Arabian Journal for Science and Engineering (2023) 48:1947–1971
https://doi.org/10.1007/s13369-022-07055-2

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

A Robust Four-Factor Authentication Protocol for Resource Mining

Diksha Rangwani[1] · Hari Om[1]

Received: 12 December 2021 / Accepted: 9 June 2022 / Published online: 23 July 2022

Abstract
Resource mining is the basic necessity for a country's technological and economic development. Rare-earth elements reshape the economy of a country. Resources like silicon, aluminum, gold, quartz, potassium, iron, chromium, etc., are being used to develop the smart devices. With the current technological advancements, wireless sensor networks (WSNs) have become the backbone for resource mining. Using WSNs for resource mining gives ease of remote usage and decision-making with just a few clicks. However, the wireless channels are prone to various threats due to their openness, and safeguarding such a hazardous and valuable environment is essential. In this paper, we analyze the Wu et al.'s scheme and find that it suffers from various attacks like gateway impersonation, man-in-the-middle, privileged insider, sensor node impersonation, sensor node capture, session-specific transient information, and denial-of-service. We design a four-factor authentication protocol for resource mining. The formal verification of the proposed protocol is carried out using the Burrows–Abadi–Needham (BAN) logic and real-or-random (ROR) model. Its robustness is ensured by using the informal security analysis. The protocol is simulated using ProVerif to ensure the secrecy of variables and security from various attacks. Finally, a comparative analysis with the existing protocols is carried out to show its efficacy in a practical environment.

Keywords Multi-factor security · ROR model · Mining · Authentication

1 Introduction

Natural resources like fossil fuels, nuclear material, rare-earth elements, heavy metals, etc., play a vital role in a nation's technological and economic development. Mining these resources is very costly and time-consuming as the human power involved in underground mining is always at the risk of their lives. Due to the advancements in technologies, the use of WSNs in mining resources has increased remarkably. A WSN generally consists of the sensors and gateways. A sensor node gathers the data like the concentration of explosive gases, smoke, air velocity, pressure, temperature, etc. The gateway does data aggregation and data forwarding and works as a mediator for communicating between the sensors and users [1]. The sensor nodes are spatially deployed in an unreachable area (hazardous for humans) to form a network. Since the sensors are low-cost and low-power devices, they can function efficiently only with low overheads.

The basic architecture for resource mining with WSN as its backbone is depicted in Fig. 1. A remote user can access the data collected by the sensor nodes for various purposes like studying the environment, designing the safety rules as per the underground conditions in a mine, checking the availability of the resources, suitability of conditions to deploy the human resources, etc. The person monitoring the underground conditions on the ground is just another user of the system present in the vicinity of the mines; he can make emergency decisions regarding safety, evacuation, etc. The server primarily provides the distributed and shared working environment in terms of the data stored in a database that could be related to the previous conditions of casualties, etc. The data can be shared among the system users to study the conditions of safety monitoring and verify the availability of specific resources under certain conditions or which conditions of the underground environment could be safe to deploy the human resources, etc.

The WSNs use the public channels, which are vulnerable to various security threats like eavesdropping, forgery,

✉ Diksha Rangwani
diksharangwani@gmail.com

Hari Om
hariom4india@gmail.com

1 Department of Computer Science and Engineering, Indian Institute of Technology(ISM) Dhanbad, Police Line Road, Dhanbad, Jharkhand 826004, India

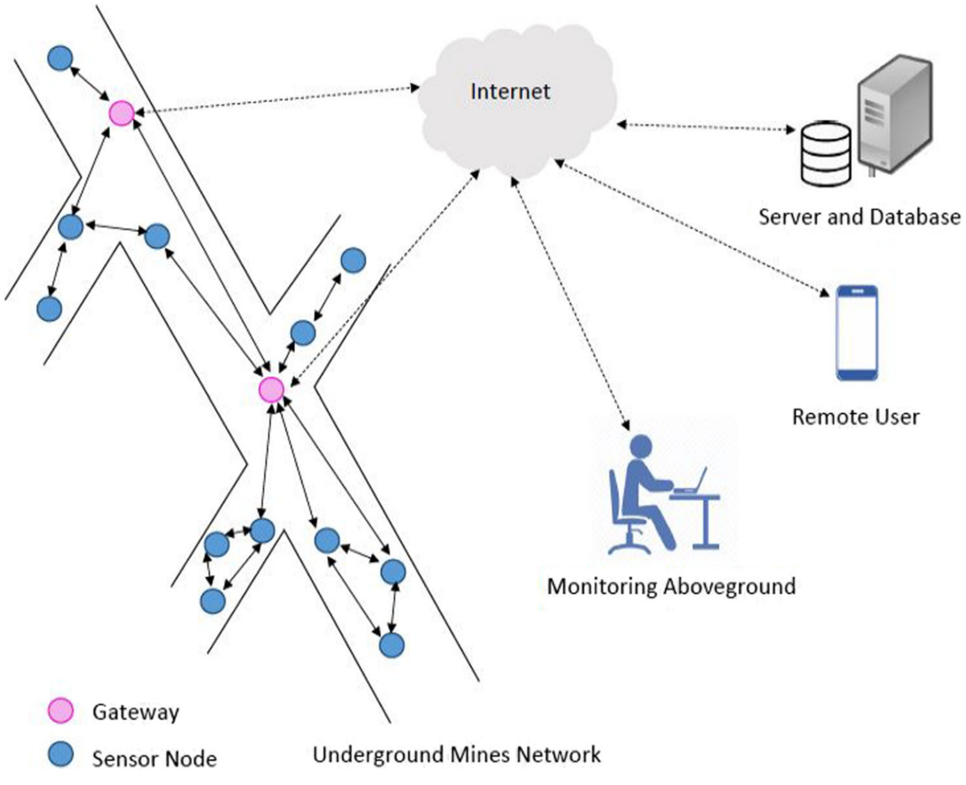


Fig. 1 Architecture of resource mining

impersonation, denial-of-service, man-in-the-middle, replaying, etc. [2,3]. Hence, the security becomes a vital factor for designing an authentication protocol for underground mining. Recently, Wu et al. [4] have discussed an authentication protocol for the Internet of Things (IoT) using WSNs as an underlying network. In this paper, we cryptanalyze the Wu et al.'s protocol and show that it suffers from several attacks: gateway impersonation, man-in-the-middle, privileged insider, sensor node impersonation, sensor node capture, session-specific transient information, and denial-of-service. Further, we propose a novel protocol for overcoming these drawbacks.

1.1 Related Work

Althobaiti et al. [5] discuss a lightweight mutual authentication protocol using the user's iris scan and biometric encryption to generate a robust key to authenticate a user for WSNs. Das et al. [6] report that the protocol [5] cannot resist the sensor node capture attack, impersonation attack, and man-in-the-middle attack. They discuss a mutual authentication protocol for WSNs by overcoming these vulnerabilities. Turkanovic et al. [7] discuss a lightweight key negotiation and user authentication protocol for the ad hoc WSN environment using only XOR and hash functions; it is, however, susceptible to various threats. Further, this protocol [7] incurs significant communication and storage costs, making it less usable in practical scenarios, as mentioned in [8]. Choi et al. [9] discuss a protocol based on WSNs for user authentication using ECC. Wu et al. [4] report that the protocol [9] suffers from the attacks that include offline password predicting, theft of user's identity, and stolen-verifier. Kumari and Om [10] discuss a remote user authentication for coal mining based on WSNs using lightweight functions like hash and EX-OR. Kumar et al. [11] report that the protocol [10] suffers from the lack of sensor anonymity, un-traceability, resilience to stolen smart card attack, resilience to denial of service attack, and resilience to stolen verifier attack. Ansari et al. [12] report that the protocol [11] does not resist the sensor node capture attack, stolen smart-card attack, user impersonation attack, and they introduce a protocol for safety monitoring using WSNs for the coal mining environment. A robust authentication protocol for healthcare based on WSNs is discussed by Kumar et al. [13], which is claimed to be resistant to all the active and passive attacks. He et al. [14] report that the protocol [13] suffers from the offline password predicting attack, privileged insider attack and lack of user anonymity. They introduce an authentication protocol for wireless medical sensor networks. Wu et al. [4] have recently discussed an authentication protocol for IoT based on WSNs. In this paper, we analyze the Wu et al.'s protocol [4] and find that it suffers from various attacks: privileged-insider, denial-of-service, sensor node capture, session-specific transient information, sensor node impersonation, man-in-the-middle, and gateway impersonation. We introduce a novel four-factor protocol that overcomes these security flaws.

1.2 Motivation

In the case of resource mining, some issues [15] need to be addressed, which include ensuring the safety of people working and the machinery deployed for the purpose as it could incur colossal loss, not only in terms of money but also in terms of human lives. Consider a scenario where the sensors gather the data about the concentration of the carbon monoxide gas in an underground mine. The sensors send the necessary probe during the critical conditions generally monitored by a user or a person monitoring the system conditions on the ground. When there is a condition that the gas is about to reach the threshold concentration, the user needs to send a warning to the people working in the underground mine for evacuation, follow the safety measures, and counteract to reduce the gas concentration. In such a situation, if some intruder morphs the gas concentration data such that it appears as a normal condition to the user, then the user will not send the warning message that could lead to an uncontrolled fire in the mine due to incorrect information supplied. Considering the same example, when the user is supposed to execute the safety protocol and call the emergency services; but, due to the attack, the user could not take any action or his actions are delayed, it could result in the loss of human lives and substantial monetary loss due to the machinery damage. For eavesdropping or threat to confidentiality, consider a firm that finds the availability of rare-earth elements or some other vital resources like coal, petroleum, etc., during mining. Suppose these details are leaked to an attacker, media, or competitive firms. In that case, it could delay the standard working procedure leading to monetary loss as many investors' money is in line. Likewise, there could be an attempt toward the forgery, impersonation, replay, denial of service, offline attacks, and other active and passive attacks, to disrupt the work of resource mining, which could end up in disaster to life and money in unimaginable way and extent. So, while designing an authentication protocol for such purposes, the security is a vital factor to be taken care of.

1.3 Contribution

Here, we summarize our contributions as follows.

1. We scrutinize the Wu et al.'s protocol [4] and find that it is not resistant to various attacks, namely, man-in-the-middle, privileged-insider, gateway impersonation, denial-of-service, sensor node capture, session-specific transient information, and sensor node impersonation.
2. The Wu et al.'s protocol [4] has some other issues like high communication cost, lack of proper confidentiality, key leakage, and lack of proper user anonymity.
3. We propose a four-factor robust and lightweight user authentication protocol for resource mining based on WSNs by overcoming the flaws of the protocol [4].
4. The formal verification using the BAN logic and RoR model of the proposed protocol is carried out to show that it preserves the secrecy of the session key.
5. We informally analyze the security of the proposed protocol to show its perseverance for active and passive attacks.
6. The proposed protocol is simulated using ProVerif to prove the secrecy of all the credentials and the communicating parties against all the known security threats.
7. Its comparative performance analysis with similar protocols is done to show its efficacy.

1.4 Threat Model

We assume the well-known Dolev–Yao (DY) [16] and Canetti and Krawczyk (CK) [17,18] adversarial model for our proposed protocol. We also discuss the assumptions considered in our protocol to evaluate its security.

1. An adversary $\mathbb{A}$ can overhear, morph, inject, replay, redirect, or delete the messages communicated over a public channel.
2. $\mathbb{A}$ cannot intercept the communications carried out over a private/secure channel [10].
3. $\mathbb{A}$ can pickpocket the smart device of a legal user of the system and obtains its secret credentials using the power analysis [19].
4. $\mathbb{A}$ can carry out an offline identity or password predicting attack separately. However, it cannot predict both parameters simultaneously in polynomial time.
5. $\mathbb{A}$ can obtain the data stored in the sensor's memory by physically capturing it and carrying out the power analysis.
6. $\mathbb{A}$ may be a legal user of the system (insider).

The remaining paper is arranged as follows. Section 2 reviews the Wu et al.'s protocol [4], and Sect. 3 provides its cryptanalysis. Section 4 introduces our proposed protocol, and Sect. 5 discusses its security analysis. The comparative performance analysis of the proposed protocol is provided in Sect. 6. Lastly, Sect. 7 concludes the paper.

2 Review of Wu et al.'s Protocol [4]

The Wu et al.'s protocol [4] broadly has two phases: registration and authentication phases. The symbols used in the protocol [4] are given in Table 1.

Table 1 Symbols used in Wu et al.'s Protocol [4]

Symbols	Meaning
SA	System administrator
U_i, S_j, GWN	ith User, jth Sensor, Gateway
$ID_i, PW_i, B_i, SCN_i, PID_i$	ith user's identity, password, biometric, smart card number, psuedo-identity
SID_j, ID_g, x	jth sensor's identity, Identity of gateway, Gateway's secret key
SK_u, SK_s, SK_g	user's, senor's and gateway's Session Keys
h(.), $\oplus$, $\|$, $BKG(.)$	Hash function, xor function, Concatenation, Biometric key recovery function
T_1, T_2, T_3, T_4 & ΔT	Timestamps & Transmission delay
$G_q, \mathbb{Z}_q^*$	Prime cyclic group, Multiplicative prime cyclic group

2.1 Registration Phase

In this phase, the remote user and sensors are registered with the system by executing the following steps.

2.1.1 User Registration Phase

1. User U_i sends his identity ID_i to SA.
2. SA calculates $B_1 = h(PID_i\|x)$ and $B_2 = h(SCN_i\|x)$, where SCN_i is smart card number (or identity) and PID_i is user's pseudo-identity, both chosen by SA. SA stores $\{B_1, B_2, PID_i, SCN_i, BKG(.)\}$ in the user's smart card and ID_i in its database, where $BKG(.)$ is biometric key recovery function.
3. U_i computes $C_0 = BKG(B_i)$, $HPW_i = h(C_0\|PW_i\|r_0)$, $C_1 = B_1 \oplus h(ID_i\|HPW_i)$, $C_2 = B_2 \oplus h(ID_i\|PW_i)$ and $C_3 = r_0 \oplus h(ID_i\|PW_i\|C_0)$, where B_i and r_0 are user's biometric and nonce, respectively. U_i replaces (B_1, B_2) with (C_1, C_2, C_3).

2.1.2 Sensor Registration Phase

1. Gateway GWN selects an identity SID_j and calculates secret key $x_j = h(SID_j\|x)$ for sensor S_j and sends $\{SID_j, x_j\}$ to S_j.
2. Sensor S_j keeps the credentials $\{SID_j, x_j\}$ into its memory.

2.2 Authentication Phase

1. User U_i inputs his credentials ID_i, PW_i, B_i^*; selects a nonce r_u, and timestamp T_1 to compute $C_0 = BKG(B_i^*)$, $r_0 = C_3 \oplus h(ID_i\|PW_i\|C_0)$, $HPW_i = h(C_0\|PW_i\|r_0)$, $B_1 = C_1 \oplus h(ID_i\|HPW_i)$, $B_2 = C_2 \oplus h(ID_i\|PW_i)$, $D_1 = B_1 \oplus r_u$, $D_2 = ID_i \oplus h(PID_i\|r_u\|T_1)$, $D_3 = SCN_i \oplus h(ID_i\|r_u\|T_1)$, $D_4 = SID_j \oplus h(B_2\|r_u\|T_1)$, and $D_5 = h(ID_i\|PID_i\|SCN_i\|r_u\|SID_j)$. U_i sends $M_1 = \{PID_i, D_1, D_2, D_3, D_4, D_5, T_1\}$ to GWN.
2. GWN checks if the received message is fresh and valid by verifying the similarity of ID_i as $B_1 = h(PID_i\|x)$, $r_u = D_1 \oplus B_1$, $ID_i = D_2 \oplus h(PID_i\|r_u\|T_1)$. If the message is fresh and valid, it computes $SCN_i = D_3 \oplus h(ID_i\|r_u\|T_1)$, $B_2 = h(SCN_i\|x)$, and $SID_j = D_4 \oplus h(B_2\|r_u\|T_1)$ to check if the received $D_5 = ?h(ID_i\|PID_i\|SCN_i\|r_u\|SID_j)$. If they match, the gateway computes $x_j = h(SID_j\|x)$, $D_6 = r_u \oplus h(ID_g\|x_j\|T_2)$, and $D_7 = h(r_u\|x_j\|SID_j)$, where T_2 is its current timestamp. GWN sends $M_2 = \{D_6, D_7, ID_g, T_2\}$ to S_j.
3. S_j verifies the timeliness and correctness of the received message by validating if $D_7 = ?h(r_u\|x_j\|SID_j)$. If the message is valid, it calculates $SK_s = h(r_u\|r_s)$, $D_8 = r_u \oplus r_s$, and $D_9 = h(SK_s\|x_j\|ID_g\|SID_j\|T_3)$, where r_s and T_3 are sensor's nonce and timestamp, respectively. S_j sends $M_3 = \{D_8, D_9, T_3\}$ to GWN.
4. GWN verifies the freshness and correctness of M_3 by verifying if $D_9 = ?h(SK_g\|x_j\|ID_g\|SID_j\|T_3)$. If the message is fresh and correct, it calculates $B_1^{new} = h(PID_i^{new}\|x)$, $D_{10} = B_1^{new} \oplus h(B_1\|r_u\|T_4)$, $D_{11} = PID_i^{new} \oplus h(B_1^{new}\|r_s\|T_4)$, and $D_{12} = h(SK_g\|B_1^{new}\|PID_i^{new}\|B_1\|ID_i\|SID_j)$, where PID_i^{new} and T_4 are random string and current timestamp of GWN, respectively. GWN sends $M_4 = \{D_8, D_{10}, D_{11}, D_{12}, T_4\}$ to U_i.
5. U_i verifies the timeliness of M_4 and calculates $r_s = D_8 \oplus r_u$, $B_1^{new} = D_{10} \oplus h(B_1\|r_u\|T_4)$, $PID_i^{new} = D_{11} \oplus h(B_1^{new}\|r_s\|T_4)$, and $SK_u = h(r_u\|r_s)$ to verify if $D_{12} = ?h(SK_u\|B_1^{new}\|PID_i^{new}\|B_1\|ID_i\|SID_j)$. If they match, (C_1, PID_i) is replaced with $(C_1^{new} = B_1^{new} \oplus h(ID_i\|HPW_i), PID_i^{new})$.

3 Cryptanalysis of Wu et al.'s Protocol [4]

Here, we cryptanalyze the Wu et al.'s protocol [4] and show that it suffers from various attacks as discussed below.

3.1 Privileged-Insider Attack

The privileged-insider attack [20] is performed during the mutual authentication phase, where the gateway acts as the insider, by executing the following steps:

1. Adversary $\mathbb{A}$ computes $B_1 = h(public\ PID_i \| own\ x)$ and $r_u = public\ D_1 \oplus calculated\ B_1$.
2. $\mathbb{A}$ evaluates $r_s = public\ D_8 \oplus calculated\ r_u$.
3. $\mathbb{A}$ finally computes the session key as $SK_s = SK_g = SK_u = h(calculated\ r_u \| calculated\ r_s)$.

Thus, this protocol suffers from the privileged-insider attack.

3.2 Man-in-the-Middle Attack

After capturing the sensor node and acquiring its stored parameters $\{SID_j, x_j\}$, the man-in-the-middle attack [21] in the login and authentication phase can be carried out by executing the following steps, where the man is considered in between the sensor node and the gateway.

1. Adversary $\mathbb{A}$ captures M_2 that is sent to the sensor node and selects r_{uM}, T_{2M}, and ID_{gM} as his own identity to calculate $D_{6M} = r_{uM} \oplus h(own\ ID_{gM} \| acquired\ x_j \| T_{2M})$ and $D_{7M} = h(r_{uM} \| acquired\ x_j \| acquired\ SID_j)$.
2. $\mathbb{A}$ constructs $M_{2M} = \{D_{6M}, D_{7M}, ID_{gM}, T_{2M}\}$ and forwards it to S_j, which performs the calculations based on the parameters received from $\mathbb{A}$. The modifications are nowhere to be caught, and the sensor calculates the session key as $SK_s = h(r_{uM} \| r_s)$.
3. $\mathbb{A}$ captures M_3 which is sent by the sensor to the gateway to calculate $r_s = public\ D_8 \oplus r_{uM}$ and $SK_s = SK_g = SK_u = h(r_{uM} \| r_s)$.
4. $\mathbb{A}$ picks r_{sM} and T_{3M} to calculate $r_u = public\ D_6 \oplus h(public\ ID_g \| acquired\ x_j \| public\ T_2)$, $SK_s = SK_g = SK_u = h(r_u \| r_{sM})$, $D_{8M} = r_u \oplus r_{sM}$, and $D_{9M} = h(SK_s \| acquired\ x_j \| public\ ID_g \| acquired\ SID_j \| T_{3M})$.
5. $\mathbb{A}$ constructs $M_{3M} = \{D_{8M}, D_{9M}, T_{3M}\}$ and sends it to the gateway, that performs the calculations based on the parameters received from $\mathbb{A}$. The modifications are nowhere to be caught.

We see that the presence of man amid the sensor and the gateway remains undetected; thus, this protocol suffers from the man-in-the-middle attack.

3.3 Sensor Node Capture Attack

For carrying out this attack, adversary $\mathbb{A}$ physically gets the hold of the sensor and performs power analysis to obtain the stored credential x_j. $\mathbb{A}$ calculates the secure session-key by carrying out the following steps:

1. $\mathbb{A}$ computes $r_u = public\ D_6 \oplus h(public\ ID_g \| acquired\ x_j \| public\ T_2)$ and $r_s = public\ D_8 \oplus calculated\ r_u$.
2. $\mathbb{A}$ evaluates session key as $SK_s = SK_g = SK_u = h(calculated\ r_u \| calculated\ r_s)$.

Thus, this protocol is not safe from the sensor node capture attack.

3.4 Sensor Node Impersonation Attack

After carrying out the sensor node capture attack, the values stored in the sensor $\{SID_j, x_j\}$ are obtained by power analysis. The sensor-node impersonation is performed during the login and authentication phase as follows:

1. Adversary $\mathbb{A}$ evaluates $r_u = public\ D_6 \oplus h(public\ ID_g \| acquired\ x_j \| public\ T_2)$ after eavesdropping M_2 sent to the sensor by the gateway.
2. $\mathbb{A}$ selects r_{sA} to compute $SK_s = h(calculated\ r_u \| r_{sA})$, $D_{8A} = calculated\ r_u \oplus r_{sA}$, and $D_{9A} = h(SK_s \| acquired\ x_j \| public\ ID_g \| acquired\ SID_j \| public\ T_3)$.
3. $\mathbb{A}$ composes $M_{3A} = \{D_{8A}, D_{9A}, T_3\}$ and sends it to the gateway, where the computations are done as per the altered values received from $\mathbb{A}$.
4. Similarly, $\mathbb{A}$ computes $r_s = public\ D_8 \oplus calculated\ r_u$ and $SK_s = SK_g = SK_u = h(r_u \| r_s)$.

Thus, this protocol is not safe from the sensor node impersonation attack.

3.5 Gateway Impersonation Attack

This attack [22] is carried out after obtaining the stored credentials $\{SID_j, x_j\}$ from the sensor node by carrying out the following steps in login and authentication phase of [4]:

1. Adversary $\mathbb{A}$ captures M_2, sent to the sensor node and selects r_{uA} to compute $D_{6A} = r_{uA} \oplus h(public\ ID_g \| acquired\ x_j \| public\ T_2)$ and $D_{7A} = h(r_{uA} \| acquired\ x_j \| acquired\ SID_j)$.
2. $\mathbb{A}$ composes $M_{2A} = \{D_{6A}, D_{7A}, ID_g, T_2\}$ and forwards it to the sensor node, that carries out the computations as per the altered values received from $\mathbb{A}$.
3. Similarly, $\mathbb{A}$ computes $r_s = public\ D_8 \oplus r_{uA}$, $r_u = public\ D_6 \oplus h(public\ ID_g \| acquired\ x_j \| public\ T_2)$, and $SK_s = SK_g = SK_u = h(r_u \| r_s)$.

Thus, we see that this protocol is not resilient against the gateway impersonation attack.

3.6 Denial-of-Service Attack

If D_5 is altered by adversary $\mathbb{A}$, it gets detected at the gateway after computing 6 hash, 4 exclusive OR, 1 subtraction, 1 inequality, and a search operation on the database. After so many calculations and consumption of resources [23], a genuine user is found to be illegal due to the existence of $\mathbb{A}$.

Hence, a legitimate user cannot access the system and the resources are exhausted by the repeated trial due to the overflow of bogus authentication requests [23]. The services are denied to a genuine user of the system. Thus, this protocol cannot resist the denial-of-service attack.

3.7 Session Specific Transient Information Attack

If the temporary variable r_u somehow gets exposed to adversary $\mathbb{A}$, it can calculate $r_s = public\, D_8 \oplus exposed\, r_u$ and the session key $SK_u = SK_g = SK_s = h(exposed\, r_u \,|calculated\, r_s)$. Similarly, if r_s is exposed to adversary $\mathbb{A}$, it can calculate $r_u = public\, D_8 \oplus exposed\, r_s$ and the session key $SK_u = SK_g = SK_s = h(calculated\, r_u \| exposed\, r_s)$. Thus, this protocol does not resist the session specific transient information attack.

3.8 Lack of Confidentiality

As discussed in steps 3–5 of login and authentication phase in section 2, all the communicating entities share similar session key $SK_s = SK_g = SK_u = h(r_u \| r_s)$. Hence, any confidential message sent by any sender to any receiver of the system could be read or altered by the third communicating entity of the system. So, this protocol lacks confidentiality.

3.9 Key Leakage

As discussed in step 2 of subsection 2.1.1 in user registration, the system administrator SA is found using the secret key x of the gateway for calculating B_1 and B_2. Any entity should not know the other entities' secret or private key as per DY threat model [16]. Hence, this protocol suffers from the key leakage.

3.10 Lack of User Anonymity

The system administrator SA stores the user's identity (ID_i) in raw format in its memory as discussed in step 1 of subsection 2.1.1 in user registration. As ID_i is stored in raw form, there is clear lack of user anonymity in this protocol. In case, the table from SA is leaked, the stolen verifier attack can be performed.

4 Proposed Protocol

Our proposed protocol consists of four factors for authenticating a user, among which one is the user's heartbeat, which is unique for each person. The user's heartbeat conversion to bitstring value takes place via the ADC function. The ADC function takes the user's heartbeat via electrodes as input, converts it into the analog format, and the analog result into digital format of 12 bits at 1MHz frequency. The specifications of the hardware and the details of operations, where ECG signals are delineated into different wave forms for feature extraction till the conversion to digital form with the power consumption of 7.47μw, are given in [24]. The conversion time for any ADC converter is given as the product of the clock period and the number of bits converted. Hence, for 12 bits of the converted digital data at 1 MHz, it takes 0.012 msec [25].

Our proposed protocol consists of five phases: initialization, user registration, sensor registration, user login, and mutual authentication. First, the system administrator (SA) initializes the parameters for all the communicating entities. Then the user and sensor nodes establish their public-private key pair after completing their respective registration phase. After becoming the legal part of the system, a user can log into the system to authenticate itself and access the sensor's data. The other communicating entities (gateway and sensor) get authenticated during the authentication process to ensure robust security. After successful authentication among all the communicating entities, the user and the sensor establish a secure session key for future data exchange. The notations used in our protocol are given in Table 2.

4.1 Initialization

System administrator SA selects the following parameters for the participating entities U_i, GWN_j, and SN_k to initialize the system:

1. Identity SID_k, GID_j for sensor node and gateway, respectively.
2. Pre-secret key L_{kj} for sensor node and gateway, which is stored in their memories.
3. Pre-secret key K_{ij} for user and gateway, which is stored in the smart device and memory, respectively.
4. An asymmetric encryption and decryption algorithm.
5. A standard elliptic curve E over F_q.
6. A group generator P over E/F_q of order n.

The gateway selects its private key $G\alpha_j$ to compute its public key $G\beta_j = G\alpha_j.P$.

4.2 User Registration

User U_i first registers with the gateway GWN_j to become a legit part of the system and establish his public, private key pair for further use by performing the following steps.

1. U_i selects his identity UID_i, password PW_i, punches biometric fingerprint BIO_i, and records his heartbeat HB_i over the electrodes to calculate the digital data from the heartbeat as $H_i = ADC(HB_i)$. U_i calculates

Table 2 Symbols used in our protocol

Symbols	Meaning
SA, $\mathbb{A}$	System administrator, Attacker/Adversary
U_i, PW_i, UID_i, BIO_i, HB_i	ith User, his password, identity, biometric and heartbeat
SN_k, SID_k	kth Sensor, its identity
GWN_j, GID_j	jth Gateway, its identity
b_i, c_j, e_j, d_k	Random numbers of ith user, jth gateway and kth sensor
$K_{ij}\&L_{kj}$	Pre-secret key amid user-gateway and sensor-gateway
$E(.)\&D(.)$	Cryptographic asymmetric encryption and decryption functions
$h(.)$, $\|$, $ADC(.)$	Hash, Concatenation, Analog to digital conversion
T_{1i}, T_{2j}, T_{3k}, T_{4j}, T_{5i}, & ΔT	Timestamps & Acceptable transmission delay
SK_{ik}	Session key amid user-sensor
q, F_q, E/F_q	Large Prime, Prime finite field, Elliptic curve E over F_q
G_q, $\mathbb{Z}_q^*$	Prime cyclic group, Multiplicative prime cyclic group

$A_{1i} = h(UID_i\|PW_i\|BIO_i\|H_i)$, that is used in login process.

2. U_i selects $b_i \in \mathbb{Z}_q^*$ to calculate $B_i = b_i.P$, $N_{1i} = h(GID_j\|G\beta_j\|K_{ij}\|B_i)$, $N_{2i} = A_{1i}.P$, and $A_{2i} = h(N_{1i}\|N_{2i}\|K_{ij})$. U_i forwards a fresh registration request to GWN_j as $\{A_{2i}, N_{2i}, B_i\}$.
3. GWN_j obtains A_{2i}, N_{2i}, B_i to verify the user by checking if the received $A_{2i} = ?A'_{2i}$ by calculating $N'_{1i} = h(GID_j\|G\beta_j\|K_{ij}\|received\ B_i)$ and $A'_{2i} = h(N'_{1i}\|received\ N_{2i}\|K_{ij})$. If the values do not match, the connection is terminated.
4. GWN_j chooses $c_j \in \mathbb{Z}_q^*$ to calculate $C_j = c_j.P + K_{ij}.P$, $A_{3j} = h(GID_j\|received\ A_{2i}\|C_j)$, and $A_{4j} = h(A_{3j}\|K_{ij}\|received\ B_i)$. GWN_j stores A_{2i} and sends the reply message containing $\{A_{3j}, A_{4j}\}$ to U_i.
5. U_i evaluates $A'_{4j} = h(received\ A_{3j}\|K_{ij}\|B_i)$ to check if $A'_{4j} =?received\ A_{4j}$. If the parameters match, U_i constructs his private key $U\alpha_i = h(b_i\|A_{1i}\|received\ A_{3j})$ and public key $U\beta_i = U\alpha_i.P$. Finally, U_i stores $\{A_{2i}, A_{1i}\}$.

This process and flowchart for the same are depicted in Figs. 2 and 3a, respectively.

4.3 Sensor Node Registration

The sensor nodes SN_k also need to register with the gateway GWN_j to become a legit part of the system and establish its public, private key pair for further use by performing the following steps.

1. SN_k chooses $d_k \in \mathbb{Z}_q^*$ to compute $D_k = d_k.P$, $A_{5k} = h(SID_k\|G\beta_j)$, and $A_{6k} = h(A_{5k}\|D_k\|L_{kj})$. SN_k sends the registration request containing $\{A_{5k}, A_{6k}, D_k\}$ to GWN_j.
2. GWN_j verifies the authenticity of SN_k by calculating $A'_{6k} = h(received\ A_{5k}\|received\ D_k\|L_{kj})$ to evaluate if $A'_{6k} =?received\ A_{6k}$. If the values do not agree, the session is terminated.
3. GWN_j picks $e_j \in \mathbb{Z}_q^*$ to evaluate $E_j = e_j.P + L_{jk}.P$, $A_{7j} = h(GID_k\|E_j\|received\ A_{5k})$, and $A_{8j} = h(A_{7j}\|received\ A_{6k}\|received\ D_k\|L_{kj})$. It stores A_{5k} and sends the reply message comprising of $\{A_{7j}, A_{8j}\}$ to SN_k.
4. SN_k computes $A'_{8j} = h(received\ A_{7j}\|A_{6k}\|D_k\|L_{kj})$ to verify if $A'_{8j} =?received\ A_{8j}$. If both the values are same, then SN_k constructs its private key $S\alpha_k = h(d_k\|A_{5k}\|received\ A_{7j})$ and public key $S\beta_k = S\alpha_k.P$. Finally, SN_k stores A_{5k}.

Figure 3b gives the workflow of sensor registration phase, and Fig. 4 summarizes this process.

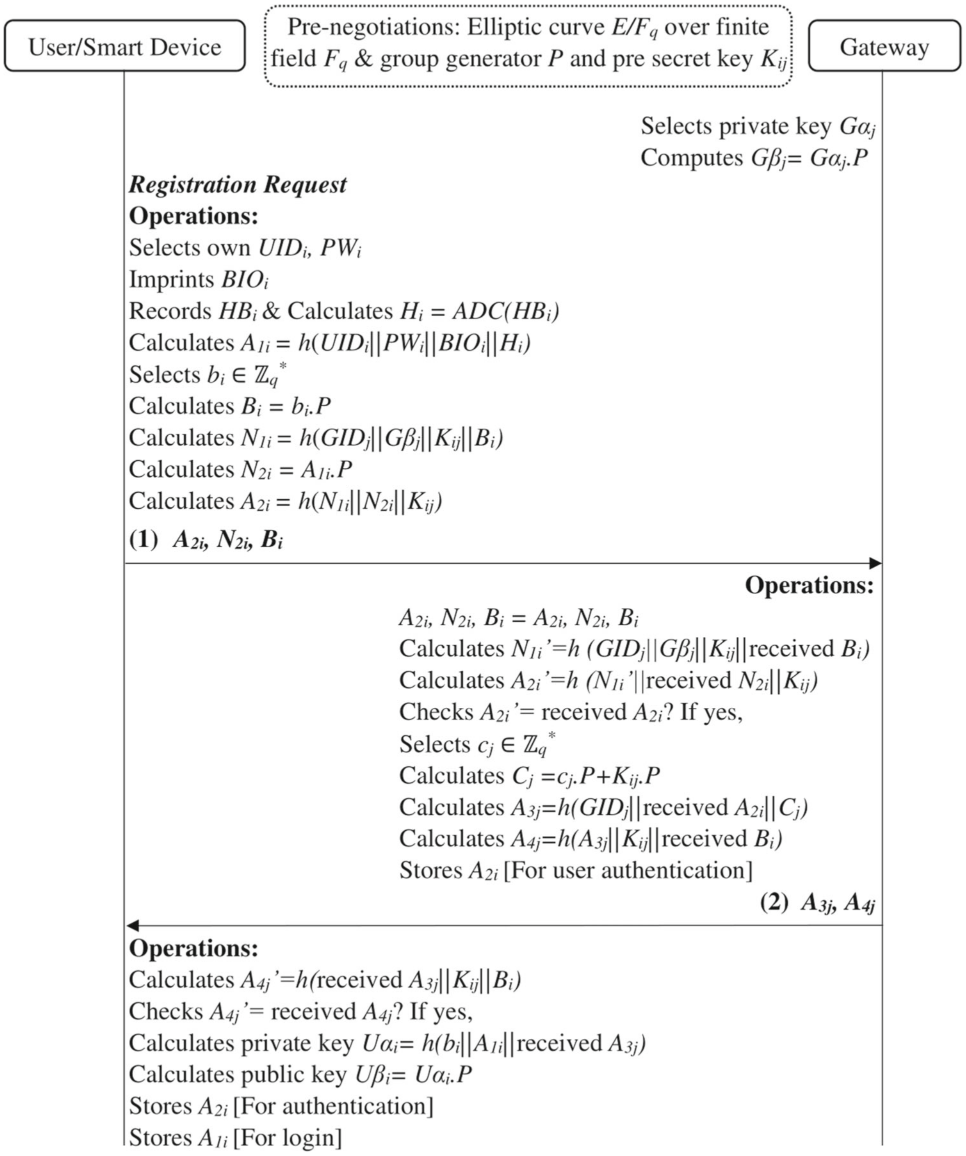
4.4 Remote User Login

Remote user U_i monitoring the conditions on the ground needs to log into the system after successful registration for accessing the system by performing the following steps:

1. U_i inputs his credentials UID_i, PW_i, BIO_i, HB_i into the smart device as a login request message.
2. Smart device calculates $H_i = ADC(HB_i)$ and $A'_{1i} = h(received\ UID_i\|received\ PW_i\|received\ BIO_i\|H_i)$ to check if $A'_{1i} =?stored\ A_{1i}$. If the values match, the user is a genuine registered user and he is given access to the system for further use; otherwise, the connection is terminated.

Figure 5 schematically shows the process of login, and its workflow is depicted in Fig. 6.

Fig. 2 User registration

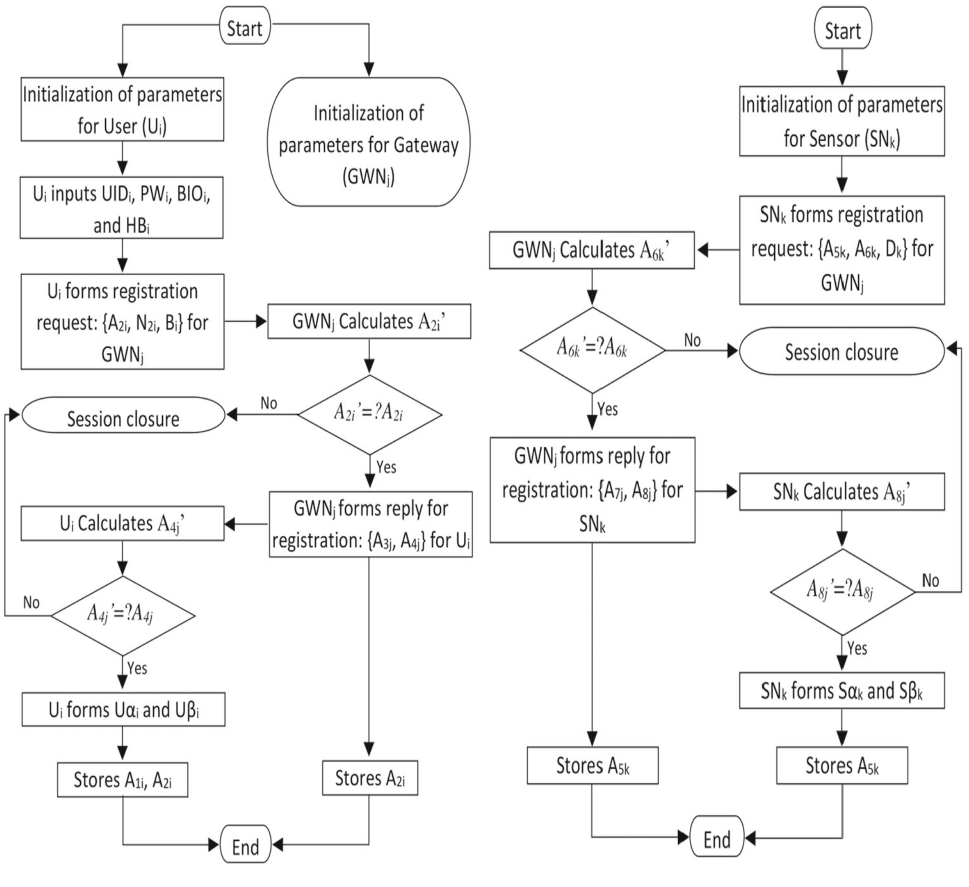

User/Smart Device
Pre-negotiations: Elliptic curve E/F_q over finite field F_q & group generator P and pre secret key K_{ij}
Gateway
Selects private key $G\alpha_j$
Computes $G\beta_j = G\alpha_j.P$
Registration Request
Operations:
Selects own UID_i, PW_i
Imprints BIO_i
Records HB_i & Calculates $H_i = ADC(HB_i)$
Calculates $A_{1i} = h(UID_i||PW_i||BIO_i||H_i)$
Selects $b_i \in \mathbb{Z}_q^*$
Calculates $B_i = b_i.P$
Calculates $N_{1i} = h(GID_j||G\beta_j||K_{ij}||B_i)$
Calculates $N_{2i} = A_{1i}.P$
Calculates $A_{2i} = h(N_{1i}||N_{2i}||K_{ij})$
(1) A_{2i}, N_{2i}, B_i
Operations:
A_{2i}, N_{2i}, B_i = A_{2i}, N_{2i}, B_i
Calculates $N_{1i}' = h(GID_j||G\beta_j||K_{ij}||$received $B_i)$
Calculates $A_{2i}' = h(N_{1i}'||$received $N_{2i}||K_{ij})$
Checks A_{2i}' = received A_{2i}? If yes,
Selects $c_j \in \mathbb{Z}_q^*$
Calculates $C_j = c_j.P + K_{ij}.P$
Calculates $A_{3j} = h(GID_j||$received $A_{2i}||C_j)$
Calculates $A_{4j} = h(A_{3j}||K_{ij}||$received $B_i)$
Stores A_{2i} [For user authentication]
(2) A_{3j}, A_{4j}
Operations:
Calculates $A_{4j}' = h($received $A_{3j}||K_{ij}||B_i)$
Checks A_{4j}' = received A_{4j}? If yes,
Calculates private key $U\alpha_i = h(b_i||A_{1i}||$received $A_{3j})$
Calculates public key $U\beta_i = U\alpha_i.P$
Stores A_{2i} [For authentication]
Stores A_{1i} [For login]

4.5 Mutual Authentication

The user and sensor authenticate each other via the gateway and establish a session key for secure data communication. After then, the user could access the data of the field from the sensor as and when needed, and the sensor could send alert messages in case of an emergency by performing the following steps.

1. U_i calculates $I_{1ij} = h(stored\ A_{2i} \| K_{ij} \| U\beta_i)$ and $I_{2ij} = h(I_{1ij} \| T_{1i} \| B_i)$, where T_{1i} is his current timestamp. U_i sends the authentication request packet $Pckt_1 = \{I_{2ij}, B_i, T_{1i}\}$ to GWN_j.
2. GWN_j checks if the received request is afresh by ensuring that the packet has been received within the threshold limit of packet reception time ΔT. If it is fresh, it calculates $I'_{1ij} = h(stored\ A_{2i} \| K_{ij} \| U\beta_i)$ and $I'_{2ij} = h(I'_{1ij} \| received\ T_{1i} \| received\ B_i)$ to authenticate U_i by checking $I'_{2ij} =?received\ I_{2ij}$. If both the parameters are equal, then U_i is authenticated.
3. GWN_j calculates $I_{3jk} = h(stored\ A_{5k} \| L_{kj} \| G\beta_j)$ and $I_{4jk} = h(I_{3jk} \| T_{2j} \| received\ B_i)$, where T_{2j} is its current timestamp. GWN_j sends a fresh authentication request data packet $Pckt_2 = \{I_{4jk}, B_i, T_{2j}\}$ to SN_k.
4. SN_k ensures the timeliness of the received request by checking that the packet has been received within ΔT time limit. If the timeliness is ensured, it computes $I'_{3jk} = h(stored\ A_{5k} \| L_{kj} \| G\beta_j)$ and $I'_{4jk} = h(I'_{3ji} \| received\ T_{2j} \| received\ B_i)$ to validate the gateway by verifying if $I'_{4jk} =?received\ I_{4jk}$. If the condition is verified, GWN_j is authenticated.

Fig. 3 Flowchart of registration phases


Start
Initialization of parameters for User (Ui)
Initialization of parameters for Gateway (GWNj)
Ui inputs UIDi, PWi, BIOi, and HBi
Ui forms registration request: {A2i, N2i, Bi} for GWNj
GWNj Calculates A2i'
A2i'=?A2i
No
Session closure
Yes
GWNj forms reply for registration: {A3j, A4j} for Ui
Ui Calculates A4j'
A4j'=?A4j
No
Yes
Ui forms Uαi and Uβi
Stores A1i, A2i
Stores A2i
End
Start
Initialization of parameters for Sensor (SNk)
SNk forms registration request: {A5k, A6k, Dk} for GWNj
GWNj Calculates A6k'
A6k'=?A6k
No
Session closure
Yes
GWNj forms reply for registration: {A7j, A8j} for SNk
SNk Calculates A8j'
A8j'=?A8j
No
Yes
SNk forms Sαk and Sβk
Stores A5k
Stores A5k
End

(a) Workflow of user and gateway **(b)** Workflow of sensor and gateway

5. SN_k computes $I_{5kj} = h(GID_j \| L_{kj} \| S\beta_k \| I'_{3jk})$ and $I_{6kj} = h(I_{5kj} \| T_{3k} \| B_i \| D_k)$, where T_{3k} is its current timestamp. SN_k sends a new authentication request data packet $Pckt_3 = \{I_{6kj}, D_k, T_{3k}\}$ to GWN_j.
6. GWN_j checks the freshness of the authentication request by assuring that the reception is within ΔT limit. GWN_j evaluates $I'_{5kj} = h(GID_j \| L_{kj} \| S\beta_k \| I_{3jk})$ and $I'_{6kj} = h(I'_{5kj} \| received\, T_{3k} \| B_i \| received\, D_k)$ to authenticate SN_k by the condition $I'_{6kj} =?I_{6kj}$. If the condition holds, SN_k is authenticated.
7. GWN_j evaluates $I_{7ji} = h(stored\, A_{2i} \| K_{ij} \| G\beta_j)$ and $I_{8ji} = h(I_{7ji} \| T_{4j} \| received\, D_k)$, where T_{4j} is its current timestamp. GWN_j sends afresh authentication request data packet $Pckt_4 = \{I_{8ji}, D_k, T_{4j}\}$ to U_i.
8. U_i checks the freshness of the authentication request by verifying that the reception is within ΔT limit. U_i computes $I'_{7ji} = h(stored\, A_{2i} \| K_{ij} \| G\beta_j)$ and $I'_{8ji} = h(I'_{7ji} \| received\, T_{4j} \| received\, D_k)$ to validate GWN_j by verifying if $I'_{8ji} =?I_{8ji}$. If the condition is verified, GWN_j is authenticated. U_i finally calculates the session key $SK_{ik} = b_i.S\beta_k + D_k.U\alpha_i$. Similarly, the sensor node calculates the same session-key as $SK_{ik} = d_k.U\beta_i + B_i.S\alpha_k$. The user and sensor can directly exchange the data by using the freshly formulated session-key SK_{ik}.

Figure 6 shows the workflow of mutual authentication phase, and the complete process is summarized in Fig. 7.

5 Security Analysis

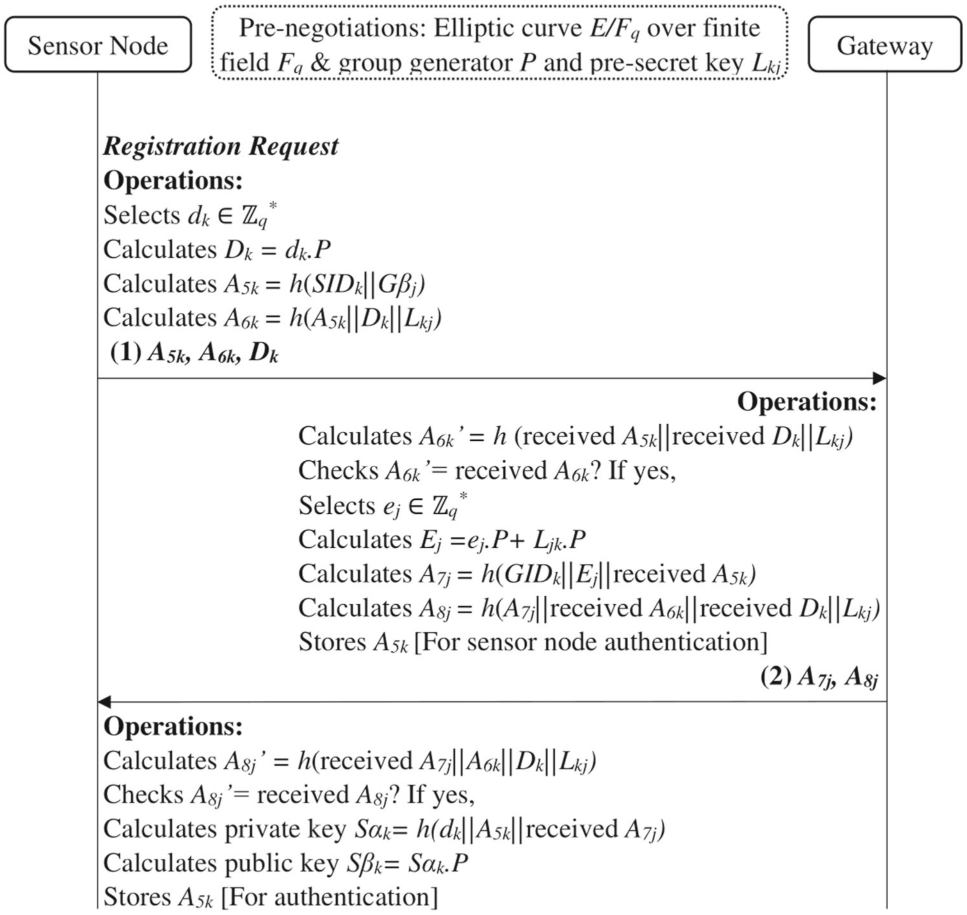
In this section, we discuss the security analysis of the proposed protocol that includes security attributes, formal validation via BAN and by ROR model, and simulations using ProVerif.

5.1 Security Attributes

Here, we discuss the security features provided by our protocol, as discussed below.

a) *It provides secure multi-factor mutual authentication.* The proposed protocol has identity (UID_i), password (PW_i), biometric (BIO_i) and heartbeat (HB_i) as four security factors for user U_i. For an attacker $\mathbb{A}$ to breach into the system, $\mathbb{A}$ would need to create $Pckt_1 =$

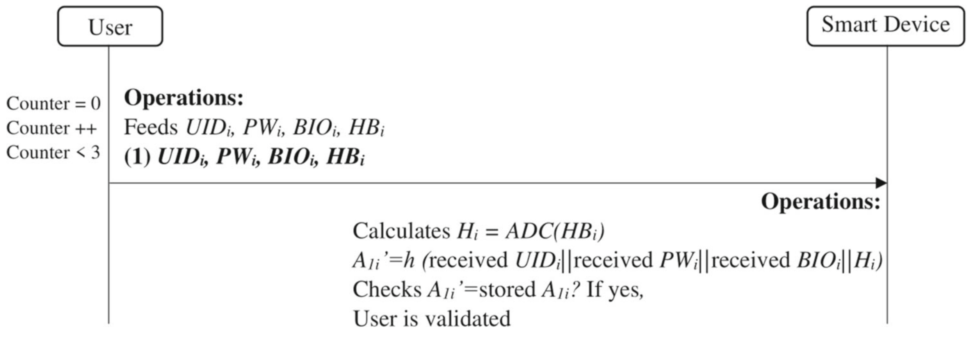
Fig. 4 Sensor node registration


Sensor Node
Pre-negotiations: Elliptic curve E/F_q over finite field F_q & group generator P and pre-secret key L_{kj}
Gateway
Registration Request
Operations:
Selects $d_k \in \mathbb{Z}_q^*$
Calculates $D_k = d_k.P$
Calculates $A_{5k} = h(SID_k||G\beta_j)$
Calculates $A_{6k} = h(A_{5k}||D_k||L_{kj})$
(1) A_{5k}, A_{6k}, D_k
Operations:
Calculates $A_{6k}' = h$ (received A_{5k}||received $D_k||L_{kj}$)
Checks A_{6k}'= received A_{6k}? If yes,
Selects $e_j \in \mathbb{Z}_q^*$
Calculates $E_j = e_j.P + L_{jk}.P$
Calculates $A_{7j} = h(GID_k||E_j||$received $A_{5k})$
Calculates $A_{8j} = h(A_{7j}||$received $A_{6k}||$received $D_k||L_{kj})$
Stores A_{5k} [For sensor node authentication]
(2) A_{7j}, A_{8j}
Operations:
Calculates $A_{8j}' = h$(received $A_{7j}||A_{6k}||D_k||L_{kj})$
Checks A_{8j}'= received A_{8j}? If yes,
Calculates private key $S\alpha_k = h(d_k||A_{5k}||$received $A_{7j})$
Calculates public key $S\beta_k = S\alpha_k.P$
Stores A_{5k} [For authentication]

$\{I_{2ij}, B_i, T_{1i}\}$. Constructing this message by morphing, eavesdropping, or accidental exposure of the secret data is not possible because I_{2ij} is composed of I_{1ij} that contains two secret parameters A_{2i}, K_{ij}, and predicting two security parameters in polynomial time is impractical, where K_{ij} is the pre-secret key that is known only to the user and gateway, thus making its knowledge unavailable to $\mathbb{A}$. Moreover, A_{2i} itself is a complex arrangement of four security parameters hashed, scalar point-multiplied, and hashed again. Hence, construction of A_{2i} is infeasible, which makes the authentication request manifold complex and infeasible. Additionally, out of four security factors, the two factors being the biometric credentials of the user are impossible to forge. Thus, our protocol assures secure multi-factor mutual authentication.

b) *It is resistant against the stolen smart device attack.* In case the smart device is pick-pocketed by an attacker $\mathbb{A}$, he can get A_{1i} and A_{2i} stored in it during the registration process. For carrying out the attack, $\mathbb{A}$ needs to generate a valid authentication request packet $Pckt_1 = \{I_{2ij}, B_i, T_{1i}\}$. But, the generation of $Pckt_1$ is infeasible due to the complex creation of I_{2ij} as discussed in point 'a', where $I_{2ij} = h(I_{1ij}\|T_{1i}\|B_i)$, $I_{1ij} = h(stored\ A_{2i}\|K_{ij}\|U\beta_i)$, $A_{2i} = h(N_{1i}\|N_{2i}\|K_{ij})$, $N_{1i} =$

Fig. 5 Remote user login


User
Smart Device
Counter = 0
Counter ++
Counter < 3
Operations:
Feeds UID_i, PW_i, BIO_i, HB_i
(1) UID_i, PW_i, BIO_i, HB_i
Operations:
Calculates $H_i = ADC(HB_i)$
$A_{1i}'=h$ (received UID_i||received PW_i||received $BIO_i||H_i$)
Checks A_{1i}'=stored A_{1i}? If yes,
User is validated

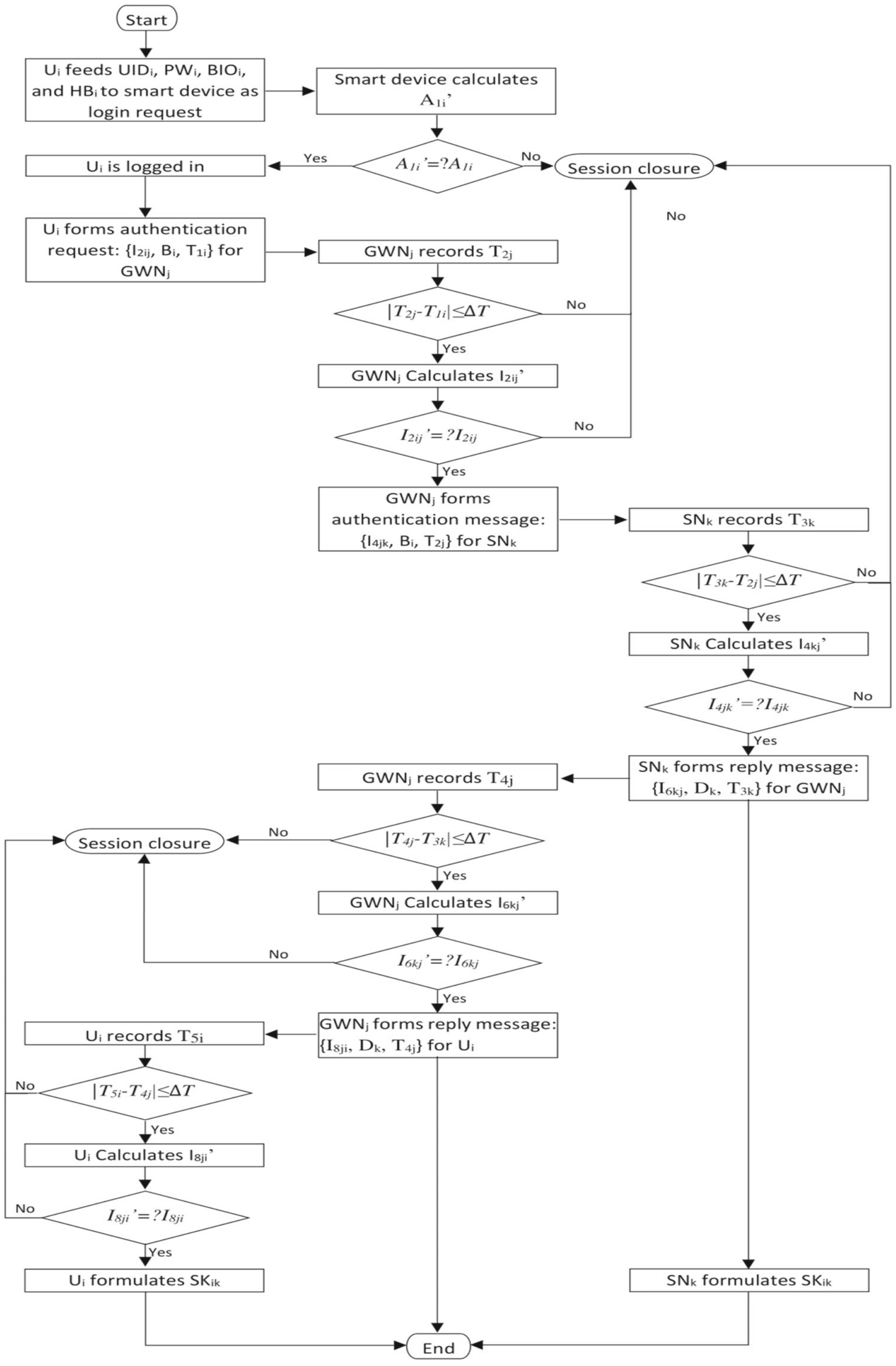
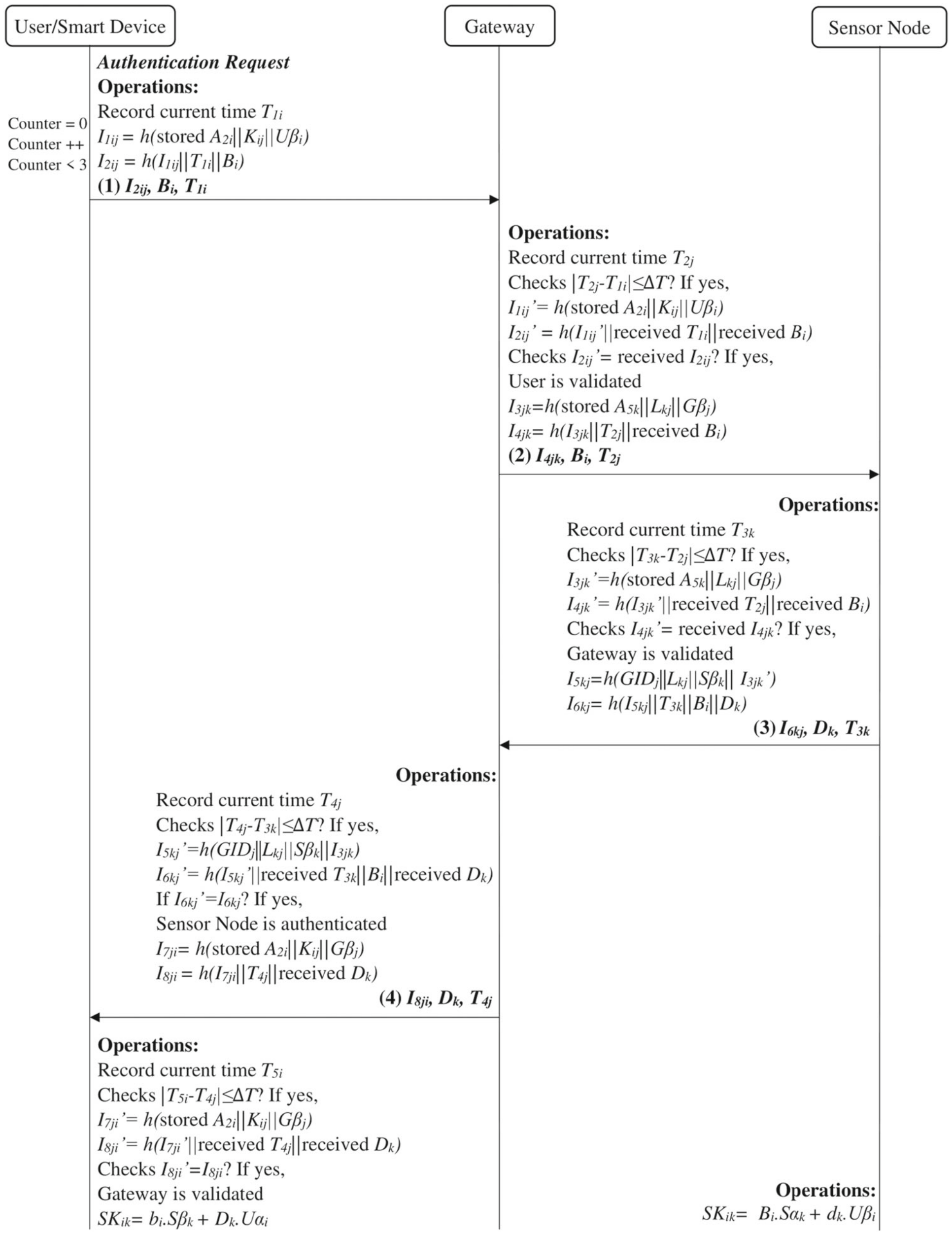


Fig. 6 Flowchart of login and authentication phase

$h(GID_j \| G\beta_j \| K_{ij} \| B_i), N_{2i} = A_{1i}.P, A_{1i} = h(UID_i \| PW_i \| BIO_i \| H_i)$, and $H_i = ADC(HB_i)$. Besides the complex formation of I_{2ij}, I_{1ij} contains the pre-secret key that is shared amidst the user and gateway in hashed format. Hence, the formation of I_{1ij} itself is intricate that makes it impossible to generate further parameters. Thus, the proposed protocol provides strong counteraction against the stolen smart device attack.

c) *It is resistant against the remote user impersonation attack.* Attacker $\mathbb{A}$ cannot impersonate [26] a user in our protocol because for that $\mathbb{A}$ needs to forge the message $Pckt_1 = \{I_{2ij}, B_i, T_{1i}\}$. If $\mathbb{A}$ overhears the message sent on a public channel in the authentication process, $\mathbb{A}$

Fig. 7 Mutual authentication


User/Smart Device
Gateway
Sensor Node
Authentication Request
Operations:
Record current time T_{1i}
Counter = 0
Counter ++
Counter < 3
$I_{1ij} = h(\text{stored } A_{2i}||K_{ij}||U\beta_i)$
$I_{2ij} = h(I_{1ij}||T_{1i}||B_i)$
(1) I_{2ij}, B_i, T_{1i}
Operations:
Record current time T_{2j}
Checks $|T_{2j}-T_{1i}|\leq\Delta T$? If yes,
$I_{1ij}' = h(\text{stored } A_{2i}||K_{ij}||U\beta_i)$
$I_{2ij}' = h(I_{1ij}'||\text{received } T_{1i}||\text{received } B_i)$
Checks I_{2ij}'= received I_{2ij}? If yes,
User is validated
$I_{3jk}=h(\text{stored } A_{5k}||L_{kj}||G\beta_j)$
$I_{4jk}= h(I_{3jk}||T_{2j}||\text{received } B_i)$
(2) I_{4jk}, B_i, T_{2j}
Operations:
Record current time T_{3k}
Checks $|T_{3k}-T_{2j}|\leq\Delta T$? If yes,
$I_{3jk}'=h(\text{stored } A_{5k}||L_{kj}||G\beta_j)$
$I_{4jk}'= h(I_{3jk}'||\text{received } T_{2j}||\text{received } B_i)$
Checks I_{4jk}'= received I_{4jk}? If yes,
Gateway is validated
$I_{5kj}=h(GID_j||L_{kj}||S\beta_k|| I_{3jk}')$
$I_{6kj}= h(I_{5kj}||T_{3k}||B_i||D_k)$
(3) I_{6kj}, D_k, T_{3k}
Operations:
Record current time T_{4j}
Checks $|T_{4j}-T_{3k}|\leq\Delta T$? If yes,
$I_{5kj}'=h(GID_j||L_{kj}||S\beta_k||I_{3jk})$
$I_{6kj}'= h(I_{5kj}'||\text{received } T_{3k}||B_i||\text{received } D_k)$
If $I_{6kj}'=I_{6kj}$? If yes,
Sensor Node is authenticated
$I_{7ji}= h(\text{stored } A_{2i}||K_{ij}||G\beta_j)$
$I_{8ji} = h(I_{7ji}||T_{4j}||\text{received } D_k)$
(4) I_{8ji}, D_k, T_{4j}
Operations:
Record current time T_{5i}
Checks $|T_{5i}-T_{4j}|\leq\Delta T$? If yes,
$I_{7ji}'= h(\text{stored } A_{2i}||K_{ij}||G\beta_j)$
$I_{8ji}'= h(I_{7ji}'||\text{received } T_{4j}||\text{received } D_k)$
Checks $I_{8ji}'=I_{8ji}$? If yes,
Gateway is validated
$SK_{ik}= b_i.S\beta_k + D_k.U\alpha_i$
Operations:
$SK_{ik}= B_i.S\alpha_k + d_k.U\beta_i$

would need to alter the contents of $Pckt_1$, which is not feasible. Altering I_{2ij} would be detected by the gateway during the integrity check as it contains two more public parameters B_i and T_{1i} along with I_{1ij} in the irreversible hashed format, where $I_{1ij} = h(stored\ A_{2i}\|K_{ij}\|U\beta_i)$ and K_{ij} is the pre-secret key. Changing B_i or T_{1i} would directly alter the hash value of I_{2ij}, and it would be caught at the gateway. Additionally, a slight variation in the timestamp would result in the failed conditional check over the freshness of the message as per the packet delay threshold limit ΔT. Thus, our protocol assures proper counteraction against the user impersonation attack.

d) *It is resistant against the gateway impersonation attack.* During authentication if attacker $\mathbb{A}$ tries to impersonate the gateway, it would need to forge either $Pckt_2 = \{I_{4jk}, B_i, T_{2j}\}$ or $Pckt_4 = \{I_{8ji}, D_k, T_{4j}\}$. For impersonating by $Pckt_2$, $\mathbb{A}$ would need to alter I_{4jk}, where $I_{4jk} = h(I_{3ji}\|T_{2j}\|B_i)$, $I_{3jk} = h(A_{5k}\|L_{kj}\|G\beta_j)$, $A_{5k} = h(SID_k\|G\beta_j)$ and L_{kj} is the pre-secret of the sensor and gateway. Any modification in I_{4jk} would be detected by the sensor during the integrity check as it contains two other public parameters B_i, T_{2j}, and also I_{3jk} is composed of the parameter of the sensor itself. Morphing B_i or T_{2j} changes I_{4jk}, which can easily be detected by the sensor during the integrity check. Moreover, a slight deviation in T_{2j} will cause the failed inequality check $|T_{3k} - T_{2j}| \leq ?\Delta T$. For impersonating via $Pckt_4$, $\mathbb{A}$ has to alter I_{8ji}, where $I_{8ji} = h(I_{7ji}\|T_{4j}\|D_k)$, $I_{7ji} = h(A_{2i}\|K_{ij}\|G\beta_j)$, $A_{2i} = h(N_{1i}\|N_{2i}\|K_{ij})$, $N_{1i} = h(GID_j\|G\beta_j\|K_{ij}\|B_i)$, $A_{1i} = h(UID_i\|PW_i\|BIO_i\|H_i)$, $H_i = ADC(HB_i)$, $N_{2i} = A_{1i}.P$ and K_{ij} is the pre-secret key of the user and gateway. Morphing I_{8ji} will get detected by the user as it is composed of D_k, T_{4j} and I_{8ji} is formulated from the parameter of the user itself. Altering D_k or T_{4j} changes I_{8ji}, which can easily be detected by the user. Also, variation in T_{4j} will result in the failed inequality check. So, neither $Pckt_2 = \{I_{4jk}, B_i, T_{2j}\}$ nor $Pckt_4 = \{I_{8ji}, D_k, T_{4j}\}$ can be forged. Thus, our protocol ensures resilience against the gateway impersonation attack.

e) *It is resistant against the sensor node impersonation attack.* For carrying out the sensor node impersonation attack, attacker $\mathbb{A}$ has to alter the authentication request message $Pckt_3 = \{I_{6kj}, D_k, T_{3k}\}$. For this, $\mathbb{A}$ has to alter I_{6kj} that is composed as $I_{6kj} = h(I_{5kj}\|T_{3k}\|B_i\|D_k)$, $I_{5kj} = h(GID_j\|L_{kj}\|S\beta_k\|I'_{3kj})$, $I'_{3kj} = h(stored\, A_{5k}\|L_{kj}\|G\beta_j)$ and L_{kj} is the pre-secret key of the sensor and gateway. When there is any variation encountered in I_{6kj}, it will be detected by the gateway while performing the integrity check as the variable contains three public parameters B_i, D_k, and T_{3k}. Moreover, I_{5kj} is formulated by using the parameters of both the sensor and gateway. Additionally, changing D_k, B_i or T_{3k} will alter the value of I_{6kj}, which cannot go undetected at the gateway. Also, slight deviation in T_{3k} will result in the malfunctioned check $|T_{4j} - T_{3k}| \leq ?\Delta T$ at the gateway. Hence, our protocol ensures resilience against the sensor-node impersonation attack.

f) *It is resistant against the man-in-the-middle attack.* Consider a man amidst the user and the gateway. Attacker $\mathbb{A}$ generates its own $Pckt_{1M} = I_{2ijM}, B_{iM}, T_{1iM}$ by calculating $I_{2ijM} = h(I_{ijM}\|T_{1iM}\|B_{iM})$, $I_{1ijM} = h(A_{2iM}\|K_{ijM}\|U\beta_{iM})$, $A_{2iM} = h(N_{1iM}\|N_{2iM}\|K_{ijM})$, $N_{1iM} = h(GID_j\|G\beta_j\|K_{ijM}\|B_{iM})$, $A_{1iM} = h(UID_{iM}\|PW_{iM}\|BIO_{iM}\|H_{iM})$, $H_{iM} = ADC(HB_{iM})$ and $N_{2iM} = A_{1iM}.P$. Even if $\mathbb{A}$ generates all these parameters by eavesdropping or predicting, forming a valid A_{1i} is not possible because it has two biometric credentials of the user that makes it impossible to predict in polynomial time. Hence, construction of any further parameters is just vague. Also, the gateway stores A_{2i} that is composed by hashing pre-secret of the gateway and the user, credentials of the gateway, and point multiplied hashed user credentials. Any changes in A_{2i} can easily be caught at the gateway, resulting in the session termination on the failed integrity check. Thus, the presence of a man amidst the communicating parties cannot go undetected. In this way, the proposed protocol provides strong protection against the man-in-the-middle attack.

g) *It is resistant against the replay attack.* For protecting our protocol from the replay attack [27], every message packet and the variables being communicated over a public channel are constricted by using the current timestamp in hashed format. So, when attacker $\mathbb{A}$ replays the message by altering the timestamp, it would result in the failed inequality checks for the freshness of the received message as per ΔT. Also, the integrity checks at the other entity would not pass due to the changed hash values as the messages include timestamp. Hence, our protocol resists the replay attack.

h) *It is resistant against the privileged-insider attack.* Assuming an intruder to be within the gateway, it would know the secret parameters of other users of the system. This information can be used by the insider for the purpose of fraudulence [28]. In order to mitigate this attack, the proposed protocol stores the secret credentials (A_{2i}, A_{5k}) at the gateway in the registration phase in irreversible hash format. To acquire any parameter from reverse engineering hash is unworkable. Moreover, even if the insider has access to A_{2i}, A_{5k}, it would not be able to generate the session key SK_{ik} as these parameters are not used to compute the session key. So, the proposed protocol ensures protection against the privileged insider attack.

i) *It is resistant against the offline user identity and password predicting attack.* Predicting the user's identity (UID_i) or password (PW_i) by eavesdropping the messages being communicated over a public channel or by pick pocketing the smart device of the user U_i is not feasible. Consider that attacker $\mathbb{A}$ pick pockets a smart device that stores A_{1i}, A_{2i} and applies the power analysis on it. After acquiring these parameters, $\mathbb{A}$ cannot get the user's identity and password as these parameters are in one-way hash format whose reverse functioning is not feasible. Also, these parameters contain two biometric credentials of the user whose simultaneous prediction or forgery is not possible. Again, consider that $\mathbb{A}$ tries to eavesdrop the public channel for the purpose of guessing the user's credentials. It is a failed attempt as nowhere in the authentication phase the proposed protocol uses any user credential in the raw format. Thus, our protocol is safe from the offline user identity and password predicting attack.

j) *It is resistant against the session key computation attack.* The session key SK_{ik} is computed afresh for each session by the establishing parties, i.e., user and sensor [29]. The session key is computed as $SK_{ik} = b_i.S\beta_k + D_k.U\alpha_i = d_k.U\beta_i + B_i.S\alpha_k$. The construction of a session key involves random numbers (b_i, d_k), public keys $(U\beta_i, S\beta_k)$, private keys $(U\alpha_i, S\alpha_k)$ and scalar point-multiplied variables (B_i, D_k) of both the entities. Even if attacker $\mathbb{A}$ has access to the public parameters B_i, D_k and $U\beta_i$, $S\beta_k$; $\mathbb{A}$ has no knowledge of b_i, d_k and $U\alpha_i$, $S\alpha_k$; and predicting them simultaneously is not possible. Suppose b_i, d_k or $U\beta_i$, $S\beta_k$ are leaked to $\mathbb{A}$, the other parameter holds the session key strong. Thus, our protocol is robust against the session-key computation attack.

k) *It is resistant against the known session-specific transient information attack.* Consider that by some means some of the transient parameters (b_i, D_k, d_i, B_i) is leaked to attacker $\mathbb{A}$. For constructing the current session-key SK_{ik}, $\mathbb{A}$ needs the private key of any one of the key establishing entities, i.e., user or sensor ($U\alpha_i$, $S\alpha_k$). Along with the private key, $\mathbb{A}$ would need two transient parameters simultaneously that would add up to the prediction of three parameters, making it unworkable for $\mathbb{A}$ to perform the attack [30]. Thus, our protocol is resistant against the known session-specific transient information attack.

l) *It is resistant against the stolen verifier attack.* For carrying out the stolen verifier attack, attacker $\mathbb{A}$ has to steal the table where the gateway stores the user related secret credentials like identity or passwords. By stealing the table, $\mathbb{A}$ can use this information for fraudulence or change the information stored in table to hamper the system. The proposed protocol provides security against the stolen verifier attack as the gateway never stores the communicating entities' secret credentials in the raw format in table.

m) *It is resistant against the de-synchronization attack.* For performing the de-synchronization attack, attacker $\mathbb{A}$ needs to inject the contradictory values of the variables in the memory or database of the communicating parties. In the proposed protocol, $\mathbb{A}$ cannot change the stored data as the storing occurs in the registration phase that is carried out over a private channel. So, $\mathbb{A}$ can change the data in the current memory of the communicating parties. During authentication, even if $\mathbb{A}$ changes any parameter, it would be caught during the integrity checks. Thus, our protocol is safe from the de-synchronization attack.

n) *It provides the user anonymity.* Suppose, attacker $\mathbb{A}$ eavesdrops the authentication request message. Still $\mathbb{A}$ cannot predict the user identity (UID_i), as already discussed in point 'i'. Moreover, the proposed protocol ensures the anonymity [8] as UID_i is not sent in the plain text format over a public channel. UID_i is communicated over a private channel in the user registration phase in the hashed format that ensures complete anonymity.

o) *It is resistant against the un-traceability attacks.* For tracing any user of the system, attacker $\mathbb{A}$ should be able to differentiate the source of the messages in different sessions. $\mathbb{A}$ cannot identify the source of a message in our protocol as the parameters being sent over a public channel are in the hash digest format that were constructed with the current timestamps and random numbers. Since these parameters change in each active session, the source of a message remains indistinguishable. Thus, the proposed protocol ensures the un-traceability.

p) *It provides ideal forward secrecy.* The forward secrecy assures the strength of a session-key despite the situation where any of the long-term keys K_{ij} or L_{kj} is exposed to attacker $\mathbb{A}$. Our protocol provides perfect forward secrecy as the session key SK_{ik} is constructed in such a way that these long term keys just aid in establishing the key, not computing it. So, despite the fact that the long term keys are leaked, $\mathbb{A}$ cannot compute the current session key $SK_{ik} = b_i.S\beta_k + D_k.U\alpha_i = d_k.U\beta_i + B_i.S\alpha_k$, as discussed in point 'j'. Hence, our protocol provides robust perfect forward secrecy.

q) *It provides mutual authentication.* In our protocol, the authentication phase is carried out over a public channel. First, the user U_i requests the gateway GWN_j to authenticate it via the message $Pckt_1 = \{I_{2ij}, B_i, T_{1i}\}$. After successful integrity check, GWN_j authenticates U_i. Then, GWN_j composes a fresh authentication request for sensor SN_k as $Pckt_2 = \{I_{4jk}, B_i, T_{2j}\}$. Similarly, SN_k authenticates GWN_j. Thereafter, SN_k composes a new authentication request message for GWN_j as $Pckt_3 = \{I_{6kj}, D_k, T_{3k}\}$. In the same way, GWN_j authenticates SN_k. Lastly, GWN_j sends $Pckt_4 = \{I_{8ji}, D_k, T_{4j}\}$ to U_i for authenticating it. Thus, all the communicating parties U_i, GWN_j, SN_k mutually authenticate each another via four communicated messages.

5.2 Burrows–Abadi–Needham (BAN) Logic

BAN logic verifies the cryptographic protocols by using propositional logic. It is used to find the flaws in the security protocols, and its many variants have been developed over the years [31]. The fundamental principle of the logic is the faith of an entity in the authenticity of the formula [32]. It results in illuminating derivations which disclose precise errors in the protocol [33]. The formal validation of our proposed protocol is shown by using the well-accepted BAN logic [34]. The proof shows that the proposed protocol preserves the privacy of the shared session key only among the user U_i and sensor node SN_k. For validating the proposed protocol, we first idealize the message sent on a public channel, which is just the elaborate form of how each component of the message is composed. Then, few assumptions regarding random numbers, timestamps, shared keys, and variables are considered. Following this, the fundamental rules of the BAN and the assumptions taken together result in some statements regarding the proposed protocol. These statements, when put together, help to fulfill the AIM considered to prove the robustness of the protocol [34]. Table 3 contains the symbols

Table 3 Symbols used for BAN logic proof

Symbols	Statements
$U\mid\equiv D$	U trusts D to be true
$\#(D)$	D is treated as new
$U\mid\sim D$	U formerly mentioned D
D, K_L	D/K is encoded using symmetric key L
$\langle D\rangle_K$	D comprises of K
$\langle D\rangle_{L\mapsto U}$	D is encoded using public key L of U
(D, K)	D/K is part of (D,K)
$U \triangleleft D$	U perceives D
$(D, K)_L$	D/K is hashed using key L
D/K	If D holds then K is followed
$U \xleftrightarrow{L} V$	U and V can safely communicate by using shared key L
$U \Rightarrow D$	U has rights on D

used for this logical proof, Table 4 contains the fundamental rules [32] of the proof, Table 5 contains the ideal message forms, and Table 6 contains the assumptions made for the proof.

Aims

Here we consider four AIMs for the BAN logic proof that are to be achieved for proving the security strength of our protocol.

Table 4 Ground rules for BAN logic proof

Rules	Explanations
Message Meaning Rule (MMR)	$\frac{S\mid\equiv S\xleftrightarrow{L}R,\, S\triangleleft\langle I\rangle_J}{S\mid\equiv R\mid\sim I}$
If S trusts that L is shared with R and perceives $\langle I\rangle J$,	
then S trusts, R formerly mentioned I	
Freshness Conjuncatenation Rule (FCR)	$\frac{S\mid\equiv\#(I)}{S\mid\equiv\#(I,J)}$
If S trusts, I is new,	
then S trusts newness of (I, J)	
Nonce Verification Rule (NVR)	$\frac{S\mid\equiv\#(I),\, S\mid\equiv R\mid\sim I}{S\mid\equiv R\mid\equiv I}$
If S trusts that I is new and R formerly mentioned I,	
then S trusts R which further trusts I	
Belief Rule (BR)	$\frac{S\mid\equiv(I),\, S\mid\equiv(J)}{S\mid\equiv(I,J)}$
If S trusts I and it also trusts J,	
then S trusts (I, J)	
Jurisdiction Rule (JR)	$\frac{S\mid\equiv R\Rightarrow I,\, S\mid\equiv R\mid\equiv I}{S\mid\equiv I}$
If S trusts, R has rights on I and R trusts I,	
then S trusts I	
Session Key Rule (SKR)	$\frac{S\mid\equiv\#(I),\, S\mid\equiv R\mid\equiv I}{S\mid\equiv S\xleftrightarrow{L}R}$
If S trusts, I to be new and R trusts I, an essential factor of the session key,	
then S trusts that it shares the session key L with R	

AIM 1: $U_i\mid\equiv U_i \xleftrightarrow{SK_{ik}} SN_k$
AIM 2: $U_i\mid\equiv SN_k\mid\equiv SN_k \xleftrightarrow{SK_{ik}} U_i$
AIM 3: $SN_k\mid\equiv SN_k \xleftrightarrow{SK_{ik}} U_i$
AIM 4: $SN_k\mid\equiv U_i\mid\equiv U_i \xleftrightarrow{SK_{ik}} SN_k$

Proof

Through *Message 1*, we perceive $Statement_1 : SN_k \triangleleft B_{i_{b_i}}$ As of A_{11}, $Statement_1$ and MMR we get $Statement_2 : SN_k\mid\equiv U_i\mid\sim B_i$ As of $Statement_2$, A_6 and NVR we get $Statement_3 : SN_k\mid\equiv U_i\mid\equiv B_i$ As of A_6, $Statement_3$ and SKR we get $Statement_4 : SN_k\mid\equiv SN_k \xleftrightarrow{SK_{ik}} U_i$ where, $SK_{ik} = d_k.U\beta_i + S\alpha_k.B_i$ ...**(AIM 3)** As of A_6, $Statement_4$ and NVR we get $Statement_5 : SN_k\mid\equiv U_i\mid\equiv U_i \xleftrightarrow{SK_{ik}} SN_k$ where, $SK_{ik} = d_k.U\beta_i + S\alpha_k.B_i$ ...**(AIM 4)** Through *Message 3*, we perceive $Statement_6 : U_i \triangleleft D_{k_{d_k}}$ As of A_{12}, $Statement_6$ and MMR we get $Statement_7 : U_i\mid\equiv SN_k\mid\sim D_k$ As of A_4, $Statement_7$ and NVR we get $Statement_8 : U_i\mid\equiv SN_k\mid\equiv D_k$ As of A_4, $Statement_8$ and SKR we get $Statement_9 : U_i\mid\equiv U_i \xleftrightarrow{SK_{ik}} SN_k$ where, $SK = b_i.S\beta_k + U\alpha_i.D_k$ ...**(AIM 1)** As of A_4, $Statement_9$ and NVR we get $Statement_{10} : U_i\mid\equiv SN_k\mid\equiv SN_k \xleftrightarrow{SK_{ik}} U_i$ where, $SK = b_i.S\beta_k + U\alpha_i.D_k$ ...**(AIM 2)**

5.3 Real-Or-Random (ROR) Model

Here, the semantic security strength of the proposed protocol is discussed by verifying it via the ROR model [35]. Recently, the ROR model has become popular due to its wide use in modern cryptography to analyze various mutual authentication protocols to prove the secrecy of the session key. Abdalla et al. [36] developed it as a probabilistic polynomial-time Turing machine based on the well-accepted DY threat model. It is used to ensure the conservation of the session key from various attacks. As per the ROR model, the attacker needs to differentiate between the genuine keys or random numbers to be successfully able to attack the system. The security model of the proposed protocol is designed using a game amid the challenger $\mathbb{C}$ and attacker $\mathbb{A}$. Different kinds of active or passive attacks are modeled in each game, and we evaluate the attacker's gain in breaking the system security. We provide some definitions which are essential for the analysis of the proposed protocol, as follows.

a) **Irreversible collision-free cryptographic hash function h(.)** : The function h(.) is formally defined as a deterministic function that transforms the strings of variable length to a string of fixed length x bits, i.e., $\{0, 1\}^* \longrightarrow \{0, 1\}^x$. The function h(.) is accessible to attacker $\mathbb{A}$ and the authorized entities of the proposed protocol OP. Assume that $Advt_{\mathbb{A}}^{h(.)}$ is an attacker's gain

Table 5 Idealized form of messages

$Message\,1\ U_i \overset{viaGW_j}{\longrightarrow} SN_k : T_{1i}, B_i : \langle B_i \rangle_{b_i.P}, I_{2ij} : \langle I_{1ij} \rangle_{(A_{2i} \Vert K_{ij} \Vert U\beta_i)}, T_{1i}, \langle B_i \rangle_{b_i.P}$
$Message\,2\ GW_j \longrightarrow SN_k : T_{2j}, B_i : \langle B_i \rangle_{b_i.P}, I_{4jk} : \langle I_{3jk} \rangle_{(A_{5k} \Vert L_{kj} \Vert G\beta_j)}, T_{2j}, \langle B_i \rangle_{b_i.P}$
$Message\,3\ SN_k \overset{viaGW_j}{\longrightarrow} U_i : T_{3k}, D_k : \langle D_k \rangle_{d_k.P}, I_{6kj} : \langle I_{5kj} \rangle_{(GID_j \Vert L_{kj} \Vert S\beta_k \Vert I_{3jk})}, T_{3k}, \langle B_i \rangle_{b_i.P}, \langle D_k \rangle_{d_k.P}$
$Message\,4\ GW_j \longrightarrow U_i : T_{4j}, D_k : \langle D_k \rangle_{d_k.P}, I_{8ji} : \langle I_{7ji} \rangle_{(A_{2i} \Vert K_{ij} \Vert G\beta_j)}, T_{4j}, \langle D_k \rangle_{d_k.P}$

Table 6 Assumptions

$A_1 : U_i \vert \equiv \#\{T_{1i}\}$	$A_7 : U_i \vert \equiv U_i \overset{K_{ij}}{\longleftrightarrow} GW_j$	$A_{10} : SN_k \vert \equiv SN_k \overset{L_{kj}}{\longleftrightarrow} GW_j$
$A_2 : GW_j \vert \equiv \#\{T_{2j}, T_{4j}\}$	$A_8 : GW_j \vert \equiv GW_j \overset{K_{ij}}{\longleftrightarrow} U_i$	$A_{11} : SN_k \vert \equiv SN_k \overset{B_i, D_i}{\longleftrightarrow} U_i$
$A_3 : SN_k \vert \equiv \#\{T_{3k}\}$	$A_9 : GW_j \vert \equiv GW_j \overset{L_{kj}}{\longleftrightarrow} SN_k$	$A_{12} : U_i \vert \equiv U_i \overset{B_i, D_i}{\longleftrightarrow} SN_k$
$A_4 : U_i \vert \equiv \#\{b_i, B_i, c_j, C_j, d_k, D_k, e_j.E_j\}$		
$A_5 : GW_j \vert \equiv \#\{b_i, B_i, c_j, C_j, d_k, D_k, e_j.E_j\}$		
$A_6 : SN_k \vert \equiv \#\{b_i, B_i, c_j, C_j, d_k, D_k, e_j.E_j\}$		

in concluding that the hash function results in collision within the time T_{MAX}, where $Advt_{\mathbb{A}}^{h(.)}(T_{MAX}) = Prob[(st_1, st_2) \longleftarrow_r \mathbb{A} : st_1 \neq st_2, h(st_1) = h(st_2)])$ and $(st_1, st_2) \longleftarrow_r \mathbb{A}$ denotes that the string pair st_1, st_2 is arbitrarily picked by attacker. The pair (δ, T_{MAX}) signifies that the resistance of the hash function being collision free is $Advt_{\mathbb{A}}^{h(.)}(T_{MAX}) \leq \delta$, where maximum execution time of the query for finding the collision is T_{MAX}.

b) **ECDLP :** ECDLP states that it is computationally hard to find $a \in \mathbb{Z}_q^*$ from $A = a.P$, where A and P are two elliptic curve points, i.e., $A, Q \in E/F_q$.

c) **ECDHP :** ECDHP states that it is infeasible to calculate the values of $a, b \in \mathbb{Z}_q^*$ from $a.b.P$, where $a.P, b.P, P \in E/F_q$.

In our protocol, the entities involved in the security model are remote user U_i, gateway GWN_j, and sensor nodes SN_k. The components of the ROR model are given below.

a) **Participants :** Let I_1, I_2 and I_3 be the occurrences of user U_i, gateway GWN_j and sensor node SN_k, respectively, which are denoted as $\prod_{U_i}^{I_1}, \prod_{GWN_j}^{I_2}$ *and* $\prod_{SN_k}^{I_3}$. Also, they are alternatively termed as Oracles.

b) **Accepted States :** Any instance $\prod^I$ is considered to be an accepted state if, upon reception of a former message, the instance enters into the acceptance state. If all the messages exchanged during the communication are arranged as per the ongoing session, then they are called session identification (SId).

c) **Partnering :** Consider two instances $\prod^{I_1}$ and $\prod^{I_2}$, when both of these instances are in the acceptance state. If they share similar SId while mutually authenticating each other and they are mutual partners of each other, then both the instances $\prod^{I_1}$ *and* $\prod^{I_2}$ are said to be the partners.

d) **Attacker :** As ROR model is based on the DY model, attacker $\mathbb{A}$ can overhear, alter, insert and delete any message communicated over a public channel among the communicating entities of the system. Apart from these, the queries mentioned below are also accessible to attacker $\mathbb{A}$.

i. **Execute :** This query allows attacker $\mathbb{A}$ to read the information exchanged amidst the legitimate entities of the system. It may also be said that this query models an eavesdropping attack.

ii. **Send :** This query allows attacker $\mathbb{A}$ to inject a message into a message of the system, which is sent to any participating instance and in turn receives a reply for the sent message. An active attack is performed using the send query.

iii. **Reveal :** When a public message contains the current session key SK_{ik} shared amid the user U_i and sensor node SN_k, which is computed by the partner instances, then the reveal query helps attacker $\mathbb{A}$ in acquiring the current session key SK_{ik}.

iv. **CorruptSmartDevice :** The CorruptSmartDevice query allows attacker $\mathbb{A}$ to obtain all the stored secret credentials from a smart device obtained by attacker $\mathbb{A}$ by some means. This query assures that the contents of the device are not corrupted and extracted safely by the power analysis attack. This is possible as the ROR model is based on the DY model, which ensures weak-corruption model. It is assumed that attacker $\mathbb{A}$ can execute only few of these queries.

v. **Test :** When attacker $\mathbb{A}$ executes the test query, prior to that a non-biased coin $\mathbb{UC}$ is tossed whose results are held secret. It is considered that the test query returns a fresh session key SK_{ik} for the output of $\mathbb{UC}$ as 1 (head), and a random value for output 0(tail), and for all other cases, the output is treated as null. Hence, it is concluded that by executing the test query, the security of a session-key SK_{ik} shared amidst the user U_i and sensor node SN_k is put to test. It is assumed that attacker $\mathbb{A}$ can execute many test queries.

e) **Freshness :** Consider two instances $\prod_A^{I_1}$ and $\prod_B^{I_2}$. They are considered to be fresh when attacker $\mathbb{A}$ is unable to acquire the session key SK_{ik} even after using the Reveal Query.

f) **Random Oracle :** Our proposed protocol uses the collision free irreversible cryptographic one-way hash function h(.). The hash function h(.) here is modeled as random oracle $\mathbb{HO}$, which is accessible not only to the communicating parties of the system but also to attacker $\mathbb{A}$. When attacker $\mathbb{A}$ executes a hash query over any message M, the challenger $\mathbb{C}$ replies with an arbitrary value R. Then, it marks the pair (M, R) for entry into its database.

The ROR model considers TA's to be fully trusted entity and the gateway to be semi-trusted entity in the system.

5.3.1 Security Proof

Here, we validate the semantic security strength of the session-key SK_{ik} established amid the remote-user U_i and sensor node SN_k in all the phases of the proposed protocol.

Theorem *Let $Advt_{\mathbb{A}}^{OP}(T_{MAX})$ be the gain of attacker $\mathbb{A}$ over our protocol OP for breaking down the security of session-key SK_{ik} in at most running time T_{MAX}. The advantage of attacker $\mathbb{A}$ is expressed as follows.*

$$Advt_{\mathbb{A}}^{OP}(T_{MAX}) \le \frac{Q_{hash}^2}{|Hash|} + \frac{Q_S}{2^{b+h-1}|D|} + 2Advt_{\mathbb{A}}^{ECDHP}(T_{MAX}). \quad (1)$$

Here, $Q_{hash}, |Hash|, Q_S, b, h, |D|$ and $Advt_{\mathbb{A}}^{ECDHP}(T_{MAX})$ represent total number of hashed queries, range of cryptographic hash function h(.), total number of the send queries, total number of bits in biometric credential of user, number of bits in heartbeat bitstream of user, size of the dictionary holding passwords, and the gain of attacker $\mathbb{A}$ in breaking the ECDHP in at most T_{MAX} time, respectively.

Proof For proving the above theorem, we consider five games, namely, $Game_0, Game_1, Game_2, Game_3$ and $Game_4$. Let $Adv_{\mathbb{A}}^{Game_i}$ denote the gain of attacker $\mathbb{A}$ in winning the game $Game_i$, where $Adv_{\mathbb{A}}^{Game_i} = prob[Success_i]$. Here, attacker $\mathbb{A}$ predicts the bit c of the event $Success_i$ in game $Game_i$. Assuming that $Advt_{\mathbb{A}}^{OP}(T_{MAX})$ is the gain of attacker $\mathbb{A}$ in breaking down the security of session-key SK_{ik} in maximum execution time T_{MAX}. We elaborate the details of each game as follows.

a) **Game$_0$** : In $Game_0$, attacker $\mathbb{A}$ predicts the bit c and carries out a real attack. In this case, the gain of attacker $\mathbb{A}$ turns out to be as follows [37].

$$Advt_{\mathbb{A}}^{OP}(T_{MAX}) = |2Advt_{\mathbb{A}}^{Game_0} - 1|. \quad (2)$$

b) **Game$_1$** : In $Game_1$, attacker $\mathbb{A}$ executes the Execute and Test queries to model the eavesdropping attack on the system. This game checks whether the session key SK_{ik} established amidst the user U_i and sensor node SN_k is an arbitrary number or it is an actual key. In our protocol, the session-key is computed by the user U_i as $SK_{ik} = b_i.S\beta_k + D_k.U\alpha_i$ and that by the sensor node SN_k as $SK_{ik} = d_k.U\beta_i + B_i.S\alpha_k$. Along with the session key SK_{ik}, the messages exchanged over a public channel among the communication parties of the system are $Pckt_1 = I_{2ij}, B_i, T_{1i}$, $Pckt_2 = I_{4jk}, B_i, T_{2j}$, $Pckt_3 = I_{6kj}, D_k, T_{3k}$ and $Pckt_4 = I_{8ji}, D_k, T_{4j}$, that are likely to be intercepted by attacker $\mathbb{A}$. From the exchanged messages, it can be seen that the messages are no aid in constructing the session key SK_{ik}. Hence, it can be concluded that via the eavesdropping attempt to attack the system, attacker $\mathbb{A}$ cannot construct the session key SK_{ik}, and as a result $\mathbb{A}$ has no increasing chance of winning the game. So, the gain of attacker $\mathbb{A}$ remains the same [37] as it was earlier, i.e.,

$$Advt_{\mathbb{A}}^{Game_1} = Advt_{\mathbb{A}}^{Game_0}. \quad (3)$$

c) **Game$_2$** : In $Game_2$, attacker $\mathbb{A}$ tries to model the masquerading attack by forging any of the above mentioned message packets $Pckt_1$, $Pckt_2$, $Pckt_3$ and $Pckt_4$. This attack can be formulated by attacker $\mathbb{A}$ by using the send query on $\mathbb{HO}$. When attacker $\mathbb{A}$ attempts to morph the message packets, $\mathbb{A}$ needs the secret credentials $A_{1i}, A_{2i}, K_{ij}, A_{5k}$ and L_{kj}, which are secured via the collision free hash function. Moreover, the message packets contain the fresh random nonce and timestamps that make them infeasible to break. Hence, in this game, the gain of attacker $\mathbb{A}$ as per implying the birthday paradox turns out to be as follows [35].

$$|Advt_{\mathbb{A}}^{Game_1} - Advt_{\mathbb{A}}^{Game_2}| \le \frac{Q_{hash}^2}{2.|Hash|}. \quad (4)$$

d) **Game$_3$** : In this game, attacker $\mathbb{A}$ tries to launch the stolen smart card attack via implementing the CorruptSmartDevice query. In this attack, attacker $\mathbb{A}$ tries to extract the credentials $\{A_{1i}, A_{2i}\}$ stored in the smart device. From these extracted credentials, $\mathbb{A}$ tries to guess the password

PW_i of user U_i via the dictionary attack. Moreover, the biometric credentials BIO_i and HB_i can be predicted with probability $\frac{1}{2^{b+h}}$. Hence, the session key SK_{ik} cannot be computed as all the credentials are in collision free cryptographic hash format and the secret credentials PW_i, BIO_i, HB_i cannot be extracted. The gain of the game results as follows [38].

$$|Advt_{\mathbb{A}}^{Game_2} - Advt_{\mathbb{A}}^{Game_3}| \leq \frac{Q_S}{2^{b+h}|D|}. \tag{5}$$

e) **Game**$_4$: In this game, attacker $\mathbb{A}$ attempts to compute the session key $SK_{ik} = b_i.S\beta_k + D_k.U\alpha_i = d_k.U\beta_i + B_i.S\alpha_k$ from the data packets transmitted over a public channel like computation of b_i, d_k from B_i, D_k. These computations are only possible if ECDHP can be solved. So, this would leave the gain of attacker $\mathbb{A}$ as follows [39].

$$|Advt_{\mathbb{A}}^{Game_3} - Advt_{\mathbb{A}}^{Game_4}| \leq Advt_{\mathbb{A}}^{ECDHP}(T_{MAX}). \tag{6}$$

At last, the attacker's gain in guessing the bit c is given as follows [40].

$$Advt_{\mathbb{A}}^{Game_4} = \frac{1}{2}. \tag{7}$$

From eqns. (2), (3) and (7), the following results are obtained:

$$\begin{aligned} \frac{1}{2}Advt^{\mathbb{O}}\mathbb{P}_{\mathbb{A}}(T_{MAX}) &= |Advt_{\mathbb{A}}^{Game_0} - \frac{1}{2}|. \\ &= |Advt_{\mathbb{A}}^{Game_1} - \frac{1}{2}|. \\ &= |Advt_{\mathbb{A}}^{Game_1} - Advt_{\mathbb{A}}^{Game_4}| \end{aligned} \tag{8}$$

Using the triangular inequality and eqns. (4), (5) and (6), the following results are obtained:

$$\begin{aligned} &|Advt_{\mathbb{A}}^{Game_1} - Advt_{\mathbb{A}}^{Game_4}| \\ &\quad = |Advt_{\mathbb{A}}^{Game_1} - Advt_{\mathbb{A}}^{Game_2}| \\ &\qquad + |Advt_{\mathbb{A}}^{Game_2} - Advt_{\mathbb{A}}^{Game_4}| \\ &\quad = |Advt_{\mathbb{A}}^{Game_1} - Advt_{\mathbb{A}}^{Game_2}| \\ &\qquad + |Advt_{\mathbb{A}}^{Game_2} - Advt_{\mathbb{A}}^{Game_3}| \\ &\qquad + |Advt_{\mathbb{A}}^{Game_3} - Advt_{\mathbb{A}}^{Game_4}| \\ &\quad \leq \frac{Q_{hash}^2}{2|Hash|} + \frac{Q_S}{2^{b+h}|D|} \\ &\qquad + Advt_{\mathbb{A}}^{ECDHP}(T_{MAX}). \end{aligned} \tag{9}$$

By solving Eqs. (8) and (9), we get the following result:

$$\begin{aligned} Advt_{\mathbb{A}}^{OP}(T_{MAX}) \;\leq\; & \frac{Q_{hash}^2}{|Hash|} + \frac{Q_S}{2^{b+h-1}|D|} \\ & + 2Advt_{\mathbb{A}}^{ECDHP}(T_{MAX}). \end{aligned} \tag{10}$$

Thus, the theorem is proved. ■

5.4 Simulations

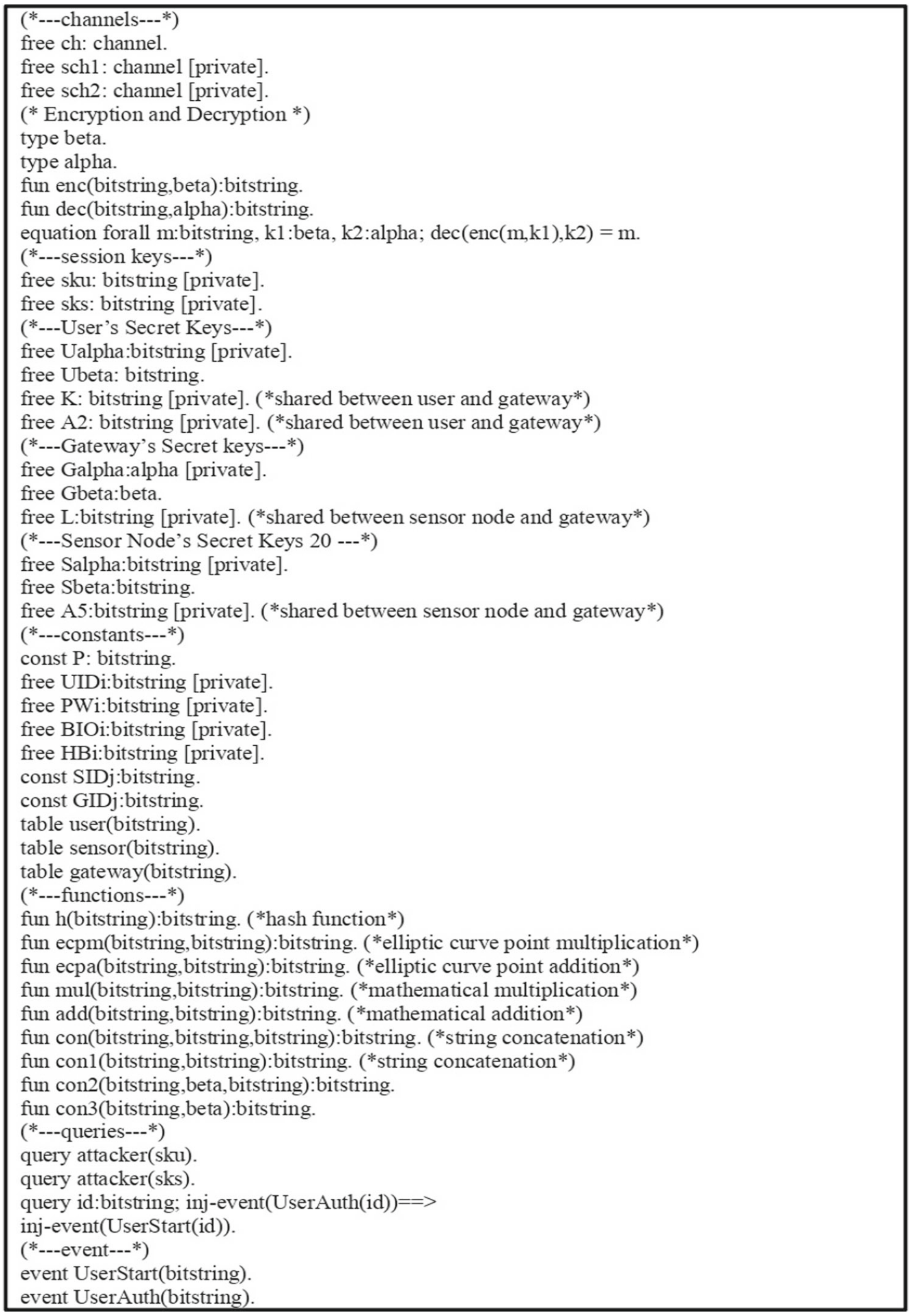
Here, we discuss the simulation results using ProVerif to show the robustness against the cryptographic attacks and privacy of all the secret credentials in our protocol.

5.4.1 ProVerif

The ProVerif is a formal verification tool for simulating an authentication protocol [41]. It uses the unbounded sessions and message space to verify the reachability of code, observational equivalence, and correspondence assertions. It uses a subset of the *pi* calculus and horn clause to model the protocol in the form of queries. It works on the well-known DY intruder model and allows a user to define various cryptographic operations like XOR, hash, encryption/decryption, etc. The attacker is modeled as a parallel process that could perform the read, write, and execute operations over a public channel [41]. The code for the protocol is shown in Figs. 8, 9, 10, 11, 12, 13, which is discussed below. The details about the channels used, session keys, communicating entities secrets, constants, functions for point multiplication, addition, concatenation, etc., and the queries for modeling an attacker and assuring the secrecy of session-key from all active/passive attacks as per the proposed protocol's functionality are all defined in Fig. 8. The correspondence relations among the communicating entities are checked via the events. The process for the communicating entities is denoted as: let User=, let Sensor= and let Gateway= which is highlighted in Figs. 9, 10, 11. In these processes, the step wise illustrations of the operations undertaken are coded where **new** gives the fresh variables, **out**() and **in**() mark the variables that navigate over the communication channels. The **insert** keyword tabulates the entry of the variables in the communicating parties' database. The process keyword models all the communicating entities working in parallel. Figure 12 highlights the authentication process among all the communicating entities and gives the code for checking the presence of any adversary. The statement **'Query not attacker is true'** in the result of the ProVerif code depicted in Fig. 13 explains that the proposed protocol is free from all the active and passive attacks.

6 Performance Analysis

Here, we discuss the performance of our protocol and compare it with that of the protocols [4,6,9,14] in terms of energy consumption for each transmitted bit, communication, storage cost, computation, and the security functionalities. We



```
(*---channels---*)
free ch: channel.
free sch1: channel [private].
free sch2: channel [private].
(* Encryption and Decryption *)
type beta.
type alpha.
fun enc(bitstring,beta):bitstring.
fun dec(bitstring,alpha):bitstring.
equation forall m:bitstring, k1:beta, k2:alpha; dec(enc(m,k1),k2) = m.
(*---session keys---*)
free sku: bitstring [private].
free sks: bitstring [private].
(*---User's Secret Keys---*)
free Ualpha:bitstring [private].
free Ubeta: bitstring.
free K: bitstring [private]. (*shared between user and gateway*)
free A2: bitstring [private]. (*shared between user and gateway*)
(*---Gateway's Secret keys---*)
free Galpha:alpha [private].
free Gbeta:beta.
free L:bitstring [private]. (*shared between sensor node and gateway*)
(*---Sensor Node's Secret Keys 20 ---*)
free Salpha:bitstring [private].
free Sbeta:bitstring.
free A5:bitstring [private]. (*shared between sensor node and gateway*)
(*---constants---*)
const P: bitstring.
free UIDi:bitstring [private].
free PWi:bitstring [private].
free BIOi:bitstring [private].
free HBi:bitstring [private].
const SIDj:bitstring.
const GIDj:bitstring.
table user(bitstring).
table sensor(bitstring).
table gateway(bitstring).
(*---functions---*)
fun h(bitstring):bitstring. (*hash function*)
fun ecpm(bitstring,bitstring):bitstring. (*elliptic curve point multiplication*)
fun ecpa(bitstring,bitstring):bitstring. (*elliptic curve point addition*)
fun mul(bitstring,bitstring):bitstring. (*mathematical multiplication*)
fun add(bitstring,bitstring):bitstring. (*mathematical addition*)
fun con(bitstring,bitstring,bitstring):bitstring. (*string concatenation*)
fun con1(bitstring,bitstring):bitstring. (*string concatenation*)
fun con2(bitstring,beta,bitstring):bitstring.
fun con3(bitstring,beta):bitstring.
(*---queries---*)
query attacker(sku).
query attacker(sks).
query id:bitstring; inj-event(UserAuth(id))==>
inj-event(UserStart(id)).
(*---event---*)
event UserStart(bitstring).
event UserAuth(bitstring).
```

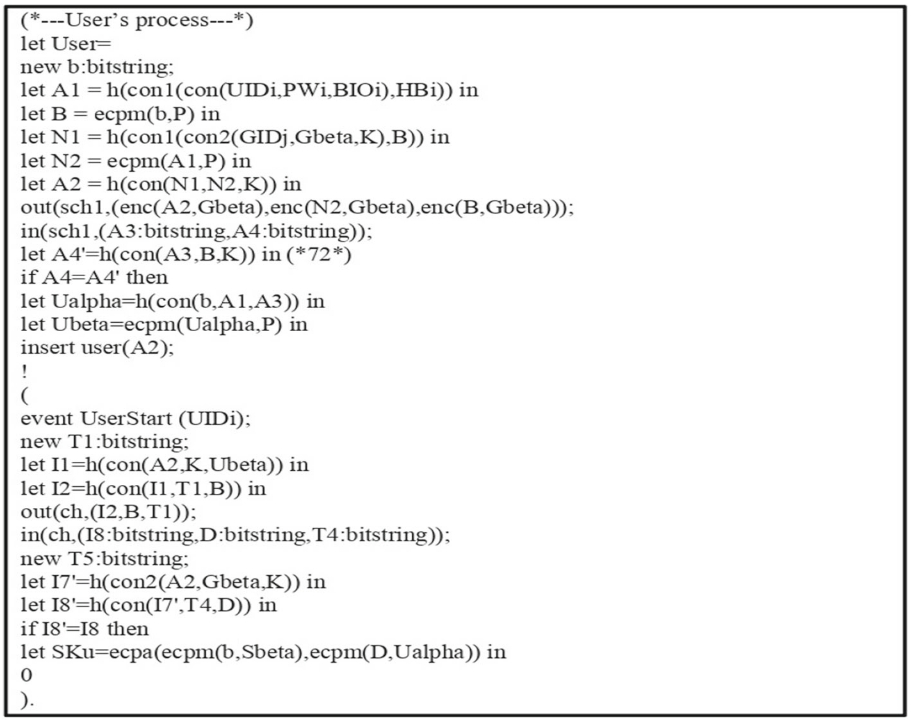
Fig. 8 Preliminaries

consider repetitively used user login and mutual authentication phases for comparison purpose.

6.1 Computation Overhead

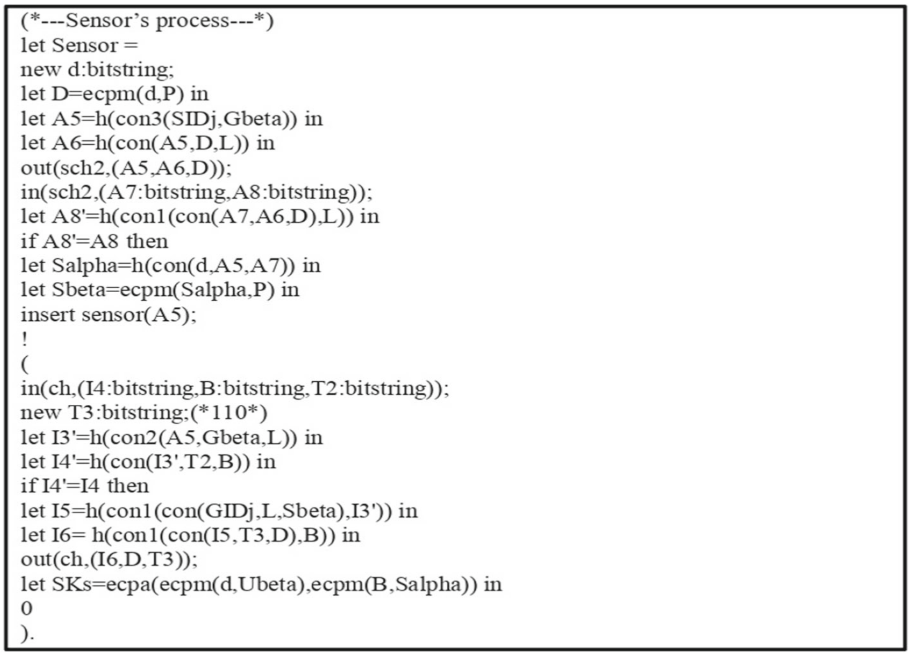
The notations used to denote the cryptographic operations and their consumption times [42] are given in Table 7. The computational complexity of our protocol along with that of the protocols [4,6,9,14] is shown in Table 8. From this table, we can see that our protocol has lower computational complexity in comparison with the protocols [6,9,14]. The protocol [4] has lower complexity, but it suffers from various security breaches. Hence, our protocol has lower computational complexity while ensuring robust security, making it useful in practical scenarios where the lack of security could create havoc.

Fig. 9 Process user



```
(*---User's process---*)
let User=
new b:bitstring;
let A1 = h(con1(con(UIDi,PWi,BIOi),HBi)) in
let B = ecpm(b,P) in
let N1 = h(con1(con2(GIDj,Gbeta,K),B)) in
let N2 = ecpm(A1,P) in
let A2 = h(con(N1,N2,K)) in
out(sch1,(enc(A2,Gbeta),enc(N2,Gbeta),enc(B,Gbeta)));
in(sch1,(A3:bitstring,A4:bitstring));
let A4'=h(con(A3,B,K)) in (*72*)
if A4=A4' then
let Ualpha=h(con(b,A1,A3)) in
let Ubeta=ecpm(Ualpha,P) in
insert user(A2);
!
(
event UserStart (UIDi);
new T1:bitstring;
let I1=h(con(A2,K,Ubeta)) in
let I2=h(con(I1,T1,B)) in
out(ch,(I2,B,T1));
in(ch,(I8:bitstring,D:bitstring,T4:bitstring));
new T5:bitstring;
let I7'=h(con2(A2,Gbeta,K)) in
let I8'=h(con(I7',T4,D)) in
if I8'=I8 then
let SKu=ecpa(ecpm(b,Sbeta),ecpm(D,Ualpha)) in
0
).
```

Fig. 10 Process sensor node



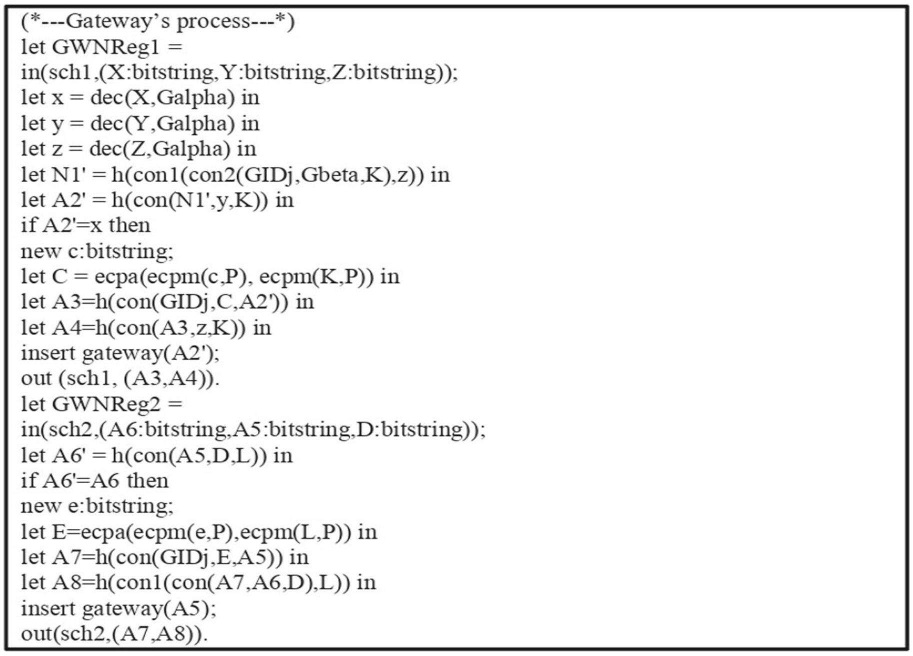
```
(*---Sensor's process---*)
let Sensor =
new d:bitstring;
let D=ecpm(d,P) in
let A5=h(con3(SIDj,Gbeta)) in
let A6=h(con(A5,D,L)) in
out(sch2,(A5,A6,D));
in(sch2,(A7:bitstring,A8:bitstring));
let A8'=h(con1(con(A7,A6,D),L)) in
if A8'=A8 then
let Salpha=h(con(d,A5,A7)) in
let Sbeta=ecpm(Salpha,P) in
insert sensor(A5);
!
(
in(ch,(I4:bitstring,B:bitstring,T2:bitstring));
new T3:bitstring;(*110*)
let I3'=h(con2(A5,Gbeta,L)) in
let I4'=h(con(I3',T2,B)) in
if I4'=I4 then
let I5=h(con1(con(GIDj,L,Sbeta),I3')) in
let I6= h(con1(con(I5,T3,D),B)) in
out(ch,(I6,D,T3));
let SKs=ecpa(ecpm(d,Ubeta),ecpm(B,Salpha)) in
0
).
```

6.2 Communication Cost

The communication cost of the proposed protocol along with that of the protocols [4,6,9,14] is shown in Table 9. Here, we have considered only the transmission overheads of each entity. The identities, timestamps, and random numbers have been considered 32 bits long, SHA-1 hashed messages 160 bits long, and symmetrically encrypted messages 512 bits long. From Table 9, we can see that the proposed protocol has minimum total communication overhead in contrast to the protocols [4,6,9,14], making it useful in the bandwidth constraint environment; also, the lower communication overhead makes the transmission process more proficient.

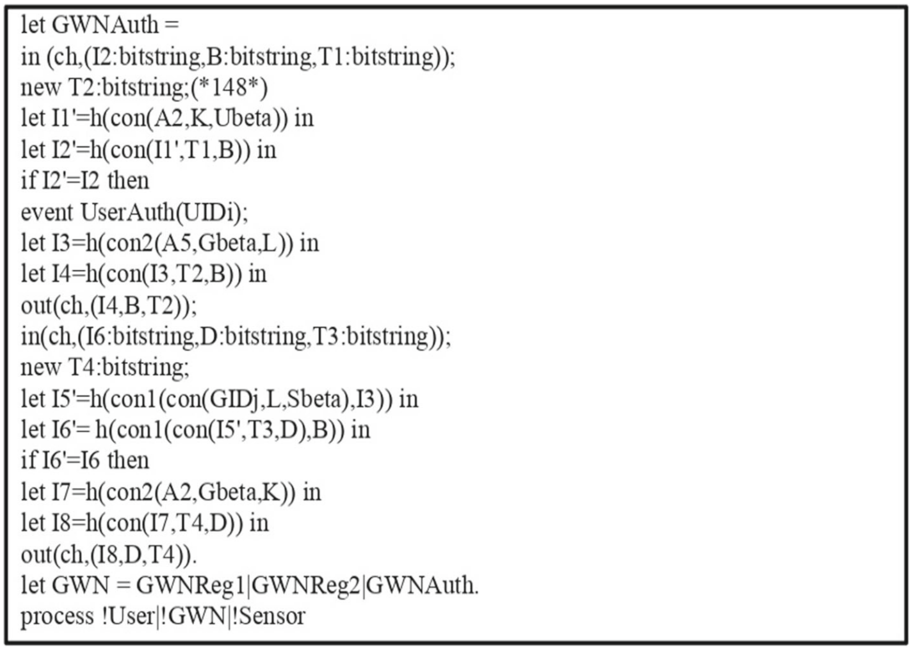
6.3 Smart Device Storage Overhead

Table 9 depicts the overhead for the smart device or card of the proposed protocol along with that of the protocols [4,6,9,14].



```
(*---Gateway's process---*)
let GWNReg1 =
in(sch1,(X:bitstring,Y:bitstring,Z:bitstring));
let x = dec(X,Galpha) in
let y = dec(Y,Galpha) in
let z = dec(Z,Galpha) in
let N1' = h(con1(con2(GIDj,Gbeta,K),z)) in
let A2' = h(con(N1',y,K)) in
if A2'=x then
new c:bitstring;
let C = ecpa(ecpm(c,P), ecpm(K,P)) in
let A3=h(con(GIDj,C,A2')) in
let A4=h(con(A3,z,K)) in
insert gateway(A2');
out (sch1, (A3,A4)).
let GWNReg2 =
in(sch2,(A6:bitstring,A5:bitstring,D:bitstring));
let A6' = h(con(A5,D,L)) in
if A6'=A6 then
new e:bitstring;
let E=ecpa(ecpm(e,P),ecpm(L,P)) in
let A7=h(con(GIDj,E,A5)) in
let A8=h(con1(con(A7,A6,D),L)) in
insert gateway(A5);
out(sch2,(A7,A8)).
```

Fig. 11 Process gateway



```
let GWNAuth =
in (ch,(I2:bitstring,B:bitstring,T1:bitstring));
new T2:bitstring;(*148*)
let I1'=h(con(A2,K,Ubeta)) in
let I2'=h(con(I1',T1,B)) in
if I2'=I2 then
event UserAuth(UIDi);
let I3=h(con2(A5,Gbeta,L)) in
let I4=h(con(I3,T2,B)) in
out(ch,(I4,B,T2));
in(ch,(I6:bitstring,D:bitstring,T3:bitstring));
new T4:bitstring;
let I5'=h(con1(con(GIDj,L,Sbeta),I3)) in
let I6'= h(con1(con(I5',T3,D),B)) in
if I6'=I6 then
let I7=h(con2(A2,Gbeta,K)) in
let I8=h(con(I7,T4,D)) in
out(ch,(I8,D,T4)).
let GWN = GWNReg1|GWNReg2|GWNAuth.
process !User|!GWN|!Sensor
```

Fig. 12 Process authentication

From this table, it can be noticed that the proposed protocol incurs minimum storage overhead, making it memory efficient and ensuring the smooth functioning of the hardware devices. The negligible storage overhead aids in providing longer lifetime of the device.

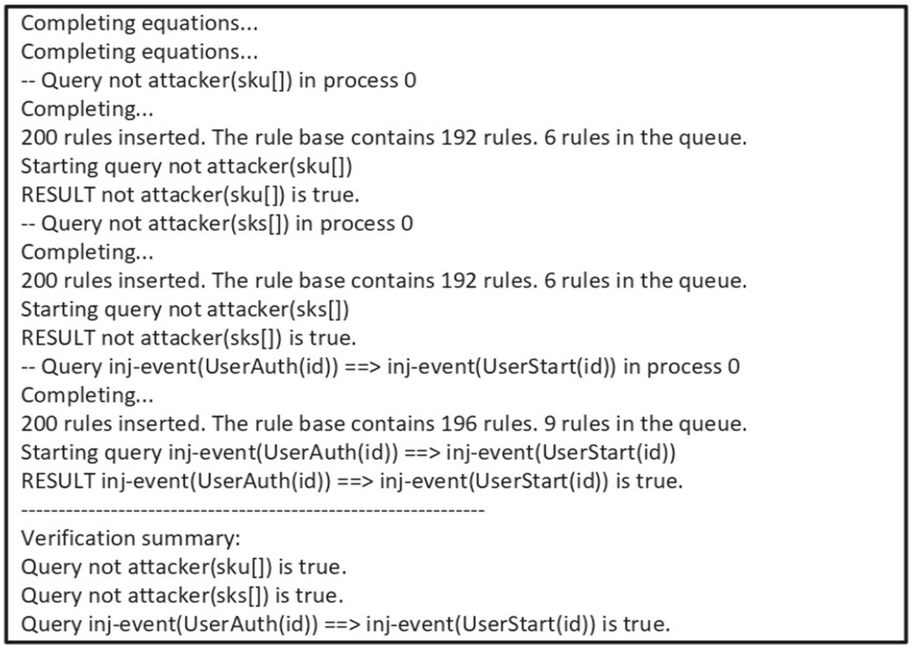
6.4 Energy Requirements

Here, we consider the energy consumption model as per [43] for calculating the energy consumed in execution of the proposed protocol for transmitting the data packets. The energy consumption for transmitting one bit is 4.602mJ [44]. Table 9 shows the energy utilization of our protocol along with that of the protocols [4,6,9,14]. It can be seen from this table that the proposed protocol has minimum energy consumption, making it lightweight for the devices and also environment friendly.

6.5 Security Functionalities

Table 10 provides the security functionalities of the proposed protocol along with that of the protocols [4,6,9,14]. This

Fig. 13 ProVerif result



```
Completing equations...
Completing equations...
-- Query not attacker(sku[]) in process 0
Completing...
200 rules inserted. The rule base contains 192 rules. 6 rules in the queue.
Starting query not attacker(sku[])
RESULT not attacker(sku[]) is true.
-- Query not attacker(sks[]) in process 0
Completing...
200 rules inserted. The rule base contains 192 rules. 6 rules in the queue.
Starting query not attacker(sks[])
RESULT not attacker(sks[]) is true.
-- Query inj-event(UserAuth(id)) ==> inj-event(UserStart(id)) in process 0
Completing...
200 rules inserted. The rule base contains 196 rules. 9 rules in the queue.
Starting query inj-event(UserAuth(id)) ==> inj-event(UserStart(id))
RESULT inj-event(UserAuth(id)) ==> inj-event(UserStart(id)) is true.
------------------------------------------------------------
Verification summary:
Query not attacker(sku[]) is true.
Query not attacker(sks[]) is true.
Query inj-event(UserAuth(id)) ==> inj-event(UserStart(id)) is true.
```

Table 7 Parameters for performance evaluation

Cryptographic operation	Notation	Computational cost (mS)
SHA-1 Hash Function	T_H	0.004
Biometric Key Generation	T_B	2.226
Elliptic Curve Point Multiplication	T_e	2.226
AES Encryption	T_E	3.85
Fuzzy Extractor	T_F	2.226
Elliptic Curve Point Addition	T_A	0.004
ADC Function	T_a	0.012

Table 8 Computation cost

Schemes	User	Gateway	Sensor	Total computational cost (mS)
Ref. [4]	$13T_H + T_B$	$15T_H$	$4T_H$	$32T_H + T_B = 2.354$ ms
Ref. [6]	$6T_H + T_E + T_F$	$3T_H + 2T_E$	$2T_H + T_E$	$11T_H + 4T_E + T_F = 17.67$ ms
Ref. [9]	$12T_H + 3T_e$	$5T_H + T_e$	$T_H + 2T_e$	$24T_H + 6T_e = 13.452$ ms
Ref. [14]	$4T_H + 3T_E$	$2T_H + 5T_E$	$T_H + 2T_E$	$7T_H + 10T_E = 38.528$ ms
Ours	$5T_H + 2T_e + T_A + T_a$	$8T_H$	$4T_H + 2T_e + T_A$	$17T_H + 4T_e + 2T_A + T_a = 8.992$ ms

Table 9 Overheads

Schemes	User (bits)	Gateway (bits)	Sensor (bits)	$U_i + GWN_j + SN_k$ (bits)	Storage (bits)	Energy (mJ)
Ref. [4]	864	896	352	2112	960	9719.424
Ref. [6]	576	576	192	1344	1440	6185.088
Ref. [9]	448	1088	352	1888	640	8688.576
Ref. [14]	544	544	544	1632	864	7510.464
Ours	224	448	224	896	320	4123.392

Table 10 Security functionalities

Characteristics	Ref. [4]	Ref. [6]	Ref. [9]	Ref. [14]	Ours
User anonymity	×	×	×	×	✓
Sensor anonymity	✓	✓	×	✓	✓
Untraceability	✓	×	×	×	✓
Counteraction against user impersonation	✓	×	✓	✓	✓
Counteraction against sensor impersonation	×	✓	✓	✓	✓
Counteraction against offline password guessing	✓	×	✓	✓	✓
Perfect mutual authentication	✓	✓	✓	✓	✓
Counteraction against session key computation	✓	✓	✓	✓	✓
Perfect forward secrecy	✓	✓	×	×	✓
Counteraction against replay attack	✓	✓	✓	✓	✓
Counteraction against stolen verifier	✓	×	✓	✓	✓
Counteraction against stolen smart card	✓	×	×	×	✓
Counteraction against privileged insider	×	×	✓	✓	✓
Counteraction against denial of service	×	×	×	×	✓
Counteraction against sensor node capture	×	✓	✓	×	✓
Counteraction against man in the middle	×	✓	✓	✓	✓
Counteraction against session specific temporary information	×	✓	✓	✓	✓
Counteraction against gateway impersonation	×	✓	✓	✓	✓

table shows that the proposed protocol ensures all the security functionalities, whereas other protocols lack one or the other. Here, we elaborate upon the details of these security functionalities.

- **Anonymity:** We say that a protocol satisfies the anonymity when the identity of a system's user (user, sensor, gateway) cannot be determined from public communications, contents of the smart card/device, or information kept in the memories of other system's users.
- **Un-traceability:** When an adversary cannot find the similarity between two different messages on comparing them and the message's source is not identified, then it is said to satisfy un-traceability.
- **Entity impersonation:** When an adversary acts as a legitimate system's user (user, sensor, gateway) to deceive the system, then it is called entity impersonation.
- **Offline password guessing:** When an adversary can predict the user's password from the public communications, contents of the smart card/device, or information kept in the memories of other system users (user, sensor, gateway), then it is called offline password guessing.
- **Perfect mutual authentication:** When all of the system's participants authenticate each other pairwise, it refers to perfect mutual authentication.
- **Session key computation:** The session key established between the communicating entities should be robust enough that if any long-term secrets/keys or any parameter of the session key itself is disclosed to an adversary, the adversary's formulation of the session key should remain infeasible.
- **Perfect forward secrecy:** Perfect forward secrecy means that forming the current session key is impossible even if an adversary obtains the current session's long-term secrets.
- **Replay attack:** The replay attack occurs when a legitimate communication is delayed or reiterated to thwart the system's working.
- **Stolen verifier:** The stolen verifier attack occurs when an adversary gains access to the gateway's or System Administrator's (SA) tables and uses that information to construct the session key or obstruct the system's functioning.
- **Stolen smart card:** The stolen smart card attack occurs when an adversary steals the user's smart card and carries out power analysis over it to extract the card's contents and use that information to construct the session key.
- **Privileged insider:** A privileged insider attack occurs when a legitimate system user acts as an adversary and attempts to thwart the system to commit fraud.
- **Denial of service:** The denial-of-service attack occurs when an adversary overwhelms the system with bogus messages, causing resource depletion. As a result, the required resources cannot be allocated to a valid user, and the service is denied.
- **Sensor node capture:** The sensor node capture attack occurs when an adversary captures the sensor node from the field and carries out power analysis over it to extract

the contents kept in its memory and use that information to construct the session key.

- **Man-in-the-middle:** The man-in-the-middle attack occurs when an adversary places himself amid the system's user (user, gateway, sensor). The man either eavesdrops or impersonates one of the legitimate parties, making it appear as a legal message interchange. The goal is to construct the session key and the presence of man going undetected.
- **Session specific temporary information:** Session specific temporary information attack occurs when any session specific temporary information is revealed to an adversary, and with the help of that, he can construct the session key.

7 Conclusion

In this paper, we have scrutinized the Wu et al.'s protocol and found that it suffers from various attacks that include privileged-insider, man-in-the-middle, gateway impersonation, sensor node impersonation, denial-of-service, session-specific transient information, and sensor node capture. Further, it is unable to impart proper confidentiality and anonymity, has key leakage, and incurs more communication overhead. We have discussed a four-factor authentication protocol for resource mining. We have validated the security strength of the proposed protocol using ROR model, BAN logic, and ProVerif simulation. The proposed protocol achieves more security features than similar protocols. The proposed protocol can accomplish desired security features with reduced overheads, proving that it meets the energy efficiency requirements and it is suitable for any practical resource constraint environment.

References

1. Shamshirband, S.; Joloudari, J.H.; Shirkharkolaie, S.K.; Mojrian, S.; Rahmani, F.; Mostafavi, S.; Mansor, Z.: Game theory and evolutionary optimization approaches applied to resource allocation problems in computing environments: a survey. Math. Biosci. Eng. **18**(6), 9190–9232 (2021)
2. Shamshirband, S.; Fathi, M.; Chronopoulos, A.T.; Montieri, A.; Palumbo, F.; Pescapè, A.: Computational intelligence intrusion detection techniques in mobile cloud computing environments: review, taxonomy, and open research issues. J. Inf. Secur. Appl. **55**, 102582 (2020)
3. Samuel, O.; Omojo, A.; Onuja, A.; Sunday, Y.; Tiwari, P.; Gupta, D.; Hafeez, G.; Yahaya, A.; Fatoba, O.; Shamshirband, S.: Iomt: a Covid-19 healthcare system driven by federated learning and blockchain. IEEE J. Biomed. Health Inf. (2022)
4. Wu, F.; Li, X.; Xu, L.; Vijayakumar, P.; Kumar, N.: A novel three-factor authentication protocol for wireless sensor networks with IoT notion. IEEE Syst. J. **15**, 1120–1129 (2020)
5. Althobaiti, O.; Al-Rodhaan, M.; Al-Dhelaan, A.: An efficient biometric authentication protocol for wireless sensor networks. Int. J. Distrib. Sens. Netw. **9**(5), 407971 (2013)
6. Das, A.K.: A secure and effective biometric-based user authentication scheme for wireless sensor networks using smart card and fuzzy extractor. Int. J. Commun Syst **30**(1), e2933 (2017)
7. Turkanović, M.; Brumen, B.; Hölbl, M.: A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the internet of things notion. Ad Hoc Netw. **20**, 96–112 (2014)
8. Amin, R.; Islam, S.H.; Biswas, G.; Khan, M.K.; Leng, L.; Kumar, N.: Design of an anonymity-preserving three-factor authenticated key exchange protocol for wireless sensor networks. Comput. Netw. **101**, 42–62 (2016)
9. Choi, Y.; Lee, D.; Kim, J.; Jung, J.; Nam, J.; Won, D.: Security enhanced user authentication protocol for wireless sensor networks using elliptic curves cryptography. Sensors **14**(6), 10081–10106 (2014)
10. Kumari, S.; Om, H.: Authentication protocol for wireless sensor networks applications like safety monitoring in coal mines. Comput. Netw. **104**, 137–154 (2016)
11. Kumar, D.; Chand, S.; Kumar, B.: Cryptanalysis and improvement of an authentication protocol for wireless sensor networks applications like safety monitoring in coal mines. J. Ambient. Intell. Humaniz. Comput. **10**(2), 641–660 (2019)
12. Ansari, A.A.; Gera, P.; Mishra, B.; Mishra, D.: A secure authentication framework for WSN-based safety monitoring in coal mines. SÂdhanÂ **45**, 98 (2020)
13. Kumar, P.; Lee, S.G.; Lee, H.J.: E-sap: efficient-strong authentication protocol for healthcare applications using wireless medical sensor networks. Sensors **12**(2), 1625–1647 (2012)
14. He, D.; Kumar, N.; Chen, J.; Lee, C.C.; Chilamkurti, N.; Yeo, S.S.: Robust anonymous authentication protocol for health-care applications using wireless medical sensor networks. Multimedia Syst. **21**(1), 49–60 (2015)
15. Deokar, S.; Wakode, J.: Coal mine safety monitoring and alerting system. Int. Res. J. Eng. Technol. (IRJET) **4**(3), 2146–2149 (2017)
16. Dolev, D.; Yao, A.: On the security of public key protocols. IEEE Trans. Inf. Theory **29**(2), 198–208 (1983)
17. Canetti, R.; Krawczyk, H.: In: International conference on the theory and applications of cryptographic techniques. (Springer, 2001), pp. 453–474
18. Canetti, R.; Krawczyk, H.: In: International Conference on the Theory and Applications of Cryptographic Techniques. (Springer, 2002), pp. 337–351
19. Messerges, T.S.; Dabbish, E.A.; Sloan, R.H.: Examining smartcard security under the threat of power analysis attacks. IEEE Trans. Comput. **51**(5), 541–552 (2002)
20. Khan, M.K.; Alghathbar, K.: Cryptanalysis and security improvements of 'two-factor user authentication in wireless sensor networks'. Sensors **10**(3), 2450–2459 (2010)
21. Bhushan, B.; Sahoo, G., Rai, A.K.: In: 2017 3rd International Conference on Advances in Computing, Communication & Automation (ICACCA)(Fall) (IEEE, 2017), pp. 1–6
22. Tribedi, D.; Sadhukhan, D.; Ray, S.: In: International Conference on Computational Intelligence, Communications, and Business Analytics. (Springer, 2018), pp. 411–424
23. Liu, H.: In: Proceedings of the 2010 ACM Workshop on Cloud Computing Security Workshop (2010), pp. 65–76
24. Vemishetty, N.; Patra, P.; Jha, P.K.; Chivukula, K.B.; Vala, C.K.; Jagirdar, A.; Gudur, V.Y.; Acharyya, A.; Dutta, A.: Low power personalized ECG based system design methodology for remote cardiac health monitoring. IEEE Access **4**, 8407–8417 (2016)
25. ADC-Converter-Basics (2011 (accessed August 10, 2020)). https://www.slideshare.net/hacker1500/adc-converter-basics

26. Ku, W.C.; Chang, S.T.: Impersonation attack on a dynamic ID-based remote user authentication scheme using smart cards. IEICE Trans. Commun. **88**(5), 2165–2167 (2005)
27. Wu, Z.; Gao, S.; Cling, E.S.; Li, H.: In: *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific* (IEEE, 2014), pp. 1–5
28. Salem, M.B.; Hershkop, S.; Stolfo, S.J.: In: Insider Attack and Cyber Security (Springer, 2008), pp. 69–90
29. Kumar, V.; Kumar, R.; Pandey, S.: Polynomial based non-interactive session key computation protocol for secure communication in dynamic groups. Int. J. Inf. Technol. **12**(1), 283–288 (2020)
30. Sarvabhatla, M.; Reddy, M.C.M.; Vorugunti, C.S.: In: 2015 Applications and Innovations in Mobile Computing (AIMoC) (IEEE, 2015), pp. 164–169
31. Boyd, C.; Mao, W.: In: Workshop on the Theory and Application of Cryptographic Techniques (Springer, 1993), pp. 240–247
32. Wessels, J.; BV, C.F.: Application of ban-logic. CMG FINANCE BV **19**, 1–23 (2001)
33. Cohen, M.; Dam, M.: In: Methods for Modalities, vol. 4 (2005)
34. Alsalhi, I.N.; Albermany, S.A.: Authentication of CRNS by using ban logic
35. Wazid, M.; Das, A.K.; Kumar, N.; Vasilakos, A.V.: Design of secure key management and user authentication scheme for fog computing services. Futur. Gener. Comput. Syst. **91**, 475–492 (2019)
36. Abdalla, M.; Chevassut, O.; Fouque, P.A.; Pointcheval, D.: In: International Conference on the Theory and Application of Cryptology and Information Security (Springer, 2005), pp. 566–584
37. Das, A.K.; Wazid, M.; Kumar, N.; Vasilakos, A.V.; Rodrigues, J.J.: Biometrics-based privacy-preserving user authentication scheme for cloud-based industrial internet of things deployment. IEEE Internet Things J. **5**(6), 4900–4913 (2018)
38. Wazid, M.; Das, A.K.; Vasilakos, A.V.: Authenticated key management protocol for cloud-assisted body area sensor networks. J. Netw. Comput. Appl. **123**, 112–126 (2018)
39. Das, A.K.; Wazid, M.; Yannam, A.R.; Rodrigues, J.J.; Park, Y.: Provably secure ECC-based device access control and key agreement protocol for IoT environment. IEEE Access **7**, 55382–55397 (2019)
40. Srinivas, J.; Mishra, D.; Mukhopadhyay, S.: A mutual authentication framework for wireless medical sensor networks. J. Med. Syst. **41**(5), 1–19 (2017)
41. Ryan, M.D.; Smyth, B.: Applied pi calculus. Formal Models and Techniques for Analyzing Security Protocols **5**, 112–142 (2011)
42. Kilinc, H.H.; Yanik, T.: A survey of SIP authentication and key agreement schemes. IEEE Commun. Surv. Tutor. **16**(2), 1005–1023 (2013)
43. Das, A.K.; Sutrala, A.K.; Kumari, S.; Odelu, V.; Wazid, M.; Li, X.: An efficient multi-gateway-based three-factor user authentication and key agreement scheme in hierarchical wireless sensor networks. Secur. Commun. Netw. **9**(13), 2070–2092 (2016)
44. Shnayder, V.; Hempstead, M.; Chen, B.R.; Allen, G.W.; Welsh, M.: In: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (2004), pp. 188–200