Wireless Personal Communications
https://doi.org/10.1007/s11277-022-09844-2

ISAKA: Improved Secure Authentication and Key Agreement protocol for WBAN

Javad Alizadeh[1] · **Masoumeh Safkhani**[2,3] · **Amir Allahdadi**[1]

Accepted: 28 May 2022

Abstract
Internet of Things (IoT) is a revolution which has influenced the lifestyle of human. Wireless Body Area Networks (WBAN)s are IoT-based applications which have a crucial role in the current healthcare systems. A WBAN is used to collect some health-related information of patients and transport and monitor them in a healthcare system. This information is crucial in the sense of the patient's life. Then the privacy of the patient and the security of his/her information are some main challenges in the WBAN. Another challenge in the WBAN is the resources limitation of the sensor nodes. This limitation imposes that a suitable scheme for the WBAN should be a lightweight one. In order to response these challenges, several lightweight Authentication and Key Agreement (AKA) schemes have been presented for WBAN so far. However, approximately none of them could reach their security and cost goals. In 2020, Narwal and Mohapatra proposed a claimed to be secure lightweight AKA protocol for WBAN named SEEMAKA. In this paper, we show that this scheme suffers from attacks including sensor node traceability, disclosure of the secret parameters of the sensor nodes and master nodes, sensor node impersonation, extracting the session key, and Denial of Service attacks. Besides that, we focus to overcome these vulnerabilities and present an improved version of SEEMAKA named ISAKA. ISAKA improves the security level and also the efficiency level of SEEMAKA. More precisely, ISAKA is safe against mentioned attacks and it improves ROM and RAM storage requirements and also computational and communication costs. We prove the security of ISAKA using two formal methods, i.e. BAN logic method and ProVerif tool.

Keywords IoT · WBAN · AKA · Sensor node traceability attack · Impersonation attack · BAN logic · ProVerif tool

✉ Masoumeh Safkhani
Safkhani@sru.ac.ir

Javad Alizadeh
Jaalizadeh@ihu.ac.ir

Amir Allahdadi
Aallahdad@ihu.ac.ir

[1] Fath Center, Faculty and Research Center of Computer, Imam Hossein University, Tehran, Iran

[2] Computer Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran

[3] Institute for Research in Fundamental Sciences (IPM), Tehran, Iran

Published online: 26 June 2022

1 Introduction

Connecting objects to the Internet and adding "sensors" to them adds a level of digital intelligence to them. This intelligence allows objects to interact with other objects using real-time data without human intervention. The Internet of Things (IoT) makes the world around us smarter and more responsive, and integrates the digital and physical worlds

Wireless body area networks (WBAN)s are IoT based technologies which adopted in the medical and healthcare systems. In the WBAN, a patient uses some sensor nodes in or around of his/her body to care his/her health. The sensors are either wearable ones or implementable ones in the patient's body. These sensors collect health-related information of the patient, such as the blood pressure and heart rate and send them to an electronic healthcare system. For this reason, the sensors use of some devices such as tablets, smart phones, and smart watches as the relays. These devices were named as the first-level sensor nodes in [15]. After receiving the sensors' information, these smart devices transmit them to a remote healthcare server through Internet. A doctor can control the health of his/her patient using information gathered in the healthcare server. The doctor also could send some necessary instructions to his/her patient through Internet and therefore there is no need to visit him/her. The WBAN model described above, is represented in Figure 1.

The information sent from the patient to the healthcare systems are important and critical. Since this information is related to the patient health and can influence his/her life, it is necessary to establish their confidentiality. An adversary wants to eavesdrop the transmitted messages between the patient and the healthcare system. For this goal, the adversary tries to impersonate the healthcare system. Then in a WBAN, it is necessary the participants authenticate each other and establish a confidential key to protect their confidential information in the public channel. Besides of these conditions, the anonymity of the patient should be satisfied in the WBAN, too. Authentication and key agreement (AKA) protocols are the cryptographic schemes that their goals are the providence of the security in the communications like in the WBAN. Some of these protocols could prevent the traceability of participants and provide their anonymity. A secure AKA protocol for WBAN should be secure against various attacks, such as participant's traceability, participant's impersonation, disclosure of the secret parameters of the participants, extracting the session key, and

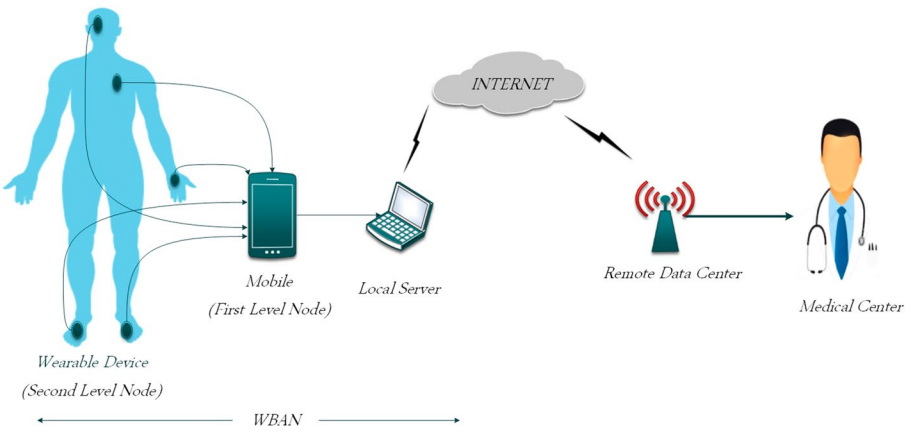

Fig. 1 A general model of WBAN

denial of service attacks. On the other hand, there are some limitations in the power consumption and required implementation resources in the WBAN field which imposes the use of lightweight AKA protocols [20].

Recently, Narwal and Mohapatra in [15] presented a new AKA scheme for two-tier WBAN topology. This scheme is categorized as a lightweight scheme because uses only hash functions and XOR operations and therefore is suitable for resource constrained devices. Furthermore, this scheme is an energy efficient one and mutually authenticates the sensor nodes with the master nodes anonymously and establishes a session key between them. The security of SEEMAKA was discussed in [15] using informal and also formal methods. Although the designers of SEEMAKA claimed its security against a vast list of known attacks, however, we show some security pitfalls for this scheme. We have shown that SEEMAKA does not provide the anonymity and a sensor node can be traceable when it uses the scheme. Moreover, we have proved that this scheme suffers from sensor node impersonation, disclosure of the secret parameters of the sensor nodes and master nodes, extracting the session key, and Denial of Service (DoS) attacks. In order to enhance the security of SEEMAKA, we also propose an improved version of this protocol named ISAKA. We also have shown that ISAKA reaches a higher security and efficiency compared to SEEMAKA. Focusing on the efficiency of ISAKA, we find that it uses only nine XORs and eight calls to hash function in the mutual authentication and key agreement phase. ISAKA was also able to reduce ROM and RAM storage and communication costs when compared to SEEMAKA. In order to prove the security of ISAKA we used formal methods such as Barrows–Abadi–Needham (BAN) logic [7] and ProVerif tool [6].

The remainder of the paper is structured as follows. Some related works in the field of AKA schemes for WBAN are introduced in Section 2. A brief review of SEEMAKA and the cryptanalysis of this scheme are presented in Sects. 3 and 4 respectively. Section 5 describes the scheme ISAKA as an improvement of SEEMAKA. The informal analysis and formal security proof of ISAKA based on the BAN logic and also using the ProVerif tool are explained in Sect. 6. Section 7 compares the security, ROM and RAM storage requirements, communication cost, computational cost, and energy dissipation of ISAKA with some similar protocols, especially SEEMAKA. Finally, Sect. 8 concludes the paper.

2 Related Works

The wireless sensor network (WSN) technology firstly used by Zimmerman [23] in the body area network. Nowadays, there are various AKA schemes which discussed about the efficiency and security requirements of WBAN. Some of the schemes are using only cryptographic hash functions, some of them are using symmetric encryption algorithms and some other are using asymmetric encryption algorithms such as the algorithms based on the elliptic curves. In this section, recent related schemes focused on the security of WBAN will be briefly reviewed.

Although, some AKA schemes for WBAN had been proposed before 2016 [8] but the first attempt to include the anonymity of the patients in a two-tier WBAN relates to Ibrahim et al. in 2016 [11]. In their protocol, the identities of the sensor nodes were not used in the protocol immediately and some identity related information of the sensor nodes were used. The Ibrahim et al.'s scheme can be considered as a lightweight one, since this scheme is designed only using simple XOR and hash operations.

In 2017, Li et al. [14] proposed another lightweight AKA protocol for WBAN. They tried to improve the energy consumption of previous AKA protocols of WBAN.

In 2018, Agha et al. [1] presented an authentication and communication suite in WBAN for IoT-based applications. Their studies include a security suite based on the KBS key management with hashing.

In 2019, Ostad-Sharif et al. [19] studied the security of [14]'s scheme and found some weaknesses of it. They showed that Li et al.'s scheme was vulnerable to the wrong session key agreement and desynchronization attacks. They also presented a lightweight AKA protocol with anonymity capability. In 2019, Ostad-Sharif et al. considered the security of the schemes presented in [4, 5, 9] and showed that all of them suffer from key compromise impersonation attack. They also proposed a novel user authentication protocol that compared to the mentioned schemes was secure. In this year, moreover Ostad-Sharif et al. [18] presented an AKA scheme for healthcare applications based on the elliptic curve cryptography. This scheme was analysed by Nikooghadam and Amintoosi in [17] which in was shown that it is vulnerable to key compromise password guessing attacks and key compromise impersonation attacks. They also presented an improved AKA scheme for healthcare applications.

Xu et al. in [22] presented a lightweight anonymous AKA scheme for WBAN. Alzahrani et al. [3] examined the security of this scheme and showed that it does not resist against some attacks, such as replay and key compromise impersonation attacks. Then they presented an improved AKA scheme for WBAN.

In 2020, Fotouhi et al. [8] proposed a lightweight hash-chain based authentication scheme for WBAN.

In 2021, Alzahrani [2] examined the security of Karthegaveni et al.'s protocol [12] and showed that this protocol is vulnerable against some attacks, such as DoS and replay attacks. He also presented a lightweight protocol based on the symmetric key cryptographic primitives. Syed Jawad Hussain et al. in [10] analysed the computational cost of some authentication schemes preserving anonymity of users in WBAN and compared them. They also introduced a new lightweight scheme to authenticate in the wearable things. Comparing to the schemes that were considered in [10], the new one had less cost and provided security and privacy. In another work, Soni and Singh [21] designed a lightweight AKA scheme where they focused on the computational cost of the scheme, similarly to [10]. Compared to some previous AKA schemes, they showed that their scheme needs less time, power, and also execution cost. Narwal and Mohapatra in [16] presented another authentication scheme with the anonymity capability in the field of WBAN in 2021 which the security of their scheme is proved in the Real-or-Random Model.

In this regard, we analyse the proposed protocol for WBAN by Narwal and Mohapatra in 2020 and show its advantages and disadvantages. This paper and the like contribute to the science of security protocols' design as they help designers become familiar with a variety of innovative attacks scenarios and avoid repeating the errors that lead to these attacks.

3 SEEMAKA

SEEMAKA is an energy-efficient mutual authentication and key agreement (AKA) protocol for two tier Wireless Body Area Networks (WBAN) introduced by Narwal and Mohapatra [15] in 2020. As depicted in Figure 2, the network model considered in

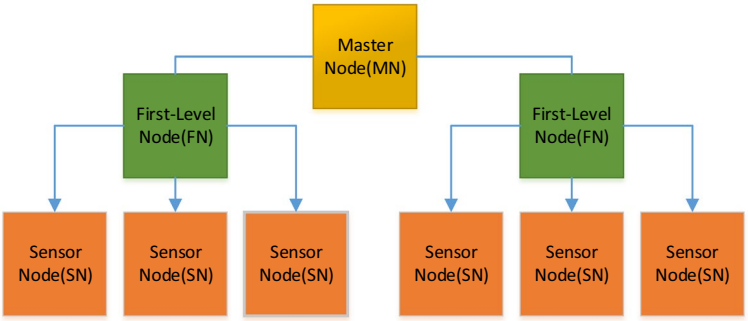

Fig. 2 The network model used in the specification of SEEMAKA

SEEMAKA has a two hop/ two tier centralized network topology where three kinds of nodes, Master Node (*MN*), First-Level Nodes (*FN*) and Second-Level Sensor Nodes (*SN*) are used. In fact, SEEMAKA scheme is an AKA protocol between *SN* and *MN* where their communications are done with the help of *FN*. Through describing SEEMAKA protocol, the notations represented in Table 1 are used which are similar to the notations introduced in [15].

Table 1 List of notations

Notation	Description
$ADMIN$	The administrator of system
SN	Second-level sensor node requesting authentication
MN	Master node/hub node
FN	First-level sensor node
PID_{SN}	SN constant identification number in SEEMAKA and SN dynamic identification number in ISAKA
PID_{FN}	FN constant identification number
K_{MNSN}	MN's shared secret key with SN in SEEMAKA and a MN's secret key related to SN which is unique for each SN in ISAKA
X_{MN}	MN's master or private key
X_{SN}	SN 's temporary secret key
TP_{SN}, PTP_{SN}	Authentication variables stored in SN's memory
B_{SN}, C_{SN}, SP_{SN}	Auxiliary variables
$Alpha$, $Beta$, $Gamma$, Eta, Mu	Authentication variables
r_{SN}, y, r_{MN}	Random nonces
t_{SN}	Timestamp generated by SN
$\oplus$	XOR operation
$A \rightarrow B : C$	A sends message C to B through a passive channel
$Skey$	Established session key
(g, q)	Concatenation of g and q
$h(.)$	Hash function

SEEMAKA protocol comprises four main phases including setup, registration, mutual authentication and key agreement, and master key update phases. In the following, we briefly review them.

3.1 Setup Phase

In this phase, *ADMIN*:

- generates a shared secret key for *MN* and *SN* named K_{MNSN} and saves it in both memories of *MN* and *SN*.
- chooses a secret key X_{MN} for *MN* and saves it in its memory.
- finally assigns an identity PID_{SN} for *SN*.

3.2 Registration Phase

In this phase, *ADMIN*:

- registers the *SN*.
- chooses a temporary secret key named X_{SN} and assigns it for *SN*.
- assigns a secret constant identification number PID_{FN} to *FN*.
- computes $TP_{SN} = h(K_{MNSN}, X_{MN}) \oplus PID_{SN}$ and $PTP_{SN} = TP_{SN} \oplus X_{MN} \oplus X_{SN}$ and saves the tuples $< PID_{SN}, TP_{SN}, PTP_{SN}, K_{MNSN} >$ and $< PID_{FN}, PID_{SN}, TP_{SN}, PTP_{SN} >$ in *SN*'s and *FN*'s memories, respectively.

3.3 Mutual Authentication and Key Agreement Phase

In the authentication and key agreement phase of the protocol, *SN* and *MN* mutually authenticate each other and agree on a session key, named *Skey*. In this phase, *FN* helps the communications between the *SN* and *MN* as a relay. This phase is depicted in Figure 3 and is described as follows:

1. *SN* selects a random nonce r_{SN} and generates the current time as t_{SN}. Then using PID_{SN}, TP_{SN}, and PTP_{SN}, it calculates $B_{SN} = PID_{SN} \oplus TP_{SN}$, $C_{SN} = r_{SN} \oplus B_{SN}$, and $SP_{SN} = h(r_{SN}, t_{SN}, PID_{SN}, C_{SN}) \oplus PTP_{SN}$. After that *SN* sends the message $M_1' =< SP_{SN}, t_{SN}, C_{SN}, TP_{SN}, PTP_{SN} >$ to *FN*.
2. *FN* adds its identity i.e. PID_{FN}, to the message M_1' and sends the message $M_1 =< SP_{SN}, t_{SN}, C_{SN}, TP_{SN}, PTP_{SN}, PID_{FN} >$ to *MN*.
3. Once received M_1, *MN* using its local values of PID_{FN} and X_{MN} checks the validity of the received PID_{FN} and the time t_{SN}. Then it:
   - calculates $PID^*_{SN} = TP_{SN} \oplus h(K_{MNSN}, X_{MN})$, $B^*_{SN} = PID^*_{SN} \oplus TP_{SN}$, $r^*_{SN} = B^*_{SN} \oplus C_{SN}$ and $SP^*_{SN} = h(r^*_{SN}, t_{SN}, PID^*_{SN}, C_{SN}) \oplus PTP_{SN}$. Then it considers whether the received SP_{SN} of the message M_1 is equal to the calculated SP^*_{SN} or not. It worth noting that in SEEMAKA, this step must be repeated for all sensors stored in the *MN* to find $SP^*_{SN} = SP_{SN}$, which multiplies the processing time and number of operations by the number of sensors stored in the *MN*.
   - If so, it chooses a new X^+_{SN} and generates the random nonce y. Afterwards, it computes $Alpha = SP_{SN} \oplus y$, $Beta = r^*_{SN} \oplus Alpha$, $TP^+_{SN} = h(K_{MNSN}, X_{MN}, X^+_{SN}, PID^*_{SN})$,

SN	FN	MN
$< PID_{SN}, TP_{SN}, PTP_{SN}, K_{MNSN} >$	$< PID_{FN} >$	$< PID_{FN}, X_{MN}, K_{MNSN} >$
Chooses the random r_{SN},		
Considers the time t_{SN},		
$B_{SN} = PID_{SN} \oplus TP_{SN}$,		
$C_{SN} = r_{SN} \oplus B_{SN}$,		
$SP_{SN} = h(r_{SN}, t_{SN}, PID_{SN}, C_{SN})$		
$\oplus PTP_{SN}$.		
$M_1' = \{SP_{SN}, t_{SN}, C_{SN}, TP_{SN},$		
$PTP_{SN}\} \longrightarrow$		
	$M_1 = \{SP_{SN}, t_{SN}, C_{SN}, TP_{SN},$	
	$PTP_{SN}, PID_{FN}\} \longrightarrow$	
		Checks that PID_{FN} exists,
		Checks validity of t_{SN},
		$PID^*_{SN} = TP_{SN} \oplus h(K_{MNSN}, X_{MN})$,
		$B^*_{SN} = PID^*_{SN} \oplus TP_{SN}$,
		$r^*_{SN} = B^*_{SN} \oplus C_{SN}$,
		$SP^*_{SN} = h(r^*_{SN}, t_{SN}, PID^*_{SN}, C_{SN})$
		$\oplus PTP_{SN}$,
		Checks $SP^*_{SN} \stackrel{?}{==} SP_{SN}$,
		Chooses the random y
		and X^+_{SN},
		$Alpha = SP_{SN} \oplus y$,
		$Beta = r^*_{SN} \oplus Alpha$,
		$TP^+_{SN} = h(K_{MNSN}, X_{MN}, X^+_{SN},$
		$PID^*_{SN})$,
		$PTP^+_{SN} = TP^+_{SN} \oplus X_{MN} \oplus X^+_{SN}$,
		$Gamma = h(Alpha, Beta, SP_{SN}, y,$
		$r^*_{SN}, K_{MNSN})$,
		$Eta = Gamma \oplus TP^+_{SN}$,
		$Mu = Gamma \oplus PTP^+_{SN}$,
		$Skey = h(B^*_{SN}, r^*_{SN}, y, PID^*_{SN})$.
		$\longleftarrow M_2' = \{Alpha, Beta, Gamma,$
		$Eta, Mu, PID_{FN}\}$
	$\longleftarrow M_2 = \{Alpha, Beta, Gamma,$	
	$Eta, Mu\}$	
$y^* = Alpha \oplus SP_{SN}$,		
$r^*_{SN} = Beta \oplus y^*$,		
$Gamma^* = h(Alpha, Beta, SP_{SN}, y^*,$		
$r^*_{SN}, K_{MNSN})$,		
Checks $Gamma^* \stackrel{?}{==} Gamma$,		
$TP^+_{SN} = Gamma \oplus Eta$,		
$PTP^+_{SN} = Gamma \oplus Mu$,		
$Skey = h(B_{SN}, r^*_{SN}, y^*, PID_{SN})$		
Updates TP_{SN} and PTP_{SN}		
with TP^+_{SN} and PTP^+_{SN}		

Fig. 3 The authentication and key agreement procedure of SEEMAKA

$PTP^+_{SN} = TP^+_{SN} \oplus X_{MN} \oplus X^+_{SN}$, $Gamma = h(Alpha, Beta, SP_{SN}, y, r^*_{SN}, K_{MNSN})$, $Eta = Gamma \oplus TP^+_{SN}$, and $Mu = Gamma \oplus PTP^+_{SN}$.

- Finally, calculates the session key as $Skey = h(B^*_{SN}, r^*_{SN}, y, PID^*_{SN})$ and stores it in its memory. Then it sends the message $M_2' =< Alpha, Beta, Gamma, Eta, Mu, PID_{FN} >$ to FN.

4. FN deletes its identity i.e. PID_{FN}, of the message M_2' and sends the message $M_2 =< Alpha, Beta, Gamma, Eta, Mu >$ to SN.
5. Once SN received the message M_2, it firstly calculates $y^* = Alpha \oplus SP_{SN}$, $r^*_{SN} = Beta \oplus y^*$ and $Gamma^* = h(Alpha, Beta, SP_{SN}, y^*, r^*_{SN}, K_{MNSN})$. Then it checks whether the received $Gamma$ of message M_2 is equal to the calculated $Gamma^*$ or not. If so, it calculates $TP^+_{SN} = Gamma \oplus Eta$ and $PTP^+_{SN} = Gamma \oplus Mu$ and computes

and stores the session key as $Skey = h(B_{SN}, r^*_{SN}, y^*, PID_{SN})$. At the end, SN updates the amount of TP_{SN} and PTP_{SN} in its memory with TP^+_{SN} and PTP^+_{SN}, respectively.

3.4 Master Key Update Phase

The designers of SEEMAKA suggested two different ways to update the master key of master node i.e. (X_{MN}). In the first approach the master key is updated after that all sensor nodes are deployed. In the second approach, it was assumed that the master node has two kinds private key: fixed private key and dynamic private key. The fixed private key is used in the first run of the protocol and then the dynamic private key is used to update TP^+_{SN} and PTP^+_{SN}. The dynamic private key is updated as $X^+_{MN} = X_{MN} \oplus y \oplus X^+_{SN}$.

4 SEEMAKA Cryptanalysis

In this section, we examine the security pitfalls of SEEMAKA. Precisely, we show that this protocol is insecure against the sensor node traceability, sensor node impersonation, disclosure of the secret parameters of the sensor nodes and master nodes, and extracting the session key between the sensor nodes and master nodes attacks.

4.1 Sensor Node Traceability

In this section, we show that SEEMAKA is not safe against traceability of the sensor node attack and it could not achieve the anonymity property. To trace the sensor node SN, it is enough the adversary $\mathcal{A}$ acts as follows:

- Eavesdrops one session of protocol and stores transferred messages including M'_1, M_1, M'_2 and M_2.
- Using eavesdropped $Alpha$ and $Beta$ from message M_2, retrieves r_{SN} as $Alpha \oplus Beta$.
- Using eavesdropped C_{SN} from the message M_1, extracts B_{SN} as $C_{SN} \oplus r_{SN}$.
- Using eavesdropped TP_{SN} from the message M_1, extracts PID_{SN} as $B_{SN} \oplus TP_{SN}$.

Following above attack, the adversary $\mathcal{A}$ can find PID_{SN}, the fixed identity of SN, who wants to communicate with the MN and follows its communications. This attack succeeds with the probability of "1" and only one session of protocol's eavesdropping.

4.2 Disclosure of the Sensor Nodes and Master Nodes Secret Parameters

Here, we present an approach to extract the secret key X_{SN} of the sensor node SN and also the secret key X_{MN} of the master node MN when the adversary is a malicious master node and a malicious sensor node, respectively. Given the registration phase, the malicious master node/(sensor node) knows that $PTP_{SN} = TP_{SN} \oplus X_{MN} \oplus X_{SN}$. The node can get the amount of TP_{SN} and PTP_{SN} in the registration phase or from the message M_1. Then the malicious master node retreives the secret key of the sensor node SN i.e. X_{SN} as $TP_{SN} \oplus X_{MN} \oplus PTP_{SN}$. Similarly, the malicious sensor node extracts the secret key of the master node MN i.e. X_{MN} as $TP_{SN} \oplus X_{SN} \oplus PTP_{SN}$.

4.3 Sensor Node Impersonation

In this section, we prove that the adversary $\mathcal{A}$ can impersonate the sensor node SN in SEEMAKA.

As mentioned before in Section 4.1, $\mathcal{A}$ can retrieve B_{SN} and so finds the amount of $h(K_{MNSN}, X_{MN})$. Now $\mathcal{A}$ for impersonating SN, it is enough does as follows:

- Eavesdrops one session of protocol between the target sensor node and master node and finds the identity of the sensor node SN i.e. PID_{SN}, and B_{SN} following the method described in Section 4.1.
- Calculates $TP_{SN} = h(K_{MNSN}, X_{MN}) \oplus PID_{SN}$ as $B_{SN} \oplus PID_{SN}$.
- Gets the secret keys X_{MN} and X_{SN} using the method described in Section 4.2.
- Calculates PTP_{SN} as $TP_{SN} \oplus X_{MN} \oplus X_{SN}$.
- Chooses the random nonce $r_{\mathcal{A}}$ instead of the nonce r_{SN}, considers the current time $t_{\mathcal{A}}$ and using B_{SN} calculates $C'_{SN} = r_{\mathcal{A}} \oplus B_{SN}$, and $SP'_{SN} = h(r_{\mathcal{A}}, t_{\mathcal{A}}, PID_{SN}, C'_{SN}) \oplus PTP_{SN}$.
- Sends the message $M'_1 =< SP'_{SN}, t_{\mathcal{A}}, C'_{SN}, TP_{SN}, PTP_{SN} >$ to FN.
- FN adds its identity i.e. PID_{FN}, to the message M'_1 and sends the message $M_1 =< SP'_{SN}, t_{\mathcal{A}}, C'_{SN}, TP_{SN}, PTP_{SN}, PID_{FN} >$ to MN.
- Once received M_1, MN using its local values of PID_{FN} and X_{MN} checks the validity of the received PID_{FN} and the time $t_{\mathcal{A}}$ which are correct. Then it for each registered sensor node K_{MNSN}:
  - calculates $PID^*_{SN} = TP_{SN} \oplus h(K_{MNSN}, X_{MN}), B^*_{SN} = PID^*_{SN} \oplus TP_{SN}, r^*_{SN} = B^*_{SN} \oplus C'_{SN} = r_{\mathcal{A}}$ and $SP^*_{SN} = h(r^*_{SN}, t_{SN}, PID^*_{SN}, C'_{SN}) \oplus PTP_{SN} = h(r_{\mathcal{A}}, t_{SN}, PID^*_{SN}, C'_{SN}) \oplus PTP_{SN} = SP'_{SN}$.
  - considers whether the received SP'_{SN} of the message M_1 is equal to the calculated SP^*_{SN} or not which it holds.
  - so, authenticates the adversary as a legitimate sensor node.

  The success probability of the above attack equals to "1" and its complexity is one run of protocol's eavesdropping and as well as the adversary's cooperation with a malicious master node and a malicious sensor node.

4.4 Extracting the Session Key

The adversary $\mathcal{A}$ can extract the established session key of SEEMAKA as follows:

- Eavesdrops one session of protocol and stores transferred messages including M'_1, M_1, M'_2 and M_2.
- Finds SP_{SN} from the message M_1 and $Alpha$ from the message M_2. She/he calculates y as $Alpha \oplus SP_{SN}$.
- Finds B_{SN}, r_{SN}, and PID_{SN} following the method described in Section 4.1.
- Calculates the session key $Skey$ as $h(B_{SN}, r_{SN}, y, PID_{SN})$.

  It was observed that according to the above attack scenario, the attacker can achieve the created common key just by eavesdropping of a session and performing a number of simple calculations with the success probability of "1".

4.5 Denial of Service Attack

SEEMAKA is vulnerable against Denial of Service (DoS) attack. Although *Eta* and *Mu* are computed in the *MN*'s side and are used to update the shared secrets of *SN*'s side, but their integrity is not guaranteed in this protocol. So, the adversary $\mathcal{A}$ could modify *Eta* and *Mu* in the message M_2. After that the *SN* received the modified message M_2, it updates TP_{SN} and PTP_{SN} using them. When the *SN* decides to run the next session of the protocol, it is not authenticated by the *MN*. Therefore, it could not earn the necessary service from the *MN*.

5 ISAKA, Improved SEEMAKA

The main weaknesses of SEEMAKA, was related to the leakage of r_{SN} in the computations of the *MN*'s side and also not verification of the integrity of *Eta* and *Mu* in the *SN*'s side. In this section, an improvement of SEEMAKA is presented such that it resists against the attacks discussed in Section 4. The new protocol has been named ISAKA(Improved Secure Authentication and Key Agreement) protocol and similar to SEEMAKA comprises four main phases: setup phase, registration phase, mutual authentication and key agreement phase, and master key update phase. The setup, registration, and master key update phases in ISAKA are similar to these ones in SEEMAKA with some differences. In the registration phase of ISAKA there is no need to X_{SN} and also PTP_{SN} is not used in this phase. Also, in the master key update phase in ISAKA, we use r_{MN} instead of y. In ISAKA, the value of K_{MNSN} is a secret value associated with each sensor node *SN* that is stored only on the *MN* with its related TP_{SN}.

With this difference that PTP_{SN} is not used by ISAKA, for the sake of simplicity, we do not repeat their description in this section. In ISAKA, *MN* stores X_{MN}, PID_{FN} and (TP_{SN}, K_{MNSN}) for each sensor node. The mutual authentication and key agreement phase of ISAKA as represented in Figure 4 runs as follows:

1. *SN* chooses the random nonce r_{SN} and generates the current time as t_{SN}. Then it computes $B_{SN} = PID_{SN} \oplus TP_{SN}, C_{SN} = r_{SN} \oplus B_{SN}$ and $SP_{SN} = h(r_{SN}, t_{SN}, PID_{SN}, C_{SN})$ using PID_{SN} and TP_{SN}. Then it sends the message $M_1' =< SP_{SN}, t_{SN}, C_{SN}, TP_{SN} >$ to *FN*.
2. *FN* adds its identity i.e. PID_{FN}, to the message M_1' and sends the message $M_1 =< SP_{SN}, t_{SN}, C_{SN}, TP_{SN}, PID_{FN} >$ to *MN*.
3. After receiving M_1 by *MN*, it uses its local values of PID_{FN} and X_{MN} and firstly checks the validity of the received PID_{FN} and the time t_{SN}. Then it based on received TP_{SN} finds its related K_{MNSN} and calculates $PID^*_{SN} = TP_{SN} \oplus h(K_{MNSN}, X_{MN})$, $B^*_{SN} = PID^*_{SN} \oplus TP_{SN}$, $r^*_{SN} = B^*_{SN} \oplus C_{SN}$, and $SP^*_{SN} = h(r^*_{SN}, t_{SN}, PID^*_{SN}, C_{SN})$. Then it checks whether the received SP_{SN} of the message M_1 is equal to the calculated SP^*_{SN} or not. If so, it chooses the random nonce r_{MN} and a new X^+_{SN}. Afterwards, it calculates $C_{MN} = r^*_{SN} \oplus r_{MN}$, $TP^+_{SN} = h(K_{MNSN}, X^+_{SN}, X_{MN}, PID^*_{SN})$, $MTP^+_{SN} = TP^+_{SN} \oplus r_{MN}$ and $SP_{MN} = h(C_{MN}, r_{MN}, TP^+_{SN})$. Finally, it computes the session key i.e. *Skey* as $h(B^*_{SN}, r^*_{SN}, r_{MN}, PID^*_{SN})$ and stores it in its memory. Then it sends the message $M_2' =< C_{MN}, MTP^+_{SN}, SP_{MN}, PID_{FN} >$ to *FN*. It should be noted that in ISAKA, unlike SEEMAKA, the *MN* does not need to repeat the calculations to the number of stored

SN	FN	MN
$< PID_{SN}, TP_{SN} >$	$< PID_{FN} >$	$< PID_{FN}, X_{MN}, (K_{MNSN}, TP_{SN}) >$
Chooses a random r_{SN}, Considers the time t_{SN}, $B_{SN} = PID_{SN} \oplus TP_{SN}$, $C_{SN} = r_{SN} \oplus B_{SN}$, $SP_{SN} = h(r_{SN}, t_{SN}, PID_{SN}, C_{SN})$. $M_1' = \{SP_{SN}, t_{SN}, C_{SN}, TP_{SN}\} \longrightarrow$		
	$M_1 = \{SP_{SN}, t_{SN}, C_{SN}, TP_{SN},$ $PID_{FN}\} \longrightarrow$	
		Checks that PID_{FN} exists, Checks validity of t_{SN}, Based on TP_{SN}, receives K_{MNSN} from its memory $PID^*_{SN} = TP_{SN} \oplus h(K_{MNSN}, X_{MN})$, $B^*_{SN} = PID^*_{SN} \oplus TP_{SN}$, $r^*_{SN} = B^*_{SN} \oplus C_{SN}$, $SP^*_{SN} = h(r^*_{SN}, t_{SN}, PID^*_{SN}, C_{SN})$, Checks $SP^*_{SN} \stackrel{?}{==} SP_{SN}$, Chooses a random r_{MN} and the new amount X^+_{SN}, $C_{MN} = r^*_{SN} \oplus r_{MN}$, $TP^+_{SN} = h(K_{MNSN}, X^+_{SN}, X_{MN}, PID^*_{SN})$, $MTP^+_{SN} = TP^+_{SN} \oplus r_{MN}$ $SP_{MN} = h(C_{MN}, r_{MN}, TP^+_{SN})$, $Skey = h(B^*_{SN}, r^*_{SN}, r_{MN}, PID^*_{SN})$. $\longleftarrow M_2' = \{C_{MN}, MTP^+_{SN}, SP_{MN}, PID_{FN}\}$
	$\longleftarrow M_2 = \{C_{MN}, MTP^+_{SN}, SP_{MN}\}$	
$r^*_{MN} = C_{MN} \oplus r_{SN}$, $TP^{+*}_{SN} = MTP^+_{SN} \oplus r^*_{MN}$, $SP^*_{MN} = h(C_{MN}, r^*_{MN}, TP^{+*}_{SN})$, Checks $SP^*_{MN} \stackrel{?}{==} SP_{MN}$, Updates TP_{SN} with TP^{+*}_{SN} Updates PID_{SN} with $B_{SN} \oplus TP^{+*}_{SN}$.		

Fig. 4 The authentication and key agreement procedure of ISAKA

sensor node information. From TP_{SN}, MN realizes which K_{MNSN} should be used, which greatly reduces the complexity and time of the calculations.

4. FN deletes its identity, PID_{FN}, of the message M_2' and sends the message $M_2 =< C_{MN}, MTP^+_{SN}, SP_{MN} >$ to SN.
5. When SN received the message M_2, it firstly calculates $r^*_{MN} = C_{MN} \oplus r_{SN}$, $TP^{+*}_{SN} = MTP^+_{SN} \oplus r^*_{MN}$, and $SP^*_{MN} = h(C_{MN}, r^*_{MN}, TP^{+*}_{SN})$. Then it checks whether the received SP_{MN} of the message M_2 is equal to the calculated SP^*_{MN} or not. If so, it computes and stores the session key as $Skey = h(B_{SN}, r_{SN}, r^*_{MN}, PID_{SN})$. At the end, SN updates the amount of TP_{SN} and PID_{SN} in its memory with TP^{+*}_{SN} and $B_{SN} \oplus TP^{+*}_{SN}$ respectively.

6 Security Analysis of ISAKA

Here, we prove the security of the proposed scheme, ISAKA, using the informal method and also formal methods including BAN logic and ProVerif tool. The main goals of ISAKA is enhancing the security of SEEMAKA. Similar to SEEMAKA, ISAKA provides mutual authentication and key agreement. Moreover, the new scheme prevents the sensor node's traceability and reaches the anonymity property which cannot be satisfied in SEEMAKA. ISAKA is also safe against the attacks had influenced SEEMAKA.

In other words, ISAKA reaches higher security level compared to its predecessor, SEEMAKA.

6.1 Informal Security Proof

In this section, we use an informal method to demonstrate that ISAKA is resistant to the attacks that were successful on SEEMAKA.

6.1.1 Resistance to sensor node traceability

Unlike to SEEMAKA, ISAKA uses the dynamic identity of *SN* i.e. PID_{SN} which is updated after each successful session. Moreover, in SEEMAKA, we could use some exchanged messages to find PID_{SN}, however in ISAKA, the exchanged messages between *SN* and *MN* are produced in a way that an adversary could not use them to find PID_{SN} or some other fixed information related to the identity of *SN*. Therefore, the adversary could not trace an *SN* in ISAKA. In other words, ISAKA provides *SN* untraceability property.

6.1.2 Resistance to Disclosure of the Sensor Nodes and Master Nodes Secret Parameters

In SEEMAKA, there was the parameter PTP_{SN} which consists of the XOR of three parameters, TP_{SN}, X_{MN}, and X_{SN}. Since PTP_{SN} and TP_{SN} were the two common parameters between the *SN* and *MN*, then a malicious *SN* or *MN* could get the secret parameter of *MN* i.e. (X_{MN}) or *SN* i.e. (X_{SN}), respectively. In ISAKA, PID_{SN} and TP_{SN} are stored in *SN* which the XOR of them is equal to $PID_{SN} \oplus TP_{SN} = B_{SN} = h(K_{MNSN}, X_{MN})$ which is a constant value for each sensor node. Therefore, endangering one sensor node will not threaten the security of other sensor nodes or *MN*s. It is obvious that another adversary besides of the malicious *SN* or *MN*, also could not apply the attack on ISAKA.

6.1.3 Resistance to Sensor Node Impersonation

In order to impersonate the sensor node *SN* in ISAKA, the adversary must be calculating a valid C_{SN} such that when the *MN* controls its integrity, it accepts C_{SN}. Since C_{SN} contains XOR of r_{SN} and B_{SN} and B_{SN} is calculated using PID_{SN}, the identity of *SN* and TP_{SN}, a private common parameter between the *SN* and *MN*. In other words, the validity of C_{SN} generated with the adversary is related to the validity of PID_{SN} and also TP_{SN} chosen with him/her. In ISAKA, there is no approach to find a valid PID_{SN} or TP_{SN}. So, we conclude that ISAKA is secure against the sensor node impersonation attack.

6.1.4 Confidentiality of Session Key

In ISAKA, the session key is calculated as $Skey = h(B_{SN}, r_{SN}, r_{MN}, PID_{SN})$ in which B_{SN} is a secure common parameter between *SN* and *MN*, r_{SN} is a secret nonce generated with the *SN* and transmitted in a secure way to the *MN*, r_{MN} is a secret nonce generated with the *MN* and transmitted in a secure way to the *SN*, and PID_{SN} is the *SN*'s identity that only *MN* could calculate it. An adversary could find none of the mentioned parameters. Then she/he could not extract the session key between the *SN* and *MN*.

6.1.5 Resistance to Denial of Service Attack

In ISAKA, the integrity of all messages received by the *SN* or *MN* are controlled using a hash calculation. Then an adversary could not change, delete, or add some messages between the messages sent to the *MN* or *SN* to block the services of a side. On the other hand, there is no extra encryption or hash computation in ISAKA which can be used with the adversary to exploit a denial of service attack.

6.1.6 Providing Forward Secrecy

In ISAKA, the session key is computed as $Skey = h(B_{SN}, r_{SN}, r_{MN}, PID_{SN})$, which is a factor of ephemeral values r_{SN} and r_{MN}, respectively generated by the sensor node and the master node. Assuming that the master parameters for the sensor node or the master node is leaked at any later time, the session key of the previous sessions will not be compromised thanks to those ephemeral parameters, assuming that there is at least one session between the compromising time and the session key generation time for which the adversary has not eavesdropped the transferred messages between *SN* and *MN*. The reason comes from the fact the secret parameters of *SN*, i.e. PID_{SN} and TP_{SN}, are updated each session using no-invariable hash function. However, similar to any other symmetric cryptography based protocol forward secrecy is guaranteed if and only if the adversary can not monitor all sessions.

6.1.7 Misbehavior Traceability of the Medical Center

In ISAKA, we assume the patient, as a sensor node, is connected through a WBAN to the healthcare service provider, as a master node and the master node interacts as a bridge to connect the patient to the medical center where the doctor receives the data and provides appropriate action. To trace any misbehavior or dispute behaviour of the medical center/doctors, the master node keeps a log of all transferred information for the later usage.

6.2 Security Proof Based on BAN Logic Formal Method

In 1990, Burrows, Abadi, and Needham [7] presented a logic-based approach to verify the security of protocols in a formal method. Their approach was named BAN logic or BAN in short. In BAN logic, the protocol and its security goals were described based on BAN logic and it is deduced that the participants of the protocol believe the security goals or not. The main steps of a security proof using BAN logic are as follows [7]:

1. Consider the protocol and derive an idealized version of it. The idealization means considering only the necessary information to describe the protocol. In other words, in this manner some extra parameters of the protocol such as plaintext messages can be ignored.
2. Determine the assumptions which designers have considered to reach the security of their protocol.
3. Apply BAN logic rules to each step of the protocol and determine what is the beliefs of the participants in the step.
4. Conclude about the security of the protocol or some extra assumptions which are necessary for security of the protocol.

Here, we prove the security of ISAKA using BAN logic method and we show that the protocol's participants (the *SN* and *MN*) can achieve the mutuality belief in the shared key i.e. *Skey*. The notations which were used in the proof are represented in Table 2 where *A* and *B* represents the protocol participants and *X* and *Y* are some messages or concepts related to the protocol.

After that the protocol was idealized using Table 2's notations, the BAN logic rules must be applied to them. There are some rules introduced for BAN logic which some of them (used in this paper), are as follows where the numerator represents the assumption/s of rules and denominator represents the conclusion of it.

- P_1: $\dfrac{A \mid\equiv (A \leftrightarrow KB), A \lhd \{X\}_K}{A \mid\equiv B \mid\sim X}$
- P_2: $\dfrac{A \mid\equiv \#(X)}{A \mid\equiv \#(X, Y)}$
- P_3: $\dfrac{A \mid\equiv B \mid\sim X, A \mid\equiv \#(X)}{A \mid\equiv B \mid\equiv X}$
- P_4: $\dfrac{A \mid\equiv (X, Y)}{A \mid\equiv (X)}$
- P_5: $\dfrac{A \mid\equiv X, A \mid\equiv Y}{A \mid\equiv (X, Y)}$
- P_6: $\dfrac{A \mid\equiv (X)}{A \mid\equiv (X)_h}$
- P_7: $\dfrac{A \mid\equiv B \mid\equiv X, A \mid\equiv B \Rightarrow X}{A \mid\equiv X}$

ISAKA in BAN logic format

- $M_1 : MN \lhd SP_{SN} = (r_{SN}, t_{SN}, PID_{SN}, C_{SN})_h, t_{SN}, C_{SN}, TP_{SN}$
- $M_2 : SN \lhd C_{MN}, MTP^{+}_{SN}, SP_{MN} = (C_{MN}, r_{MN}, TP^{+}_{SN})_h$

Idealization of ISAKA

- $IM_1 : MN \lhd \{(MN \leftrightarrow B_{SN}SN), r_{SN}, t_{SN}, PID_{SN}\}_{B_{SN}}$
- $IM_2 : SN \lhd \{r_{MN}\}_{r_{SN}}, \{TP^{+}_{SN}\}_{r_{MN}}, \{(MN \leftrightarrow B_{SN}SN), (SN \leftrightarrow K_{MNSN}MN), X_{MN}, r_{MN}, r_{SN}, PID_{SN}, X^{+}_{SN}\}_{B_{SN}}$

Table 2 List of notations used in the security proof of ISAKA based on BAN logic

Notation	Description
$A \mid\equiv X$	*A* believes *X* (*A* believes *X* is true)
$A \lhd X$	*A* sees/receives *X*
$A \mid\sim X$	*A* once said *X* (or *A* has sent *X*)
$A \Rightarrow X$	*A* controls *X*
$\#(X)$	*X* is fresh
$\langle X \rangle_Y$	*X* is combined with *Y*
$\{X\}_Y$	*X* is encrypted with *Y*
$A \leftrightarrow KB$	*K* is a shared key between *A* and *B*
$Y = (X)_h$	*Y* is the hashing result of *X*

The assumptions of ISAKA which are used in its BAN logic security proof, are as follows:

- A_1: $MN \mid\equiv (MN \leftrightarrow TP_{SN}SN)$
- A_2: $SN \mid\equiv (MN \leftrightarrow TP_{SN}SN$
- A_3: $SN \mid\equiv (SN \leftrightarrow B_{SN} = h(K_{MNSN}, X_{MN})MN)$
- A_4: $MN \mid\equiv (MN \leftrightarrow B_{SN} = h(K_{MNSN}, X_{MN})SN)$
- A_5: $MN \mid\equiv \#(r_{MN})$
- A_6: $MN \mid\equiv SN \Rightarrow (MN \leftrightarrow B_{SN}SN)$
- A_7: $SN \mid\equiv MN \Rightarrow (MN \leftrightarrow B_{SN}SN)$
- A_8: $SN \mid\equiv \#(r_{SN})$
- A_9: $SN \mid\equiv MN \Rightarrow (MN \leftrightarrow SkeySN)$
- A_{10}: $MN \mid\equiv SN \Rightarrow (MN \leftrightarrow SkeySN)$

In order to decide about the security of ISAKA, the following security goals must be satisfied in this protocol.

- SG_1: $MN \mid\equiv SN \mid\equiv (MN \leftrightarrow SkeySN)$
- SG_2: $SN \mid\equiv MN \mid\equiv (MN \leftrightarrow SkeySN)$
- SG_3: $SN \mid\equiv (MN \leftrightarrow SkeySN)$
- SG_4: $MN \mid\equiv (MN \leftrightarrow SkeySN)$

Now we can consider the idealization form and assumptions of ISAKA and use BAN logic rules to conclude its security goals.

6.2.1 Security Goal SG_1

In order to prove SG_1, given IM_1 which is $MN \lhd \{(MN \leftrightarrow B_{SN}SN), r_{SN}, t_{SN}, PID_{SN}\}_{B_{SN}}$ and $A4$ and based on rule P_1 we get:

D_1: $MN \mid\equiv SN \mid\sim ((MN \leftrightarrow B_{SN}SN), r_{SN}, t_{SN}, PID_{SN})$.

From A_5 and given $r_{SN} = r_{MN} \oplus C_{MN}$, D_2 is concluded as follows:

D_2: $MN \mid\equiv \#(r_{SN})$.

Given D_2 and based on P_2 we get:

D_3: $MN \mid\equiv \#((MN \leftrightarrow B_{SN}SN), r_{SN}, t_{SN}, PID_{SN})$.

From D_1 and D_3 and by applying the rule P_3 we get:

D_4: $MN \mid\equiv SN \mid\equiv ((MN \leftrightarrow B_{SN}SN), r_{SN}, t_{SN}, PID_{SN})$.

By considering D_4 based on P_4, D_5 is concluded as below:

D_5: $MN \mid\equiv SN \mid\equiv (MN \leftrightarrow B_{SN}SN)$.

Given that $r_{SN} = B_{SN} \oplus C_{SN}$, from D_5 and based on P_4 we get:

D_6: $MN \mid\equiv SN \mid\equiv (MN \leftrightarrow r_{SN}SN)$.

Similarly, given $PID_{SN} = B_{SN} \oplus TP_{SN}$, D_5 and based on P_4, we can conclude D_7 as follows:

D_7: $MN \mid\equiv SN \mid\equiv (MN \leftrightarrow PID_{SN}SN)$.

Given $r_{MN} = C_{MN} \oplus r_{SN}$, therefore using D_6 and based on P_4, we can conclude D_8 as follows:

D_8: $MN \mid\equiv SN \mid\equiv (MN \leftrightarrow r_{MN}SN)$.

Now, we use D_5, D_6, D_7, and D_8 and apply P_5 to get:

D_9: $MN \mid\equiv SN \mid\equiv (MN \leftrightarrow B_{SN}, r_{SN}, r_{MN}, PID_{SN}SN)$.

Finally from D_9 and by applying P_6 we conclude D_{10} as follows: D_{10} : $MN \mid\equiv SN \mid\equiv (MN \leftrightarrow (B_{SN}, r_{SN}, r_{MN}, PID_{SN})_h SN)$.

Since $Skey = (B_{SN}, r_{SN}, r_{MN}, PID_{SN})_h$, so we can conclude $\Rightarrow MN \mid\equiv SN \mid\equiv (MN \leftrightarrow SkeySN)$ which is same SG_1.

6.2.2 Security Goal SG_2

By using IM_2 and A_3 based on the rule P_1 we find that D_{11} as:

$D_{11} : SN \mid\equiv MN \mid\sim ((MN \leftrightarrow B_{SN}SN), (SN \leftrightarrow K_{MNSN}MN), X_{MN}, r_{MN}, r_{sN}, PID_{sN}, X^+_{sN})$.

From A_8 and the rule P_2, D_{12} is concluded as follows:

D_{12}: $SN \mid\equiv \#(()(MN \leftrightarrow B_{SN}SN), (SN \leftrightarrow K_{MNSN}MN), X_{MN}, r_{MN}, r_{SN}, PID_{SN}, X^+_{sN})$.

From D_{11} and D_{12} and after applying the rule P_3, we get:

D_{13}: $SN \mid\equiv MN \mid\equiv ((MN \leftrightarrow B_{SN}SN), (SN \leftrightarrow K_{MNSN}MN), X_{MN}, r_{MN}, r_{SN}, PID_{SN}, X^+_{sN})$.

By considering D_{13} with the rule P_4, D_{14} is concluded as below:

$D_{14} : SN \mid\equiv MN \mid\equiv (MN \leftrightarrow B_{SN}SN)$.

Given $r_{SN} = B_{SN} \oplus C_{SN}$, then from D_{14} and based on P_4, we get:

D_{15}: $SN \mid\equiv MN \mid\equiv (MN \leftrightarrow r_{SN}SN)$.

Similarly, given $PID_{SN} = B_{SN} \oplus TP_{SN}$ and D_{14} and based on P_4, D_{16} is concluded as follows:

D_{16}: $SN \mid\equiv MN \mid\equiv (MN \leftrightarrow PID_{SN}SN)$.

Given that $r_{MN} = C_{MN} \oplus r_{SN}$, therefore using D_{15} and based on P_4, D_{17} is concluded as follows:

D_{17}: $MN \mid\equiv SN \mid\equiv (MN \leftrightarrow r_{MN}SN)$.

By using D_{14}, D_{15}, D_{16}, and D_{17} and applying P_5, we get:

D_{18}: $SN \mid\equiv MN \mid\equiv (MN \leftrightarrow B_{SN}, r_{SN}, r_{MN}, PID_{SN}SN)$.

Finally from D_{18} and by applying P_6 we conclude D_{19} as follows:

$D_{19} : SN \mid\equiv MN \mid\equiv (MN \leftrightarrow (B_{SN}, r_{SN}, r_{MN}, PID_{SN})_h SN)$.

Since $Skey = (B_{SN}, r_{SN}, r_{MN}, PID_{SN})_h$, so we can conclude $\Rightarrow SN \mid\equiv MN \mid\equiv (MN \leftrightarrow SkeySN)$ which is same SG_2.

6.2.3 Security Goal SG_3

To prove the objective SG_3, it is sufficient to consider the assumption A_9 and security goal SG_2 based on the rule P_7 which results $SN \mid\equiv (MN \leftrightarrow SkeySN)$ which is same SG_3.

6.2.4 Security Goal SG_4

From the assumption A_6 and D_5 and after applying the rule P_7, D_{20} is obtained as:

D_{20}: $MN \mid\equiv (MN \leftrightarrow B_{SN}SN)$.

As we explained about the proving of object SG_1, we recall that $r_{SN} = B_{SN} \oplus C_{SN}$ and $PID_{SN} = B_{SN} \oplus TP_{SN}$. Then from D_{20} and based on P_4, we get

D_{21}: $MN \mid\equiv (MN \leftrightarrow PID_{SN}SN)$ and D_{22}: $MN \mid\equiv (MN \leftrightarrow r_{SN}SN)$ respectively.

Similarly, we recall that $r_{MN} = C_{MN} \oplus r_{SN}$. So, given D_{22} and based on P_4, we obtain

D_{23}: $MN \mid\equiv (MN \leftrightarrow r_{MN}SN)$.

Now by using D_{20}, D_{21}, D_{22}, D_{23} and applying P_5, D_{24} is resulted as:

D_{24}: $MN \mid\equiv (MN \leftrightarrow B_{SN}, r_{SN}, r_{MN}, PID_{SN}SN)$.

Finally by applying P_6 on D_{24}, D_{25} is deduced as follows:

$D_{25} : MN \mid\equiv (MN \leftrightarrow (B_{SN}, r_{SN}, r_{MN}, PID_{SN})_h SN)$.

Since $Skey = (B_{SN}, r_{SN}, r_{MN}, PID_{SN})_h$, so we can conclude $\Rightarrow MN \mid\equiv (MN \leftrightarrow SkeySN)$ which is same SG_4.

6.3 Formal Verification of ISAKA Using Proverif Tool

In this section, we verify the security of the proposed protocol, ISAKA using the ProVerif tool [6]. ProVerif can verify the security goals of a protocol such as secrecy and authentication in an automatically manner. It supports modelling of some operations, functions, and primitives used in the protocol such as XOR operation and hash function. In order to verify the security of a protocol with ProVerif the following steps should be done.

- Modelling the protocol using one of its supported languages. The pi calculus is currently considered to be state-of-the-art and files of this sort are denoted by the file extension. pv.
- Modelling the protocol's security objects that we want to verify them.
- Analysis the output of ProVerif and decide about its security.

In order to model ISAKA and its security goals with ProVerif, we use the declarations shown in Table 3.

After that we declare the necessary types, names, constructors, destructors, events, and queries, we must model the ISAKA participants as sub-processes (macros), as shown in Table 4.

Table 3 The declarations used to model ISAKA and its security goals in ProVerif

```
(* Types and Free Names *)
free C:channel.

type Nonce.
type TimeStamp.

free PIDsn: bitstring [private].
free Tsn: TimeStamp [private].
free S: bitstring [private].

(* XOR function and its equation *)
fun xor(bitstring,bitstring):bitstring.
equation forall m:bitstring,n:bitstring;xor(xor(m,n),n)=m.

(* Hash Function *)
fun h(bitstring):bitstring.

(* Type Converter *)
fun nonce2bitstring(Nonce): bitstring [data,typeConverter].

(* Events *)
event beginSensorNode(bitstring).
event endSensorNode(bitstring).
event beginMasterNode(bitstring, Nonce).
event endMasterNode(bitstring, bitstring).
event Timestamp(TimeStamp).

(********** Query **********)
query attacker(PIDsn).
query attacker(new Xmn).
query attacker(new Kmnsn).
query attacker(new rsn).
query attacker(new rmn).
query attacker (S).
query pid1:bitstring, pid2:Nonce; inj-event(endMasterNode(pid1, pid1))==>inj-event(beginMasterNode(pid1, pid2)).
query pid:bitstring; inj-event(endSensorNode(pid))==> inj-event(beginSensorNode(pid)).
query Ts:TimeStamp, pid:bitstring; inj-event(beginSensorNode(pid))==> inj-event(Timestamp(Ts)).
```

Table 4 The sub-processes (macros) of ISAKA

```
(********** SensorNode Role Descryption **********)
let SensorNode(TPsn:bitstring, PIDsn:bitstring, Tsn:TimeStamp)=
new rsn:Nonce;
let Bsn = xor (PIDsn,TPsn) in
let rsnT = nonce2bitstring(rsn) in
let Csn = xor(rsnT,Bsn)in
let SPsn = h((rsn,Tsn,Csn,PIDsn))in

event beginMasterNode(PIDsn, rsn);

(* First Flow*)
out(C,(SPsn,Tsn,Csn,TPsn));
in(C,(Cmn:bitstring,nMTPsn:bitstring,SPmn:bitstring));

let Trsn = nonce2bitstring (rsn) in
let xrmn=xor(Cmn,Trsn) in
let xTPsn=xor(nMTPsn,xrmn) in
let xSPmn=h((Cmn,xrmn,xTPsn)) in
if xSPmn = SPmn then
let skey = h((Bsn,rsn,xrmn,PIDsn)) in
out (C, sencrypt(S, skey));

event endSensorNode(xSPmn).

(********** MasterNode Role Descryption **********)
let MasterNode(Xmn:bitstring, Kmnsn:bitstring, Tsn:TimeStamp)=
in(C,(SPsn:bitstring, xTsn:TimeStamp, Csn:bitstring, TPsn:bitstring));
if xTsn = Tsn then
event Timestamp(Tsn);
let xPIDsn = xor (TPsn, h((Kmnsn,Xmn))) in
let xBsn=xor(xPIDsn,TPsn)in
let xrsn=xor(Csn,xBsn)in
let xSPsn=h((xrsn,xTsn,xPIDsn,Csn))in
if xSPsn=SPsn then
new rmn:Nonce;
let Trmn = nonce2bitstring (rmn) in
new nXsn:bitstring;
let Cmn=xor(xrsn,Trmn)in
let nXmn=xor(Xmn,xor(Trmn,nXsn))in
let nTPsn=h((Kmnsn,nXsn,Xmn,xPIDsn))in
let nMTPsn=xor(nTPsn,Trmn)in
let SPmn=h((Cmn,rmn,nTPsn))in
let skey = h((xBsn,xrsn,rmn,xPIDsn))in

event beginSensorNode(SPmn);

(*Second Flow*)
out(C,(Cmn,nMTPsn,SPmn));
out(C, sencrypt(S, skey));

event endMasterNode(xPIDsn, xrsn).
```

Table 5 The process of ISAKA in ProVerif tool

```
process
(

new TPsn: bitstring;
new Xmn: bitstring;
new Kmnsn: bitstring;

(!SensorNode(TPsn, PIDsn, Tsn)) — (!MasterNode(Xmn, Kmnsn, Tsn))

)
```

Table 6 The results of cryptanalysis of ISAKA using ProVerif tool

```
Verification summary:

Query not attacker(PIDsn[]) is true.
Query not attacker(Xmn[]) is true.
Query not attacker(Kmnsn[]) is true.
Query not attacker(rsn[!1 = v]) is true.
Query not attacker(rmn[TPsn-1 = v,Csn-1 = v-1,xTsn = v-2,SPsn-1 = v-3,!1 = v-4]) is true.
Query not attacker(S[]) is true.

Query inj-event(endMasterNode(pid1,pid1)) ==> inj-event(beginMasterNode(pid1,pid2)) is true.
Query inj-event(endSensorNode(pid)) ==> inj-event(beginSensorNode(pid)) is true.
Query inj-event(beginSensorNode(pid)) ==> inj-event(Timestamp(Ts)) is true.
```

Finally, we use the declarations and sub-process to model ISAKA as a process. This process is shown in Table 5 and can be executed by ProVerif.

We could use the query, *query attacker (M)*, to investigate the secrecy of the term *M*. Since in ISAKA the secrecy of PID_{SN}, X_{MN}, K_{MNSN}, r_{SN}, r_{MN}, and *Skey* are necessary, then we use the mentioned query to verify their secrecy. We also used the events *beginSensorNode*, *endSensorNode*, *beginMasterNode*, and *endMasterNode* and the feature of injective correspondence of ProVerif to verify the mutual authentication of ISAKA.

After running the codes of ISAKA in ProVerif, the obtained results show that the security goals of this protocol are satisfied. Summary of these results are shown in Table 6.

7 Comparison in Terms of Security and Efficiency

In this section, we compare the new proposed protocol, ISAKA, with some other similar protocols (especially SEEMAKA) which proposed for use in WBAN in terms of security and efficiency. Precisely, the authors of [15] compared the security and efficiency of their protocol with some other similar protocols such as [11, 13, 14, 19, 22] and claimed high level security and efficiency for their protocol i.e. SEEMAKA. In the terms of security, we considered the weaknesses of SEEMAKA and showed that our improved protocol, ISAKA, resists against the vulnerabilities of SEEMAKA i.e. sensor node traceability, sensor node impersonation, disclosure of the secret parameters of the sensor nodes and master nodes, and extracting the session key attacks. We also proved the claimed security of our proposed protocol using informal and formal methods. We summarize the comparison of security of

Table 7 Security comparison of ISAKA with SEEMAKA

Vulnerability	SEEMAKA	ISAKA
Sensor node traceability	×	✓
Disclosure of the secret parameters	×	✓
Sensor node impersonation	×	✓
Session key confidentiality	×	✓
Denial of Service attack	×	✓
Forward/backward secrecy	×	✓

Table 8 ROM storage requirements of ISAKA (in bit) compared to SEEMAKA and other similar protocols

	[11]	[19]	[14]	SEEMAKA	ISAKA
SN	3(160) = 480	5(160) = 800	4(160) = 640	5(160) = 800	3(160) = 480
MN	$160 + 320w$	$320 + 320w + 160p$	$320 + 160p$	$320 + 160w + 160p$	$320 + 320w + 160p$

p : number of fixed nodes , w=number of sensor nodes

ISAKA and SEEMAKA in Table 7. To compare the security of ISAKA with some other WBAN protocols, one can consider Table 7 along with the security comparison in [15].

Similar to [15], we discuss efficiency of ISAKA in the terms of ROM and RAM storage requirements, communication cost, processing cost, processing time, and energy dissipation.

Storage requirements. With respect to the parameters that must be saved in the sensor and master node's memory when we use the protocol ISAKA, this protocol is similar to the protocol SEEMAKA with the difference where we don't need X_{SN} and also PTP_{SN}. In other words, the parameters stored in the sensor node's storage are PID_{SN}, TP_{SN}, and finally the session key, $Skey$. Also, the parameters stored in the master node's storage are X_{MN}, PID_{FN} for each FN, and K_{MNSN} and TP_{SN} for each sensor and and finally the session key, $Skey$. In these comparisons, we assume that we use the hash function SHA-1 (with the 160-bit output) in ISAKA and all variables and also the session key have the 160-bit length. Based on these assumptions, we compute the amount of ROM memory and RAM memory that we need in the protocol ISAKA and compare it with the ROM and RAM memory requirements of SEEMAKA and the other similar protocols. The results are summarized in Tables 8 and 9 respectively which show that ISAKA can improve the SEEMAKA, [19] and [14] in the terms of ROM and RAM storage requirements.

Communication Cost. In order to compute the communication cost of ISAKA, we consider the messages that are transmitted from SN to FN and FN to MN and vice versa. With respect to Figure 3 and assumptions about the length of variable and hash output in ISAKA, we see that three messages of length 160-bit and one message of length 32-bit (we assume that the time stamp size of ISAKA is similar to SEEMAKA equals to 32-bit) are transmitted from SN to FN. Similarly, we see that four messages of length 160-bit and one message of length 32-bit, three messages of length 160-bit and one message of length 32-bit, and finally two messages of length 160-bit and one message of length 32-bit are transmitted from FN to MN, MN to FN, and FN to SN, respectively. We summarize the communication cost of ISAKA in Table 10 and compare with the communication cost of SEEMAKA and the other similar protocols. It can be seen in

Table 9 RAM storage requirements of ISAKA (in bit) compared to SEEMAKA and other similar protocols

	[11]	[19]	[14]	SEEMAKA	ISAKA
SN	5(160) + 32 = 832	4(160) + 192 + 4(32) = 960	9(160) + 2(32) = 1504	7(160) + 4(32) = 1248	5(160) + 3(32) = 896
MN	8(160) + 32 = 1312	8(160) + 192 + 3(32) = 1568	14(160) + 32 = 2272	12(160) + 2(32) = 1984	8(160) + 3(32) = 1376

Table 10 Communication cost comparison of ISAKA with SEEMAKA and other similar protocols(in bits)

	[11]	[19]	[14]	SEEMAKA	ISAKA
$SN \rightarrow FN$	$3(160) = 480$	$3(160) + 32 = 512$	$4(160) + 32 = 672$	$4(160) + 32 = 672$	$3(160) + 32 = 512$
$FN \rightarrow MN$	$4(160) = 640$	$4(160) + 32 = 672$	$5(160) + 32 = 832$	$5(160) + 32 = 832$	$4(160) + 32 = 672$
$MN \rightarrow FN$	$4(160) = 640$	$2(160) + 192 + 32 = 544$	$5(160) = 800$	$6(160) = 960$	$3(160) + 32 = 512$
$FN \rightarrow SN$	$3(160) = 480$	$160 + 32 + 192 = 384$	$4(160) = 640$	$5(160) = 800$	$2(160) + 32 = 352$
Total	2240	2112	2944	3264	2048

Table 10, the proposed protocol i.e. ISAKA has been able to reduce the total amount of communication costs compared to its predecessor, [11, 19] and [14].

Processing cost, processing time, and energy dissipation In the terms of processing cost, processing time, and energy dissipation, we showed that the new proposed protocol needs less energy dissipation compared to its predecessor i.e. SEEMAKA. In order to compare the processing time and also energy dissipation of ISAKA with SEEMAKA and also some other similar protocols, we used the setup introduced in [15]. Given that the simulation setup for the hardware environment is a 32-bit Cortex-M3 microcontroller operating at 72 MHz, the hash function SHA-1 needs 0.06 ms while XOR operation takes nothing for processing. If we assumed that in ISAKA we use SHA-1 as the hash function, then we calculate the processing time for this protocol as $0.06 \times 8 = 0.48$ ms. Compared to SEEMAKA, the new proposed protocol needs the much less processing time than SEEMAKA.

Given [15], we know that the same microcontroller consumes 36 mA under 3.3 V and consequently approximately dissipates 118.8 mW energy in active mode in the temperature $27°C$. Therefore, ISAKA dissipates the energy $(0.48 \times 118.8) \div 1000 = 0.057024$ mJ. Compared to SEEMAKA, the new proposed protocol needs less energy than SEEMAKA. Note that if we do not ignore the energy dissipation of XOR operations, we can say that the energy dissipation of ISAKA is much lower than SEEMAKA's one. The comparison of ISAKA with SEEMAKA and also some other similar protocols in terms of processing cost, processing time, and energy dissipation is summarized in Table 11 .

Table 11 Comparison of ISAKA with SEEMAKA and some other similar protocols in terms of processing cost, processing time, and energy dissipation

Protocol	PC* of SN	PC of MN	PT** of SN	PT of MN	Total ED***
[11]	$5t_{hash} + 2\ t_{XOR}$	$8t_{hash} + 4\ t_{XOR}$	0.3	0.48	0.09266
[19]	$4t_{hash} + 2\ t_{XOR}$	$7\ t_{hash} + 4\ t_{XOR}$	0.24	0.42	0.0784
[14]	$3\ t_{hash}$+$7\ t_{XOR}$	$5\ t_{hash}$+$12\ t_{XOR}$	0.18	0.3	0.057024
SEEMAKA	$3\ t_{hash}$+$7\ t_{XOR}$	$(2t_{hash} + 4t_{XOR})w + 3\ t_{hash} + 6\ t_{XOR}$	0.18	1.38	0.185328
ISAKA	$3\ t_{hash}$+$4\ t_{XOR}$	$5\ t_{hash}$+$5\ t_{XOR}$	0.18	0.3	0.057024

*PC: Processing Cost, **PT: Processing Time (ms), ***ED: Energy Dissipation (mJ)

t_{hash}: Processing time of hash function, t_{XOR}: Processing time of XOR operation

w: number of sensor nodes which in these calculations is considered equal to 10

8 Conclusion

In this paper, we examined the security of SEEMAKA, a lightweight authentication and key agreement protocol for wireless body area networks, and showed its security pitfalls. We showed that this protocol suffers from the sensor node traceability, disclosure of the sensor nodes and master nodes secret parameters, sensor node impersonation, extracting the session key, and denial of service attacks. We also improved the efficiency and security of SEEMAKA and presented a new lightweight mutual authentication and key agreement scheme for wireless body area networks, named ISAKA (Improved Secure Authentication and Key Agreement) protocol. Compared to SEEMAKA, ISAKA improves the ROM and RAM storage requirements and communication costs and needs less energy dissipation. It is secure against the known attacks, especially the attacks discussed in this paper. In order to prove the security of ISAKA, we used BAN logic method and also verify its security using ProVerif tool.

Funding This work was supported by Shahid Rajaee Teacher Training University.

Data Availibility Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

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

Ethical Approval This manuscript does not contain any studies with human participants or animals performed by any of the authors.

References

1. Agha, D.-e.-S., Khan, F. H., Shams, R., Rizvi, H. H., & Qazi, F. (2018). A secure crypto base authentication and communication suite in wireless body area network (WBAN) for IoT applications. *Wireless Personal Communications*, 103(4):2877–2890, 2018
2. Alzahrani, B. A. (2021). Secure and efficient cloud-based IoT authenticated key agreement scheme for e-health wireless sensor networks. *Arabian Journal for Science and Engineering, 46*(4), 3017–3032.
3. Alzahrani, B.A., Irshad, A., Albeshri, A., & Alsubhi, K. (2020). A provably secure and lightweight patient-healthcare authentication protocol in wireless body area networks. *Wireless Personal Communications*, pp. 1–23
4. Amin, R., & Biswas, G. (2015). An improved RSA based user authentication and session key agreement protocol usable in TMIS. *Journal of Medical Systems, 39*(8), 79.
5. Arshad, H., & Rasoolzadegan, A. (2016). Design of a secure authentication and key agreement scheme preserving user privacy usable in telecare medicine information systems. *Journal of Medical Systems, 40*(11), 237.
6. Blanchet B. (2012). Proverif: automatic cryptographic protocol verifier user manual for untyped inputs
7. Burrows, M., Abadi, M., & Needham, R.M. (1871). A logic of authentication. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 426(1871):233–271
8. Fotouhi, M., Bayat, M., Das, A.K., Far, H.A.N., Pournaghi, S.M., & Doostari, M. (2020). A lightweight and secure two-factor authentication scheme for wireless body area networks in health-care IoT. *Computer Networks*, pp. 107333

9. Giri, D., Maitra, T., Amin, R., & Srivastava, P. (2015). An efficient and robust RSA-based remote user authentication for telecare medical information systems. *Journal of Medical Systems, 39*(1), 145.
10. Hussain, S. J., Irfan, M., Jhanjhi, N., Hussain, K., & Humayun, M. (2021). Performance enhancement in wireless body area networks with secure communication. *Wireless Personal Communications, 116*(1), 1–22.
11. Ibrahim, M. H., Kumari, S., Das, A. K., Wazid, M., & Odelu, V. (2016). Secure anonymous mutual authentication for star two-tier wireless body area networks. *Computer Methods and Programs in Biomedicine, 135,* 37–50.
12. I. B. Karthigaiveni M. An efficient two-factor authentication scheme with key agreement for iot based e-health care application using smart card. *Journal of Ambient Intelligence and Humanized Computing*, 2019
13. Li, X., Ibrahim, M. H., Kumari, S., & Kumar, R. (2018). Secure and efficient anonymous authentication scheme for three-tier mobile healthcare systems with wearable sensors. *Telecommunication Systems, 67*(2), 323–348.
14. Li, X., Ibrahim, M. H., Kumari, S., Sangaiah, A. K., Gupta, V., & Choo, K.-K.R. (2017). Anonymous mutual authentication and key agreement scheme for wearable sensors in wireless body area networks. *Computer Networks, 129*, 429–443.
15. Narwal, B. & Mohapatra, A.K. (2020) SEEMAKA: secured energy-efficient mutual authentication and key agreement scheme for wireless body area networks. *Wireless Personal Communications*, pp. 1–24, 2020
16. Narwal, B., & Mohapatra, A. K. (2021). Samaka: Secure and anonymous mutual authentication and key agreement scheme for wireless body area networks. *Arabian Journal for Science and Engineering, 46*(9), 9197–9219.
17. Nikooghadam, M., & Amintoosi, H. (2020). An improved secure authentication and key agreement scheme for healthcare applications. In *2020 25th International Computer Conference, Computer Society of Iran (CSICC)*, pp. 1–7. IEEE
18. Ostad-Sharif, A., Abbasinezhad-Mood, D., & Nikooghadam, M. (2019). An enhanced anonymous and unlinkable user authentication and key agreement protocol for TMIS by utilization of ECC. *International Journal of Communication Systems, 32*(5), e3913.
19. Ostad-Sharif, A., Nikooghadam, M., & Abbasinezhad-Mood, D. (2019). Design of a lightweight and anonymous authenticated key agreement protocol for wireless body area networks. *International Journal of Communication Systems, 32*(12), e3974.
20. Shaik, M. F., Komanapalli, V. L. N., & Subashini, M. M. (2018). A comparative study of interference and mitigation techniques in wireless body area networks. *Wireless Personal Communications, 98*(2), 2333–2365.
21. Soni, M., & Singh, D. K. (2021) Laka: lightweight authentication and key agreement protocol for internet of things based wireless body area network. *Wireless Personal Communications*, pp. 1–18
22. Xu, Z., Xu, C., Chen, H., & Yang, F. (2019). A lightweight anonymous mutual authentication and key agreement scheme for WBAN. *Concurrency and Computation: Practice and Experience, 31*(14), e5295.
23. Zimmerman, T. G. (1996). Personal area networks: Near-field intrabody communication. *IBM systems Journal*, 35(3.4):609–617

Javad Alizadeh received the Ph.D. degree in Cryptography from Imam Hossein Comprehensive University, Tehran, Iran, in 2016. He is now an Assistant Professor with the Faculty and Research Center of Computer, Imam Hossein Comprehensive University, Tehran, Iran. His research interests include cryptanalysis of authentication protocols and lightweight block ciphers in IoT. He is the author of over 10 articles in information security and cryptology.

Masoumeh Safkhani received the Ph.D. degree in electrical engineering from the Iran University of Science and Technology, in 2012, with the security analysis of RFID protocols as her major field. She is currently an Associate Professor with the Computer Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran. Her current research interests include the security analysis of lightweight and ultra-lightweight protocols, targeting constrained environments, such as RFID, the IoT, VANET, and WSN. She is the author/coauthor of over 50 technical articles in information security and cryptology in major international journals and conferences.

Amir Allahdadi received his B.Sc. in in Telecommunication, from Shahrood University of Technology, Shahrood, Iran, in 2017. He is currently working toward the M.Sc. degree in Cryptography at Imam Hossein Comprehensive University, Tehran, Iran. His research interests include authentication protocols in Wireless Sensor Networks.