula $\mathsf{sk} = \mathsf{H}(R_1\|R_2\|T_1\|T_2\|B_s\|\mathsf{H}(ID_p)\|\mathsf{hk}_s)$ is used to generate the secret key sk. $\mathcal{A}$ cannot derive R_1, R_2, B_s, $\mathsf{H}(ID_p)$, and hk_s, since, these values are protected via either the one-time pad technique or the one-way hash function.

Privileged insider attack Assume an adversary $\mathcal{A}$ who may be a privileged sensor node that is connected to patient P_i and that tries to establish a connection with another patient's smartphone P_j. $\mathcal{A}$ cannot compute the sensor node-patient link $HREG_j$, thus, he cannot create a valid message parameter M_2. Upon validation, the patient's smartphone will abort the key establishment procedure. However, $\mathcal{A}$ can determine the other sensor nodes that are linked to P_i by eavesdropping on the communication. Using the value $HREG_i$ and the eavesdropped message parameter T_1, the identity B_s of all linked sensor nodes can be computed via $B_s = M_3 \oplus \mathsf{H}(HREG_i\|T_1^a)$.

Offline password guessing attack In our threat model, an attacker $\mathcal{A}$ can retrieve the stored secrets of a sensor node via side-channel attacks. Within the time frame of the sessions of a patient in a hospital that could range from hours to weeks, $\mathcal{A}$ should not be able to guess the identifier ID_p or PIN_p via the values $H(ID_p)$ and B_s where $B_s = A_s \oplus \mathsf{H}(y_g)$ and $y_g = \mathsf{H}(\mathsf{H}(PIN_p \oplus b)\|ID_p)$.

Anonymity and unlinkability The identity of the patient ID_p is masked in our scheme via the one-way hash function. For an attacker $\mathcal{A}$, it should be computationally infeasible to compute it. Furthermore, unlinkability is also provided via the freshness of random values and timestamps. Suppose $\mathcal{A}$ can intercept messages from the sensor node and the patient's smartphone. Using only an eavesdropping attack, the attacker should not be able to compute $B_s, \mathsf{H}(ID_p), \mathsf{H}(A_s)$, or $HREG$ from one of the message parameters within polynomial time.

Perfect forward secrecy Assume an attacker $\mathcal{A}$ compromises the session key sk, the sensor node's secret and public identity $\mathsf{H}(A_s)$, B_s and the current hash key hk_s^i, and the patient's link $HREG$. These keys do not reveal any previous session keys as the session keys are calculated using the formula $\mathsf{sk} = \mathsf{H}(R_1\|R_2\|T_1\|T_2\|B_s\|\mathsf{H}(ID_p)\|\mathsf{hk}_s^i)$. The previous hash key hk_s^{i-1} is not computable, since, during each session establishment the hash key is updated via $\mathsf{hk}_s^i = \mathsf{H}(\mathsf{hk}_s^{i-1})$. Moreover, the freshness of the session keys is ensured through the use of the timestamps, and the random values.

5.3 Comparison to related work

An overview of the security features that our work and the considered related work provide is presented in Table 2. Only our scheme covers all the required security features. Gupta et al. [4] and Shuai et al. [3] do not provide distributed key agreement, and, Gupta et al. [4], Kumar et al. [2], and Chen et al. [5] do not provide sufficient perfect forward secrecy protection. Furthermore, if we take the requirement to input an identifier and a password into account (R.6), our scheme and that of Gupta et al. and Shuai et al. [3] would only be eligible, because, the schemes of Kumar et al. [2] and Chen et al. [5] do not provide this feature. The TLS protocol using a Pre-Shared-Key (PSK) configuration, as analysed by Winderickx et al. [6], can only provide mutual authentication, protection against general attacks, and distributed authentication. It is lacking all or some of the other requirements in comparison to respectively ours and the other considered schemes.

Table 2 Comparison of the available security features. The security features R.1-R.5 are described in Section 5 (R.1: Anonymity and unlinkability, R.2: Mutual authentication, R.3: Perfect forward secrecy, R.4: Protection against general attacks, and R.5: Distributed authentication, and R.6: two-factor authentication)

Security features	R.1	R.2	R.3	R.4	R.5	R.6
Kumar [2]	Y	Y	N	Y	Y	N
Gupta [4]	Y	Y	N	Y	N	Y
Chen [5]	Y	Y	N	Y	Y	N
Shuai [3]	Y	Y	Y	Y	N	Y
TLSv1.2-PSK [6]	N	Y	N	Y	Y	N
Ours	Y	Y	Y	Y	Y	Y

6 Implementation results

The implementation results are generated using a generic and an empirical study. The computationally expensive operations are identified and generalised for each entity of our scheme and related key agreement protocols. Furthermore, the communication impact is compared using the amount of messages and bits that are exchanged. Next, the impact of the proposed scheme is validated empirically through a proof-of-concept implementation.

The results of the pre-shared-key configuration of the TLS protocol that were generated by Winderickx et al. [6] did not include the computation impact using our format. Therefore, we analysed the cryptographic operations that are used in a mbed TLS based implementation of a TCP server and TCP client using TLS version 1.2 with the TLS-PSK-WITH-AES-128-CCM and the TLS-PSK-WITH-AES-128-GCM-SHA256 ciphersuite configuration. The computation impact results are based on the reported module initiation of cryptographic operations like the SHA256 hash operation and the reported actions as indicated in the debug logs during the handshake procedure. The other ciphersuite configurations analysed by Winderickx et al. were not considered, because they use public-key operations that are at least a factor of 30 times more computationally expensive as demonstrated by the authors [6, 13].

The computation impact of the key agreement protocol is compared to related work in Table 3 in terms of the number of hash functions (Th), the number of symmetric-key encryptions (Te), the number of symmetric-key decryptions (Td), and the number of fuzzy key extractions (Tfe). Our scheme requires five and nine hash operations for respectively the sensor node and the patient. Also, our scheme does not require input of the RC during key agreement. In total, our scheme ranks third in terms of performance. Only Chen et al. [5] and Kumar et al. [2] feature less operations. Though, our scheme additionally provides Perfect forward secrecy and two-factor authentication in comparison to these two schemes while only requiring respectively one or five more hash computations.

The communication impact compared to related work is given in Table 4. We have chosen for the following message parameter sizes: 24-byte ID, 32-byte hash digest (SHA256), 32-byte random, 4-byte timestamp (Unix epoch), and 4-byte counter. The results of Kumar et al. [2] and Chen et al. [5] were estimated using the above mentioned message parameter sizes because the original resulting value was either not available or had inconsistent message parameter sizes. Our scheme features the lowest amount of bits exchanged. Furthermore, only two messages rounds are required, which is equal for the protocol of Kumar et al. [2] and Chen et al. [5].

Table 3 Comparison of the computation impact of the key agreement scheme with other relevant state-of-the-art schemes (Th: hash, Te: symmetric encryption, Td: symmetric decryption, and Tfe: fuzzy extraction)

Protocol	Sensor	User	RC	Total
Kumar [2]	2Th + 1Te + 1Td	2Th + 1Td + 2Te	-	4Th + 3Te + 2Td
Gupta [4]	4Th	7Th	5Th	16Th
Chen [5]	5Th	8Th	-	13Th
Shuai [3]	7Th	Tfe + 11Th	12Th	Tfe + 30Th
TLSv1.2-PSK	7Th + Te + Td	7Th + Te + Td	-	14Th + 2Te + 2Td
Ours	5Th	9Th	-	14Th

Table 4 Comparison of the communication impact of the key agreement scheme with other relevant state-of-the-art schemes (*: estimated values)

Protocol	Rounds	Total size (bits)
Kumar [2]	2	1568*
Gupta [4]	5	3808
Chen [5]	2	2080*
Shuai [3]	5	1824
TLSv1.2-PSK [6]	4	4792
Ours	2	1376

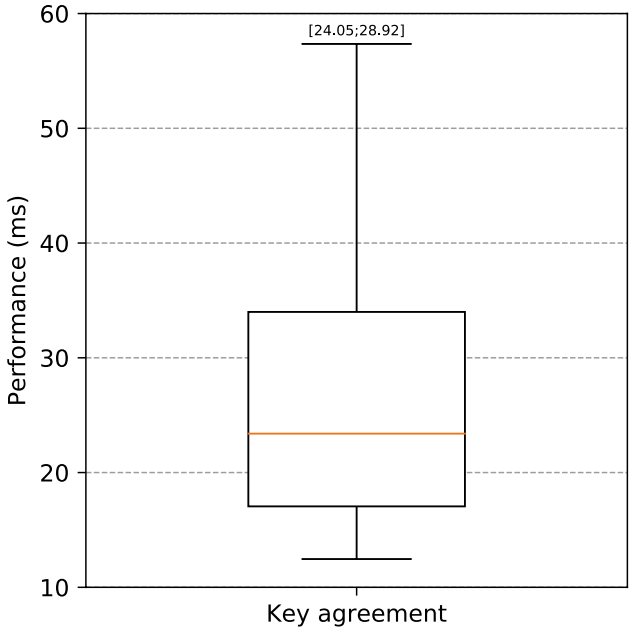

Fig. 7 Performance box plot of Action 4 based on 75 runs. The box plot indicates the distribution of the data, i.e. divided into a quartile. The orange line represents the median result, and the two numbers inside parentheses show the 96% confidence interval

An empirical study of our proposed scheme is done using a proof-of-concept implementation on a custom-designed board featuring a TI MSP432P4011 MCU [11] and a TI CC3120 network processor [12]. The MCU is a 32-Bit Arm Cortex-M4F running at 48 Mhz with 2 MB of Flash and 256 KB of RAM. The code is compiled using the GNU GCC compiler tools version 7.2.1 with the -O3 optimisation setting. The performance of Action 4 is measured on this health sensor node by measuring the total duration of Action 4 using the available 24-bit system timer, and, 75 iterations were performed to increase the confidence on the result. The results are presented using a box plot and a 96% confidence interval inside parentheses in Fig. 7. On average, Action 4 takes about 26.5 *ms*. Using the power consumption of the microcontroller and the network interface, the energy usage is estimated to be about 507 μJ. This is considerably lower than the estimated energy consumption of the TLS protocol based implementation of Winderickx et al. [6], which ranged from 1.5 *mJ* to 147 *mJ*. Also, the energy consumption of the key establishment of our proposed scheme is negligible in comparison to the energy required to perform the measurements and communication of sensor data in one 10 *s* period, which was estimated to be around 698.2 *mJ* by Winderickx et al. [6].

7 Conclusion

This work proposes the design and implementation of a symmetric-key security protocol for wearable healthcare devices. Both the association phase, in which the wearable device is anonymously linked to the patient, and the communication phase, in which end-to-end secured data exchange takes place, are described in detail. An analysis of the security features and the communication and computation cost of the protocol shows that our solution outperforms previously proposed protocols. In terms of future work, one additional security feature is worth considering, namely the protection against privileged insider attacks, in which the adversary can determine the identity of the sensor nodes that are linked to a patient.

Funding This work was funded by the WearIT4Health project which is carried out under Interreg V-A Euregio Meuse-Rhine and is supported by the European Union and the European Regional Development Fund and with financial support of the province of Limburg - Belgium. This work was also supported by CyberSecurity Research Flanders with reference number VR20192203.

Declarations

Conflicts of interest The authors declare that they have no conflict of interest.

References

1. Parliament E. Regulation (eu) 2016/679 of the european parliament and of the council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation). Off J Eur Union. 2016;(679):88. https://eur-lex.europa.eu/eli/reg/2016/679/oj.
2. Kumar P, Braeken A, Gurtov A, Iinatti J, Ha PH. Anonymous Secure Framework in Connected Smart Home Environments. IEEE Trans Inf Forensics Secur. 2017;12(4):968–79. https://doi.org/10.1109/TIFS.2016.2647225. http://ieeexplore.ieee.org/document/7803595/.
3. M. Shuai, B. Liu, N. Yu, and L. Xiong. Lightweight and Secure Three-Factor Authentication Scheme for Remote Patient Monitoring Using On-Body Wireless Networks. Security and Communication Networks, 2019:1–14, June 2019. https://doi.org/10.1155/2019/8145087. https://www.hindawi.com/journals/scn/2019/8145087.
4. Gupta A, Tripathi M, Shaikh TJ, Sharma A. A lightweight anonymous user authentication and key establishment scheme for wearable devices. Comput Netw. 2019;149:29–422. https://doi.org/10.1016/j.comnet.2018.11.021. https://doi.org/10.1016/j.comnet.2018.11.021.
5. Chen CM, Xiang B, Wu T-Y, Wang K-H. An Anonymous Mutual Authenticated Key Agreement Scheme for Wearable Sensors in Wireless Body Area Networks. Appl Sci. 2018;8(7):1074. https://doi.org/10.3390/app8071074. http://www.mdpi.com/2076-3417/8/7/1074.

6. Winderickx J, Bellier P, Duflot P, Coppieters D, and Mentens N. WiP: Communication and security trade-offs for wearable medical sensor systems in hospitals. In Proceedings of The ACM SIGBED International Conference on Embedded Software (EMSOFT), New York, NY, USA, ACM. 2019. p. 2.
7. Dolev D, Yao A. On the security of public key protocols. IEEE Trans Inf Theory. 1983;29(2):198–208. https://doi.org/10.1109/TIT.1983.1056650. http://ieeexplore.ieee.org/document/1056650/.
8. Braeken A, Liyanage M, Kumar P, Murphy J. Novel 5G Authentication Protocol to Improve the Resistance Against Active Attacks and Malicious Serving Networks. IEEE Access. 2019;7:64040–522. https://doi.org/10.1109/ACCESS.2019.2914941. https://ieeexplore.ieee.org/document/8706883/.
9. Rubin A, Honeyman P, Nonmonotonic cryptographic protocols. In Proceedings The Computer Security Foundations Workshop VII, pp. 100–116, Franconia, NH, USA, IEEE Comput. Soc: Press; 1994. http://ieeexplore.ieee.org/document/315943/.
10. Xu Y, and Xie X. Analysis of Authentication Protocols Based on Rubin Logic. In 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing, Dalian, China, IEEE, 2008:1-5. https://doi.org/10.1109/WiCom.2008.1120. http://ieeexplore.ieee.org/document/4679028/.
11. Texas Instruments Incorporated. Msp432p4011: Simplelink ultra-low-power 32-bit arm cortex-m4f mcu with precision adc, 2mb flash and 256kb ram, 2019. http://www.ti.com/product/MSP432P4011.
12. Texas Instruments. CC3120 SimpleLink Wi-Fi Wireless Network Processor, Internet-of-Things Solution for MCU Applications, 2017. http://www.ti.com/lit/ds/swas034/swas034.pdf.
13. Winderickx J. Energy-efficient and secure implementations for the IoT, 2020. https://lirias.kuleuven.be/retrieve/567362.