Sādhanā (2020)45:98
https://doi.org/10.1007/s12046-020-1319-2

A secure authentication framework for WSN-based safety monitoring in coal mines

AMAN AHMAD ANSARI, POONAM GERA, BHARAVI MISHRA* and DHEERENDRA MISHRA

The LNM Institute of Information Technology, Jaipur, India
e-mail: 16pcs001@lnmiit.ac.in; poonamgera@lnmiit.ac.in; bharavi@lnmiit.ac.in; dheerendra.mishra@lnmiit.ac.in

MS received 12 March 2019; revised 21 January 2020; accepted 22 January 2020

Abstract. Underground coal mines are considered as one of the most dangerous workspaces as many hazardous factors regularly cause accidents. It may be avoided by real-time monitoring of environmental parameters (gas, temperature, the width of walls, etc.) of underground tunnels. Nowadays, wireless sensor network (WSN) is widely used for safety monitoring of coal mines. However, any kind of interception, modification and interruption of transmission of environmental parameters can mislead the professionals that might lead to a major accident. Therefore, security is an essential issue for WSN-based safety monitoring. Sensors in WSN have limited computational power and storage capacity, which creates a challenge to design authentication and key agreement (AKA) scheme with low computational cost. To address these issues, As this scheme is light-weight and provides mutual authentication (MA), sensor anonymity (SA) and user anonymity (UA), we have considered this scheme for a case study. We first propose the cryptanalysis of this scheme, in which we prove that this scheme fails to resist sensor node compromise (SNC), stolen smart card (SSC) and user impersonation (UI) attacks. To counteract these attacks and to provide efficient authentication scheme, we propose a lightweight AKA scheme for WSN-based safety monitoring in coal mines. We simulated the scheme on AVISPA tool. We used the random oracle model (ROM) to perform formal security analysis and also performed informal security analysis. These analyses demonstrate that the proposed scheme is secure and invulnerable to various known attacks. We compared the proposed scheme to other related schemes regarding security features and computational cost. Our scheme requires comparable computational cost and is more secure than related schemes.

Keywords. User authentication; sensor node; safety monitoring; wireless sensor network; coal mines; session key agreement.

1. Introduction

Underground coal mines are considered as one of the most dangerous workspaces. For example, some major mine accidents already happened in the past: 1,549 people lost their lives in an explosion at Benxihu (Honkeiko) colliery China, in April 1942; there were 182 casualties in explosions at Chinakuri colliery in February 1958; 268 people killed in the Dhanbad coal mine explosion in May 1965; in December 1975 at Chasnala coal mine, 372 workers lost their lives when mines were flooded after the explosion; 426 people killed by multiple explosions at Wankie colliery in Rhodesia (Zimbabwe) in June 1972. In September 1995, 64 people died due to the breaching of the Katri River in Gaslitand coal mine. In November 2010, 29 workers died because of methane explosions and toxic atmosphere after the explosion at Pike River mine. Kumar *et al* [1] proposed an AKA scheme for WSN-based safety monitoring in coal mines.These kinds of hazards may be avoided with real-time monitoring of environmental parameters (gas, temperature, the width of walls, etc.) of underground tunnels. The professionals can detect the density of gases and dust, and can take preventive measures. Monitoring temperature, early detection of the fire with location can help the fire prevention system to save the lives of workers, and reduce the loss of resources.

Many researchers have suggested monitoring of underground coal mines using wireless sensor network (WSN) [1–10]. WSN refers to a network of spatially dispersed and dedicated sensors (figure 1) to monitor the environmental and physical condition and send the collected data to the user through the gateway node that provides connectivity to other networks [2]. Sensors are low-powered, low-cost autonomous devices with sensing for data acquisition,

*For correspondence
Published online: 30 April 2020

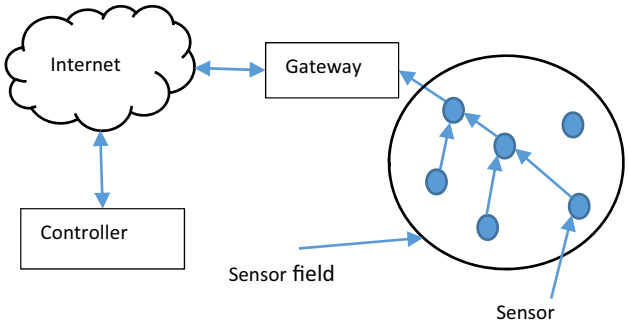

Figure 1. A typical WSN architecture.

processing subsystem for local data processing and radio subsystem for communication [11]. To power up the sensor node, a power unit is also attached to the sensor.

For smooth operation, the safety monitoring system (figure 2) needs reliable data with a real-time guarantee. To ensure the reliability of data, the security scheme must resist the possible attacks on WSN and authenticate the communicating parties correctly. The attacks vary from eavesdropping to modification and interruption of transmission that could result in unauthorized access of sensitive information; modified or vague information pushed by fake or compromised sensors can mislead the professionals that might lead to a major accident. To prevent unauthorized parties from getting into the system, the system requires an authentication and key agreement (AKA) scheme to authenticate all the communicating parties before the actual transmission. In the coal mine environment, to achieve proper monitoring without any blind spot, sensors need to be deployed in hard to reach areas; hence, replacing the power source is not an easy task as sensors have limited power backup. This work aimed to provide a lightweight AKA scheme to authenticate the communicating parties and key agreement with less computation and minimal communication involving sensors for reducing the sensors' load. Hence, it can reduce the power requirement of the sensors, which will extend the lifetime of sensors. In the last few years, many AKA schemes have been discussed to secure WSN.

In [12], the authors discussed a scheme to provide AKA in a WSN environment. Das [13] showed that this scheme [12] is safe against many logged-in users (MLIU) attack and designed a scheme. Tseng *et al* [14] demonstrated that [12] is not secured against replay attack and forgery attack and proposed a scheme by removing these security flaws. The scheme given in [13] is proved to be unsafe in [15] and [16]. Huang *et al* [15] found the scheme's [13] vulnerability against sensor node impersonation (SNI) attack and MLIU attack and designed a scheme by removing these security flaws. He *et al* [16] find that the scheme in [13] cannot withstand privileged insider (PI) and SNI attacks, and the authors presented a new scheme. Ko [17] showed that the scheme [14] did not provide mutual authentication (MA) and proposed a lightweight AKA scheme. Vaidya *et al* [18] reported that schemes given in [12, 14, 17] are vulnerable to stolen verifier (SV) attack and suggested a new scheme. Kim *et al* [19] reported that [18] is weak against user impersonation (UI), gateway-node bypassing (GNB) and password guessing (PG) attacks.

Yeh *et al* [20] designed a scheme to provide AKA in WSN using ECC. Shi and Gong [21] presented an AKA scheme based on ECC, claiming that it is more efficient and secure than [20]. Choi *et al* [22] reported that the scheme in [21] is not secured against stolen smart card (SSC) attack, session key (SSK) attack and sensor energy exhausting (SEE) attack.

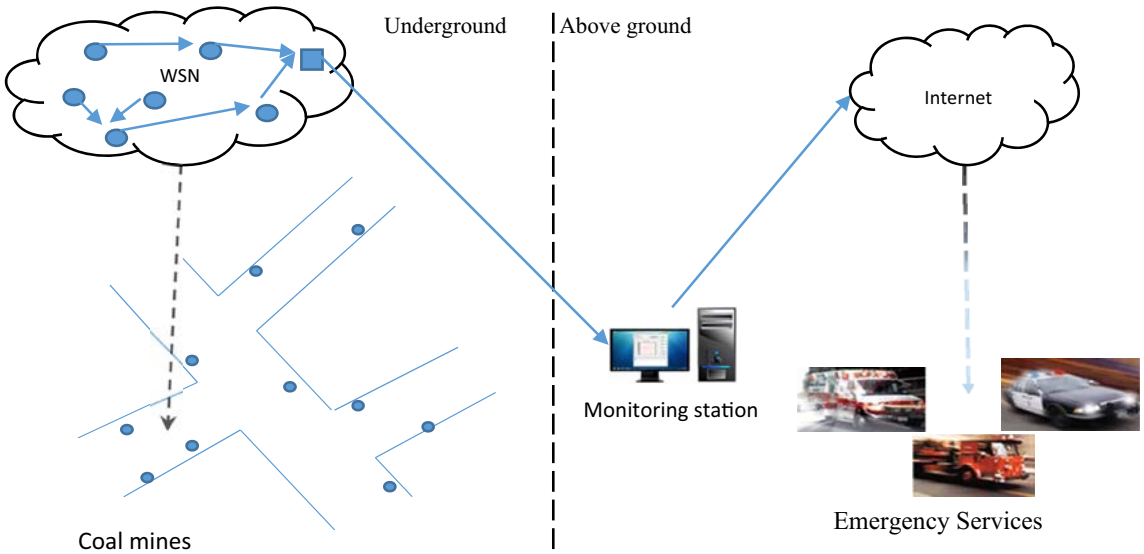

Figure 2. Architecture of the safety monitoring system.

Kumar *et al* [23] proposed the AKA scheme for the medical purpose sensor network, claiming that the scheme is capable of satisfying all the security requirements. He *et al* [24] find that the scheme given in [23] cannot withstand PI and SSC attacks. Nam *et al* [25] point out that the scheme of [24] fails to provide user anonymity (UA) and cannot withstand the SSC attack. Li *et al* [26] reported that [24] fails to provide the session key agreement (SKA) and MA. Xue *et al* [27] discussed temporal-credential-based AKA scheme. Li *et al* [28] shown that the scheme of [27] suffers from SSC, PG, SV and PI attacks. Turkanovic *et al* [29] discussed an AKA scheme for ad-hoc WSN only using hash functions. However, Chang and Le [30] report that the scheme [29] does not achieve forward secrecy (FS) and backward secrecy (BS) and is weak against sensor node spoofing (SNS), SSC and UI attacks.

Kumari and Om [31] gave an AKA scheme for WSN-based safety monitoring system for coal mines. Kumar *et al* [1] performed the cryptanalysis on the scheme presented in [31] and reported that the scheme did not achieve sensor anonymity (SA) and user un-traceability (UT) and was vulnerable to SSC, denial-of-service (DoS) and SV attacks, and proposed a scheme addressing these issues. However, this scheme is not resistant to sensor node compromise (SNC) attack, SSC attack and UI attack. We address these security issues in this paper and propose an improved scheme.

1.1 *Threat model*

To evaluate the security of the scheme, we introduce a threat model in which an attacker ($\mathcal{A}$) has several advantages or capabilities. The assumptions about the $\mathcal{A}$'s capabilities under threat model are the following:

1. $\mathcal{A}$ cannot intercept messages communicated through a private channel.
2. $\mathcal{A}$ can eavesdrop, modify, insert, resend, reroute and delete the communicated messages over public channel.
3. If $\mathcal{A}$ gets the smart card (SC) of a valid user, $\mathcal{A}$ can extract information from SC's memory.
4. $\mathcal{A}$ can guess all the possible combinations of identity (UID_i) and password (PW_i) of the user in a reasonable amount of time.
5. $\mathcal{A}$ can extract information from sensors' memory of physically captured sensors.
6. $\mathcal{A}$ may be a legitimate user.

2. Review of Kumar *et al* scheme

A review of the scheme in [1] is given in this section. Table 1 displays the notations that appear in this paper.

Table 1. Notations.

U_i	i^{th} user
RC	Registration centre
SN_k	k^{th} sensor
X_c	RC's secret key
SC	Smart card
GW_j	j^{th} gateway
$\mathcal{A}$	An attacker
UID_i	U_i's identity
b_i	Random number generated by user U_i
GID_j	GW_j's identity
SID_k	Sensor SN_k's identity
PW_i	U_i's password
G_j	GW_j's secret key
SIX_k	The shared key between GW_j and SN_k
BIO_i	U_i's biometric imprint
R_i	Nonce generated by U_i
B_i	Secret bit string generated from U_i's biometric imprint
A_i	Auxiliary bit string generated from U_i's biometric imprint
R_j	Nonce generated by GW_j
SK_i	Session key
R_k	Nonce generated by SN_k
$\oplus$	Ex-OR operator
$\|\|$	Concatenation operator
$h(.)$	One-way hash function

2.1 *Sensor and gateway node registration phase*

To register gateways and sensors, RC executes the following steps.

RC selects the secret key X_c. RC selects identity GID_j, and calculates the secret key $G_j = h(GID_j||X_c)$ for gateway GW_j.
RC selects identity SID_k and calculates $SIX_k = h(SID_k||X_c)$ for sensor node SN_k.
RC stores $\{SID_k, SIX_k\}$ in the memory of SN_k before deploying the sensor, and stores $\{GID_j, G_j, SID_k, SIX_k\}$ in the gateway.

2.2 *User registration phase*

When U_i requests GW_j for registration, the gateway performs following steps to register U_i and provide an SC to him.

U_i selects his UID_i, PW_i and a random number b_i, and sends $C_i = h(UID_i||b_i)$ and $P_i = h(PW_i||b_i)$ to GW$_j$ via a secure channel.
GW_j generates V_i and l ($2^4 \leq l \leq 2^8$) for fuzzy verifier; then computes $D_2 = V_i \oplus h(G_j)$ and $f_3 = h(V_i||G_j) \oplus$

$h(C_i||P_i)$, stores $\{D_2, h(.), f_3, l\}$ into SC and sends it to U_i through a secure channel.
U_i calculates $c = b_i \oplus h(UID_i||PW_i) \bmod l$, $HB_i = h(P_i||C_i||b_i) \bmod l$ and stores c and HB_i into the SC. Thus, SC contains $\{D_2, h(.), f_3, l, HB_i, c\}$.

2.3 *Login phase*

To log in to GW_j, U_i executes the following:

U_i inserts SC into card reader, then types in his UID_i and PW_i. SC computes $b_i = c \oplus h(UID_i||PW_i) \bmod l$, $C_i = h(UID_i||b_i)$, $P_i = h(PW_i||b_i)$ and $HB_i = h(P_i||C_i||b_i) \bmod l$. To verify the UID_i and PW_i by checking if $HB_i^* \stackrel{?}{=} HB_i$. If false, SC aborts the session. Random number R_i is generated by SC as a nonce, then SC computes $L_2 = D_2 \oplus T_1$, $B_1 = f_3 \oplus h(C_i||P_i)$, $B_2 = h(T_1||R_i||B_1)$ and $B_3 = (R_i||T_1) \oplus B_1$, where T_1 is current timestamp. Then SC sends a log-in message $\{L_2, B_2, B_3, T_1\}$ to GW_j via a public channel.

2.4 *AKA phase*

On receiving the log-in request, GW_j computes $V_i^* = L_2 \oplus h(G_j) \oplus T_1$, $B_1^* = h(V_i^*||G_j), (R_i^*||T_1^*) = B_1^* \oplus B_3$. GW_j checks if timestamp T_1 is fresh. If false, GW_j aborts the session; else, GW_j computes $B_2^* = h(T_1^*||R_i^*||B_1^*)$ and U_i is authenticated if $B_2^* = B_2$. If not correct, GW_j terminates the session: else, GW_j generates a nonce R_j, and computes $f_4 = h(SID_k||\ V_i^*||SIX_k||R_j||T_2)$, $f_5 = (R_i^*||R_j||T_2) \oplus SIX_k$ and $f_6 = V_i^* \oplus h(SID_k||\ h(R_j)||R_i^*)$. Then GW_j communicates $\{f_4, f_5, f_6\}$ to SN_k.

On receiving $\{f_4, f_5, f_6\}$ at time T_3, SN_k computes $(R_i^{**}||R_j^*||T_2^*) = f_5 \oplus SIX_k$. SN_k checks if timestamp T_2^* is fresh. If T_2^* is fresh, SN_k calculates $V_i^* = f_6 \oplus h(SID_k||h(R_j^*)||R_i^{**})$ and $f_4^* = h(SID_k||V_i^*||SIX_k||R_j^*||T_2^*)$ and then compares $f_4^* \stackrel{?}{=} f_4$. If not correct, SN_k aborts the session; otherwise, GW_j is authenticated. SN_k generates R_k randomly and computes session key $SK_i = h(R_i^{**}||R_j^*||R_k)$, $C_1 = h(T_3||R_k||SK_i||SIX_k||SID_k||T_2^*)$ and $C_2 = (R_k||T_3) \oplus R_j^*$. Then transmits $\{C_1, C_2\}$ to GW_j.

On receiving $\{C_1, C_2\}$ from SN_k, GW_j computes $(R_k^*||T_3^*) = C_2 \oplus R_j$ and checks if T_3^* is fresh GW_j, computes $SK_i = h(R_i^*||R_j||R_k^*)$ and $C_1^* = h(T_3^*||R_k^*||SK_i||SIX_k||SID_k||T_2)$. Then GW_j checks if $C_1^* \stackrel{?}{=} C_1$, then SN_k is authenticated and GW_j calculates $f_7 = h(SK_i||R_j||T_4||f_4)$ and $f_8 = (R_k^*||R_j||T_4) \oplus R_i^*$. GW_j sends $\{f_4, f_7, f_8\}$ to U_i.

The SC computes $(R_k^{**}||R_j^*||T_4^*) = f_8 \oplus R_i$ and checks out if T_4^* is fresh, then SC computes $SK_i = h(R_i||R_j^*||R_k^{**})$, $f_7^* = h(SK_i||R_j^*||T_4^*||f_4)$ and compares $f_7^* \stackrel{?}{=} f_7$. If true, GW_j and SN_k are authenticated by U_i.

2.5 *Password change phase*

This phase explains the steps to change his/her password by U_i.

U_i insert SC into card reader, then types in his UID_i^* and PW_i^*. SC computes $b_i = c \oplus h(UID_i^*||PW_i^*) \bmod l$, $P_i^* = h(PW_i^*||b_i^*)$, $C_i^* = h(UID_i^*||b_i^*)$ and $HB_i^* = h(P_i^*||C_i^*||b_i^*) \bmod l$. SC verifies the ID_i and PW_i of U_i by comparing $HB_i^* \stackrel{?}{=} HB_i$. If not, SC aborts the session.
U_i is requested for new random nonce b_{inew} and password PW_{inew}. SC computes $C_{inew} = h(UID_i^*||b_{inew})$, $P_{inew} = h(PW_{inew}||b_{inew})$, $HB_{inew} = h(P_{inew}||C_{inew}||b_{inew}) \bmod l$, $f_{3new} = f_3 \oplus h(C_i^*||P_i^*) \oplus h(C_{inew}||P_{inew})$ and $c_{new} = b_{inew} \oplus h(UID_i^*||PW_{inew}) \bmod l$. Then SC replaces HB_i, c and f_3 with, respectively, HB_{inew}, c_{new} and f_{3new}.

2.6 *Adding a new sensor node*

If there is a requirement for a new sensor node, it can be added by executing these steps.

RC choose SID_k and compute $SIX_k = h(SID_k||X_c)$ for a new sensor. RC saves $\{SID_k, SIX_k\}$ in the new sensor's memory and hands over to the technician to deploy.
To register SN_k with the closest GW_j, RC transmits $\{SID_k, SIX_k\}$. GW_j stores this value in its database.

3. Cryptanalysis of Kumar *et al* scheme

This section will provide cryptanalysis of Kumar *et al* [1] scheme and show that it does not provide security against the following attacks:

3.1 *Node compromise attack*

Assuming that $\mathcal{A}$ captures a node SN_k, $\mathcal{A}$ knows secret parameters stored at the sensor node (i.e., the identity of the sensor SID_k and secret SIX_k). Using these secret parameters and the messages communicated through public channel, $\mathcal{A}$ can compute R_i, R_j, R_k and SK_i shared among U_i, GW_j, and SN_k by performing the following steps:

$$(R_i||R_j||T_2) = f_5 \oplus SIX_k$$

where R_i and R_j are nonce generated by user and gateway, respectively, and T_2 is a timestamp.

$$(R_k||T_3) = C_2 \oplus R_j$$

where R_k is a nonce generated by the sensor node and T_3 is a timestamp. Using R_i, R_j and R_k, the attacker can calculate $SK_i = h(R_i||R_j||R_k)$

The attacker can use R_i and R_j to get the knowledge of secret parameter B_1 and the secret pseudo-identity V_i in the following manner:

$$B_1 = B_3 \oplus (R_i||T_1),$$
$$V_i = f_6 \oplus h(SID_k||h(R_j)||R_i).$$

3.2 *User impersonation attack*

Using V_i ($\mathcal{A}$ knows V_i and B_1 by performing node compromise attack) $\mathcal{A}$ can get $h(G_j) = L_2 \oplus V_i \oplus T_1$, where T_1 is a timestamp; $h(G_j)$ can be used to extract the secret pseudo-identity of any user using L_2 and T_1 from their login request message. To perform user impersonation, attacker $\mathcal{A}$ will choose a nonce $R_{\mathcal{A}}$ and a timestamp $T_{\mathcal{A}1}$, then compute

$$L_{\mathcal{A}2} = V_i \oplus h(G_j) \oplus T_{\mathcal{A}1}$$
$$B_{\mathcal{A}2} = h(T_{\mathcal{A}1}||R_{\mathcal{A}}||B_1)$$
$$B_{\mathcal{A}3} = B_1 \oplus (R_{\mathcal{A}}||T_{\mathcal{A}1})$$

and send it to gateway GW_j. After receiving login request, GW_j computes

$$V_i^* = L_{\mathcal{A}2} \oplus h(G_j) \oplus T_{\mathcal{A}1},$$
$$B_1^* = h(V_i^*||G_j),$$
$$(R_{\mathcal{A}}^*||T_{\mathcal{A}1}^*) = B_{\mathcal{A}3} \oplus B_1^*,$$
$$B_2^* = h(T_{\mathcal{A}1}^*||R_{\mathcal{A}}^*||B_1^*).$$

B_2^* is equivalent to $B_{\mathcal{A}2}$ as both are computed from the same parameters. For $B_2^* = B_{\mathcal{A}2}$, GW_j authenticates the attacker $\mathcal{A}$ as user U_i.

3.3 *SSC attack*

$\mathcal{A}$ can extract $\{D_2, HB_i, f_3, l, c, h(.)\}$ stored in SC memory by several methods [32, 33], where $f_3 = h(V_i||G_j) \oplus h(C_i||P_i) = B_1 \oplus h(C_i||P_i)$, $D_2 = V_i \oplus h(G_j)$, $P_i = h(PW_i||b_i)$, $C_i = h(UID_i||b_i)$, $HB_i = h(P_i||C_i||b_i) \bmod l$ and $c = b_i \oplus h(UID_i||PW_i) \bmod l$.

Then, $\mathcal{A}$ executes following steps to guess the password:

- $\mathcal{A}$ guesses the UID^* and PW^* as, respectively, identity and password of U_i.
- Then $\mathcal{A}$ computes

$$b_i^* = c \oplus h(UID^*||PW^*) \bmod l,$$
$$C_i^* = h(UID^*||b_i^*),$$
$$P_i^* = h(PW^*||b_i^*).$$

- Check if $h(C_i^*||P_i^*) \stackrel{?}{=} f_3 \oplus B_1$ (attacker knows B_1 from compromised sensor attack), if they are equal, then attacker $\mathcal{A}$ guessed correct ID_i and PW_i; otherwise, $\mathcal{A}$ repeats the steps for a new guess.

4. Proposed scheme

4.1 *Sensor and gateway node registration phase*

This phase (figure 3) is the same as Kumar *et al*'s scheme [1]. To register gateways and sensors, RC executes the following steps:

- RC selects X_c as secret key.
- RC selects GID_j and computes $G_j = h(GID_j||X_c)$ as identity and secret key for GW_j.
- RC selects SID_k as an identity for sensor node SN_k, then calculates $SIX_k = h(SID_k||X_c)$ as its secret key.
- RC stores $\{SID_k, SIX_k\}$ in SN_k before deploying the sensor.
- RC stores $\{GID_j, G_j, SID_k, SIX_k\}$ in the gateway.

4.2 *User registration phase*

GW_j registers U_i by executing following steps:

- U_i selects UID_i and PW_i as, respectively, his identity and password, stores his biometric imprint BIO_i at the fuzzy extractor, generates $Gen(BIO_i) = (B_i, A_i)$, computes $C_i = h(UID_i||B_i)$ and $P_i = h(PW_i||B_i)$, then transmits C_i and P_i to the GW_j through private channel.
- after generating a unique random number V_i for U_i, GW_j computes

$$f_3 = h(V_i||G_j) \oplus h(C_i||P_i)$$
$$D_2 = V_i \oplus h(G_j)$$

stores $\{D_2, h(.), f_3\}$ into SC and sends it securely to U_i.
- U_i computes $HB_i = h(P_i||C_i||B_i)$, and stores HB_i and A_i into SC.

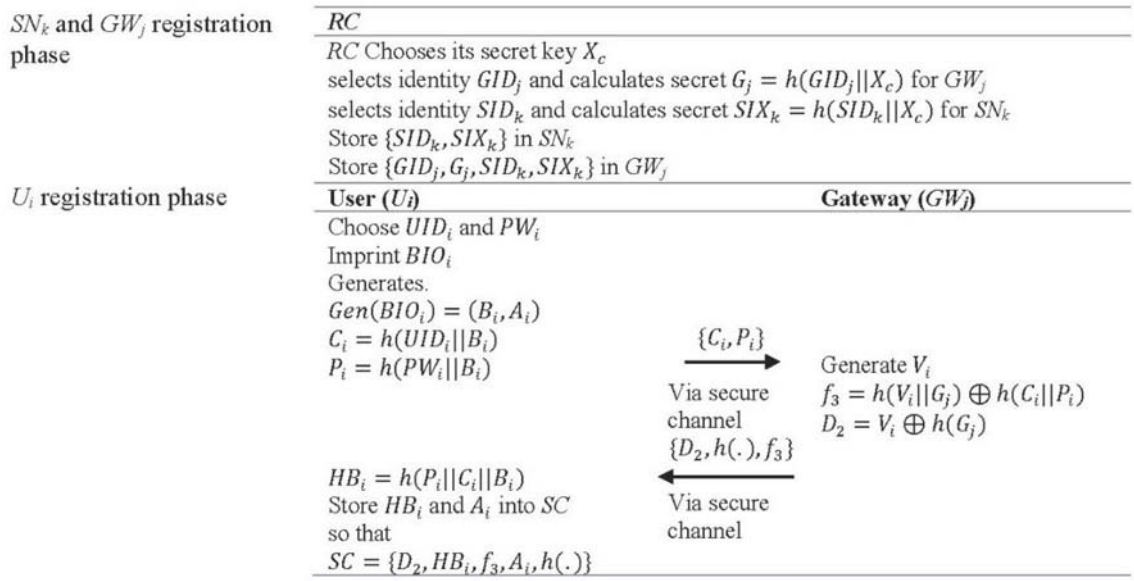

Figure 3. Registration phase.

4.3 *Login, authentication and key agreement phase*

Following are the steps (figure 4) to achieve MA among U_i, GW_j and SN_k:

- U_i inserts SC into card reader, then types in his UID_i and PW_i. U_i inputs his biometric template BIO_i to fuzzy extractor, which reproduces $B_i = Rep(BIO_i, A_i)$.
- SC calculates

$$C_i = h(UID_i||B_i)$$
$$P_i = h(PW_i||B_i)$$
$$HB_i^* = h(C_i||P_i||B_i)$$

  and SC verifies the UID_i and PW_i by checking if $HB_i^* \stackrel{?}{=} HB_i$. If false, SC aborts the session.
- SC generates a nonce R_i and computes

$$L_2 = D_2 \oplus T_1,$$
$$K_i = f_3 \oplus h(C_i||P_i) = h(V_i||G_j),$$
$$B_1 = h(K_i||T_1),$$
$$B_2 = h(T_1||R_i||B_1),$$
$$B_3 = R_i \oplus B_1.$$

  SC sends $\{L_2, B_2, B_3, T_1\}$ to GW_j through unsecure channel.
- On receiving from U_i's login request, GW_j checks $(T_1^* - T_1) \leq \Delta T$. If not, GW_j close the session; else, GW_j computes

$$V_i^* = L_2 \oplus h(G_j) \oplus T_1$$
$$K_i^* = h(V_i^*||G_j)$$
$$B_1^* = h(K_i^*||T_1)$$
$$R_i^* = B_3 \oplus B_1$$
$$B_2^* = h(T_1||R_i^*||B_1^*)$$

  and compares $B_2^* \stackrel{?}{=} B_2$. If true, U_i is authenticated; else, GW_j closes the session.
- GW_j generates the nonce R_j and computes

$$M_1 = h(V_i^*||R_j),$$
$$f_4 = h(SID_k||M_1||SIX_k||R_i^*||T_2),$$
$$f_5 = (R_i^*||R_j) \oplus h(SIX_k||T_2),$$
$$f_6 = M_1 \oplus h(SID_k||SIX_k||R_i^*),$$

  GW_j sends the message $\{f_4, f_5, f_6, T_2\}$ to sensor node SN_k.
- After obtaining the message from gateway GW_j, SN_k checks whether $(T_2^* - T_2) \leq \Delta T$; if not, SN_k drops the session; else, SN_k computes

$$(R_i^{**}||R_j^*) = f_5 \oplus h(SIX_k||T_2)$$
$$M_1^* = f_6 \oplus h(SID_k||SIX_k||R_i^{**})$$
$$f_4 = h(SID_k||M_1^*||SIX_k||R_i^{**}||T_2)$$

  and checks whether computed f_4 matches the received f_4. If not, the session is terminated; else, SN_k authenticates GW_j.

User(U_i)	**Gateway(GW_j)**	**Sensor node(SN_k)**
Ui insert SC into card reader then type in his UID_i and PW_i Imprint BIO_i Achieve $B_i = Rep(BIO_i, A_i)$ Chose a gateway GW_j Card reader calculates $C_i = h(UID_i \| \| B_i)$ $P_i = h(PW_i \| \| B_i)$ $HB_i^* = h(P_i \| \| C_i \| \| B_i)$ For $HB_i^* = HB_i$ SC generates R_i and calculates $L_2 = D_2 \oplus T_1$ $K_i = f_3 \oplus h(C_i \| \| P_i) = h(V_i \| \| G_j)$ $B_1 = h(K_i \| \| T_1)$ $B_2 = h(T_1 \| \| R_i \| \| B_1)$ $B_3 = R_i \oplus B_1$ $\{L_2, B_2, B_3, T_1\}$ →	Check $(T_1^* - T_1) \leq \Delta T$ $V_i^* = L_2 \oplus h(G_j) \oplus T_1$ $K_i^* = h(V_i^* \| \| G_j)$ $B_1^* = h(K_i^* \| \| T_1)$ $R_i^* = B_3 \oplus B_1^*$ $B_2^* = h(T_1 \| \| R_i^* \| \| B_1^*)$ For $B_2^* = B_2$, U_i is authenticated Generates a nonce R_j and calculates $M_1 = h(V_i^* \| \| R_j)$ $f_4 = h(SID_k \| \| M_1 \| \| SIX_k \| \| R_i^* \| \| T_2)$ $f_5 = (R_i^* \| \| R_j) \oplus h(SIX_k \| \| T_2)$ $f_6 = M_1 \oplus h(SID_k \| \| SIX_k \| \| R_i^*)$ $\{f_4, f_5, f_6, T_2\}$ →	
		Check $(T_2^* - T_2) \leq \Delta T$, SN_k calculates $(R_i^{**} \| \| R_j^*) = f_5 \oplus h(SIX_k \| \| T_2)$ $M_1^* = f_6 \oplus h(SID_k \| \| SIX_k \| \| R_i^{**})$ $f_4 = h(SID_k \| \| M_1^* \| \| SIX_k \| \| R_i^{**} \| \| T_2)$ for $f_4^* = f_4$, GW_j is authenticated SN_k generates a nonce R_k and calculates, $SK_i = h(R_i^{**} \| \| R_j^* \| \| R_k)$ $C_1 = h(SK_i \| \| R_k \| \| SID_k \| \| SIX_k \| \| M_1^* \| \| T_3)$ $C_2 = R_k \oplus h(SIX_k \| \| R_j^*)$ $\{C_1, C_2, T_3\}$ ←
	Check $(T_3^* - T_3) \leq \Delta T$, $R_k^* = C_2 \oplus h(SIX_k \| \| R_j)$ $SK_i = h(R_i^* \| \| R_j \| \| R_k^*)$ $C_1^* = h(SK_i \| \| R_k^* \| \| SID_k \| \| SIX_k \| \| M_1 \| \| T_3)$ For $C_1^* = C_1$, SN_k is authenticated $f_7 = h(SK_i \| \| R_j \| \| R_k^* \| \| T_4 \| \| K_i^*)$ $f_8 = (R_j \| \| R_k^*) \oplus h(K_i^* \| \| R_i^*)$ $\{f_7, f_8, T_4\}$ ←	
Check $(T_4^* - T_4) \leq \Delta T$, $(R_j^* \| \| R_k^{**}) = f_8 \oplus h(K_i \| \| R_i)$ $SK_i = h(R_i \| \| R_j^* \| \| R_k^{**})$ $f_7^* = h(SK_i \| \| R_j^* \| \| R_k^{**} \| \| T_4 \| \| K_i)$ For $f_7^* = f_7$, Both GW_j and SN_k are authenticated		

Figure 4. Login, authentication and key agreement phase.

- SN_k generates a nonce R_k, then computes

$$SK_i = h\left(R_i^{**}||R_j^*||R_k\right)$$
$$C_1 = h(SK_i||R_k||SID_k||SIX_k||M_1^*||T_3)$$
$$C_2 = R_k \oplus h(SIX_k||R_j^*)$$

  SN_k sends $\{C_1, C_2, T_3\}$ to GW_j.
- GW_j checks whether $\left(T_3^* - T_3\right) \leq \Delta T$. If not, GW_j closes the session; else, GW_j computes

$$R_k^* = C_2 \oplus h(SIX_k||R_j)$$
$$SK_i = h\left(R_i^*||R_j||R_k^*\right)$$
$$C_1^* = h(SK_i||R_k^*||SID_k||SIX_k||M_1||T_3)$$

  and GW_j checks if $C_1^* \stackrel{?}{=} C_1$. If true, SN_k is authenticated; else, GW_j aborts the session.
- GW_j computes

$$f_7 = h\left(SK_i||R_j||R_k^*||T_4||K_i^*\right)$$
$$f_8 = (R_j||R_k^*) \oplus h(K_i^*||R_i^*)$$

  and sends $\{f_7, f_8, T_4\}$ to U_i.
- On receiving $\{f_7, f_8, T_4\}$ from GW_j, SC checks if timestamp T_4 is fresh. If not, SC aborts the session; else, SC calculates

$$(R_j^*||R_k^{**}) = f_8 \oplus h(K_i||R_i),$$
$$SK_i = h\left(R_i||R_j^*||R_k^{**}\right),$$
$$f_7^* = h\left(SK_i||R_j^*||R_k^{**}||T_4||K_i\right).$$

  For $f_7^* = f_7$, both GW_j and SN_k are authenticated; else, SC aborts the session.

4.4 *Password change phase*

This phase explains the steps to change U_i's password.

- U_i inserts SC into card reader, then types in his UID_i^* and PW_i^*. U_i inputs his biometric template BIO_i^* to fuzzy extractor, which reproduces $B_i^* = Rep\left(BIO_i^*, A_i\right)$.
- SC computes

$$C_i^* = h(UID_i^*||B_i^*),$$
$$P_I^* = h(PW_i^*||B_i^*),$$
$$HB_i^* = h\left(P_i^*||C_i^*||B_i^*\right).$$

- Compares $HB_i^* \stackrel{?}{=} HB_i$. If correct, verifies the identity, password and biometric template of U_i; if not equal, session is terminated.
- U_i is requested to input new password PW_i^{new} and biometric imprint BIO_i^{new}.
- SC calculates the following:

$$\left(B_i^{new}, A_i^{new}\right) = Gen\left(BIO_i^{new}\right),$$
$$C_i^{new} = h(UID_i^*||B_i^{new}),$$
$$P_i^{new} = h(PW_i^{new}||B_i^{new}),$$
$$HB_i^{new} = h\left(P_i^{new}||C_i^{new}||B_i^{new}\right),$$
$$f_3^{new} = f_3 \oplus h(C_i^*||P_i^*) \oplus h(C_i^{new}||P_i^{new}).$$

- The SC replaces HB_i, A_i and f_3 with, respectively, HB_i^{new}, A_i^{new} and f_3^{new}.

4.5 *Adding a new sensor node*

This phase is the same as Kumar *et al*'s scheme [1]. If there is a requirement of new sensor node, it can be added by executing these steps:

- RC choose SID_k and compute $SIX_k = h(SID_k||X_c)$ for a new sensor. RC stores $\{SID_k, SIX_k\}$ into sensor's memory and hands over to the technician to deploy.
- To register SN_k with the closest GW_j, RC transmits $\{SID_k, SIX_k\}$.
- GW_j stores this value in its memory.

5. Security analysis

We provide informal security analysis, formal security analysis using random oracle model (ROM) and simulation with AVISPA of the proposed scheme.

5.1 *Formal security analysis*

This section provides security analysis of our scheme using ROM. We follow the same procedure given by Das [34] and Chandrakar and Om [35]. Definition of the *Reaveal* oracle is given here.

Reveal : return the hash input x from a comparable hash output m, where $m = h(x)$.

Theorem 1 *Assume that $\mathcal{A}$ can extract information from SC's memory and read all the messages communicated through public channel. $\mathcal{A}$ cannot obtain UID_i and PW_i of U_i, and G_j of GW_j. Hence, the proposed scheme is still safe.*

Proof Suppose $\mathcal{A}$ successfully obtains the secrets UID_i, PW_i and G_j from the extracted information from SC's memory (i.e. $\{D_2, HB_i, f_3, A_i, h(.)\}$), and communicates message $\{L_2, B_2, B_3, T_1\}$ and algorithm $ALG1_{\mathcal{A},U}^{HASH}$. The $ALG1_{\mathcal{A},U}^{HASH}$ uses *Reveal* oracle to compute the inverse of hash function $h(.)$. The success probability of $ALG1_{\mathcal{A},U}^{HASH}$

$$P_{succ1} = \left|P\left[ALG1_{\mathcal{A},U}^{HASH} = 1\right] - 1\right|.$$

Algorithm 1 $ALG1_{E,UA}^{HASH}$

1: Extract $\{D_2, HB_i, f_3, A_i, h(.)\}$ from smart card, where $D_2 = V_i \oplus h(G_j)$, $HB_i = h(P_i \| C_i \| B_i)$, $P_i = h(PW_i \| B_i)$, $C_i = h(UID_i \| B_i)$, $f_3 = h(V_i \| G_j) \oplus h(C_i \| P_i)$, A_i and B_i are generated from BIO_i using fuzzy extractor.
2: Intercept the communication message $\{L_2, B_2, B_3, T_1\}$ from a public channel, where $L_2 = D_2 \oplus T_1$, $B_2 = h(T_1 \| R_i \| B_1)$, $B_1 = h(K_i \| T_1)$, $K_i = f_3 \oplus h(C_i \| P_i) = h(V_i \| G_j)$ and $B_3 = R_i \oplus B_1$.
3: Call $(T_1' \| R_i' \| B_1') \leftarrow Reveal(B_2)$.
4: *if* $T_1' = T_1$ *then*
5: $B_1^* = B_3 \oplus R_i'$
6: *if* $B_1^* = B_1'$ *then*
7: Call $(K_i' \| T_1'') \leftarrow Reveal(B_1')$
8: *if* $T_1' = T_1''$ *then*
9: $w_1 = h(C_i \| P_i) = f_3 \oplus K_i'$
10: Call $(C_i' \| P_i') \leftarrow Reveal(w_1)$
11: Call $(UID_i' \| B_i') \leftarrow Reveal(C_i')$
12: Call $(PW_i' \| B_i'') \leftarrow Reveal(P_i')$
13: *if* $B_i' = B_i''$ *then*
14: $HB_i^* = h(P_i' \| C_i' \| B_i')$
15: *if* $HB_i = HB_i^*$ *then*
16: UID_i' and PW_i' are correct identity and password of U_i
17: Call $(V_i' \| G_j') \leftarrow Reveal(K_i')$
18: $D_2' = L_2 \oplus T_1'$
19: *if* $D_2' = \left(V_i' \oplus h(G_j')\right)$ *then*
20: G_j' is correct secret key of GW_j
21: $return(1)$ *success*
22: *else*
23: $return(0)$ *failure*
24: *else*
25: $return(0)$ *failure*
26: *else*
27: $return(0)$ *failure*
28: *else*
29: $return(0)$ *failure*
30: *else*
31: $return(0)$ *failure*
32:*else*
33: $return(0)$ *failure*

The advantage function of $ALG1_{\mathcal{A},\mathrm{U}}^{HASH}$ is $Adv_{ALG1}(t_1, q_{r1}) = MAX_{\mathcal{A}}\{P_{succ1}\}$, where q_{r1} is count of *Reveal* oracle queries and t_1 is the execution time of $ALG1_{\mathcal{A},\mathrm{U}}^{HASH}$. The proposed scheme is safe if $Adv_{ALG1}(t_1, q_{r1}) \leq \epsilon$, for some small $\epsilon > 0$. From $ALG1_{\mathcal{A},\mathrm{U}}^{HASH}$, we see that $\mathcal{A}$ can get UID_i, PW_i and G_j only if $\mathcal{A}$ can reverse the hash function. However, to calculate the inverse of hash function, there is no known method with polynomial time complexity. Hence, $\mathcal{A}$ cannot obtain any secret information UID_i and PW_i of U_i, and G_j of GW_j.

Theorem 2 *Assume that $\mathcal{A}$ can intercept the messages communicated through public channels. Still, $\mathcal{A}$ cannot compute the session key SK_i.*

Proof Suppose $\mathcal{A}$ has the capability to compute SK_i. Assume that $\mathcal{A}$ intercepts the communication messages $(\{f_4, f_5, f_6, T_2\}, \{C_1, C_2, T_3\}, \{f_7, f_8\})$ and uses random oracle *Reveal* to run $ALG2_{\mathcal{A},\mathrm{U}}^{HASH}$ for the proposed scheme. The success probability for $ALG2_{\mathcal{A},\mathrm{U}}^{HASH}$ is defined as

$$P_{succ2} = \left| P\left[ALG2_{\mathcal{A},\mathrm{U}}^{HASH} = 1\right] - 1\right|.$$

Algorithm 2 $ALG2_{\mathcal{A},\mathrm{U}}^{\mathrm{HASH}}$

1: Intercept the messages $(\{f_4, f_5, f_6, T_2\}, \{C_1, C_2, T_3\}, \{f_7, f_8\})$ communicated through public channel.
2: Call $(SK_i' \| R_k' \| SID_k' \| SIX_k' \| M_1' \| T_3') \leftarrow Reveal(C_1)$
3: *if* $T_3 = T_3'$ *then*
4: Call $(SK_i'' \| R_j' \| R_k'' \| T_4' \| K_i') \leftarrow Reveal(f_7)$
5: *if* $T_4 = T_4'$ *then*
6: *if* $SK_i' = SK_i''$ *then*
7: Call $\left(R_i' \| R_j'' \| R_k'''\right) \leftarrow Reveal(SK_i')$
8: $f_8^* = (R_j'' \| R_k''') \oplus h(k_i' \| R_i')$
9 *if* $f_8^* = f_8$ *then*
10: SK_i' is the correct session key
11: $return(1)\ success$
12: *else*
13: $return(0)\ failure$
14: *else*
15: $return(0)\ failure$
16: *else*
17: $return(0)\ failure$

The advantage function for $ALG2_{\mathcal{A},\mathrm{U}}^{HASH}$ is $Adv_{ALG2}(t_2, q_{r2}) = MAX_{\mathcal{A}}\{P_{succ2}\}$, where q_{r2} is count of *Reveal* oracle queries and t_2 is the execution time of $ALG2_{\mathcal{A},\mathrm{U}}^{HASH}$. $\mathcal{A}$ cannot calculate SK_i if $Adv_{ALG2}(t_2, q_{r2}) \leq \epsilon$, for small $\epsilon > 0$. We can see from algorithm $ALG2_{\mathcal{A},\mathrm{U}}^{HASH}$ that $\mathcal{A}$ can compute SK_i only if $\mathcal{A}$ can reverse the hash function. However, hash functions have pre-image resistance; that is, it is hard to compute the pre-image from the image. Hence, $\mathcal{A}$ cannot calculate SK_i.

5.2 *Informal security analysis*

In this section, we discuss security issues to prove that the proposed scheme provides UA, SA, MA and UT, and it is resilient to UI, PG, SNC, SNI and replay attacks under the given threat model.

5.2a *MA*: GW_j authenticates U_i by computing B_2 and SN_k by computing C_1. SN_k authenticates the GW_j by computing f_4 and finally, U_i authenticates GW_j and SN_k by computing f_7.

5.2b *Key agreement*: All three parties contribute to the generation of session key. Initially, U_i generates R_i and computes $B_3 = R_i \oplus h(K_i||T_1)$, R_i is protected by secret $K_i = h(V_i||G_j)$, the GW_j also generates R_j and securely transmits (R_i, R_j) to the SN_k by computing $f_5 = (R_i||R_j) \oplus h(SIX_k||T_2)$, R_i and R_j are protected by the shared secret between GW_j and SN_k (i.e., SIX_k) and one-way hash function. SN_k generates a nonce R_k and computes $C_2 = R_k \oplus h(SIX_k||R_j)$, R_k secured by shared secret SIX_k and one-way hash function. R_j and R_k are shared securely with U_i by GW_j using $f_8 = (R_i, R_j) \oplus h(K_i, R_i)$, secured with secret K_i and one-way hash function. On completing these steps, all 3 parties know R_i, R_j and R_k. They are parts of $SK_i = h(R_i||R_j||R_k)$.

5.2c *UA*: The proposed scheme never sends UID_i to other parties. When U_i sends $\{L_2, B_2, B_3, T_1\}$ to the GW_j, user secret V_i (a unique random number assigned by *RC*) is transmitted with $L_2 = V_i \oplus h(G_j) \oplus T_1$, protected by $h(G_j)$ (where G_j is the secret key of GW_j); without knowing $h(G_j)$ $\mathcal{A}$ will not be able to compute V_i.

5.2d *User Un-traceability*: User un-traceability means an attacker cannot distinguish different login sessions of a user. In the proposed scheme, all the parameters of login message $\{L_2, B_2, B_3, T_1\}$ include the time stamp T_1 in their computation. Because of usage of the time stamp in the calculation, all the parameters are different in different sessions whether the user communicates with the same gateway or not. Hence, the attacker $\mathcal{A}$ cannot keep track of U_i.

5.2e *User impersonation attack*: To impersonate U_i, $\mathcal{A}$ must frame the message $\{D_2, B_2, B_3, T_1\}$ to pass the authentication steps by GW_j. To compute $B_2 = h(T_1||R_i||B_1)$ and $\mathrm{B}_3 = R_i \oplus B_1$ (where $B_1 = h(h(V_i||G_j)||T_1)$, V_i is a random number fixed for user U_i, and G_j is GW_j's secret key). To gain knowledge of B_1, $\mathcal{A}$ must compute V_i from $D_2 = V_i \oplus h(G_j)$, and the attacker $\mathcal{A}$ must know the secret key G_j that is known only to GW_j. Hence, it is impossible to impersonate as U_i.

5.2f *Replay attack*: According to our attack model, $\mathcal{A}$ can monitor, eavesdrop or alter the messages communicated over public channel. Hence, U_i's old login message $\{L_2, B_2, B_3, T_1\}$ can be sent as a fresh message by $\mathcal{A}$ to GW_j, but GW_j will not authenticate because it fails the freshness test of T_1. Suppose that $\mathcal{A}$ changes the time stamp T_1 with the new timestamp T_e to pass the freshness test; this attempt will also fail to authenticate because T_e is used to compute $B_1^* = h(K_i||T_e)$ and $B_2^* = h(T_e||R_i||B_1^*)$ where $K_i = h(V_i||G_j)$, while communicated $B_2 = h(T_1||R_i||B_1)$

where $B_1 = h(K_i||T_1)$ is computed using T_1; B_2^* and B_2 will be different. For the same reason, attacker $\mathcal{A}$ could not replay the messages $\{f_7, f_8, T_4\}$ $\{C_1, C_2, T_3\}$ and $\{f_4, f_5, f_6, T_2\}$ communicated among SN_k, GW_j, and U_i.

5.2g *PG attack*: Assuming that $\mathcal{A}$ can extract $\{D_2, HB_i, f_3, A_i, h(.)\}$ from the SC of U_i, where $D_2 = V_i \oplus h(G_j)$, $f_3 = K_i \oplus h(C_i||P_i)$, $K_i = h(V_i||G_j)$, $P_i = h(PW_i||B_i)$, $C_i = h(ID_i||B_i)$ and $HB_i = h(P_i||C_i||B_i)$. Attacker $\mathcal{A}$ can guess low entropy user identity (ID_i^*) and password (PW_i^*), but to check the correctness of ID_i^* and PW_i^*, $\mathcal{A}$ must compute P_i and C_i; to do this $\mathcal{A}$ needs B_i, where B_i is generated by a fuzzy extractor using biometric information of the user. Hence, it is hard to guess B_i and without B_i, $\mathcal{A}$ cannot verify the correctness of guessed UID_i and PW_i. Hence, the attacker cannot guess U_i's UID_i and PW_i from the SC. User identity is not communicated with the GW_j or SN_k, so that computing U_i's identity from intercepted messages is not possible.

5.2h Sensor node anonymity: According to our attack model, $\mathcal{A}$ can eavesdrop communicated messages between GW_j and SN_k, i.e. $\{f_4, f_5, f_6, T_2\}$ and $\{C_1, C_2, T_3\}$ where $f_4 = h(SID_k||M_1||SIX_k||R_i||T_2)$, $f_5 = (R_i||R_j) \oplus h(SIX_k||T_2)$, $f_6 = M_1 \oplus h(SID_k||SIX_k||R_i)$, $C_1 = h(SK_i||R_k||SID_k||SIX_k||M_1||T_3)$ and $C_2 = R_k \oplus h(SIX_k||R_j)$. $\mathcal{A}$ cannot guess or compute SN_k's identity (SID_k), as SID_k is secured with the hash function.

5.2i *SNC attack*: According to our attack model, $\mathcal{A}$ can physically capture SN_k, and retrieve the information stored at the sensor node, i.e. $\{SID_k, SIX_k\}$ where $SIX_k = h(SID_k||X_c)$. $\mathcal{A}$ can compute R_i, R_j and R_k as well as SK_i for the current session with SN_k and send wrong data to U_i, but attacker $\mathcal{A}$ cannot compute any secret parameter of U_i or GW_j, as $\mathcal{A}$ can use SIX_k and time stamp T_2 to extract $(R_i||R_j) = f_5 \oplus h(SIX_k||T_2)$ and $M_1 = f_6 \oplus h(SID_k||SIX_k||R_i)$ where $M_1 = h(V_i||R_j)$. $\mathcal{A}$ cannot compute V_i from M_1 as V_i is secured with the hash function, and R_j is a session-specific random number, so M_1 cannot be used for other sessions.

Attacker $\mathcal{A}$ can use R_i to compute B_1 from B_3, i.e. $B_1 = B_3 \oplus R_i$ where $B_1 = h(K_i||T_1)$ and $K_i = h(V_i||G_j)$. Because T_1 is included in the computation of B_1, it will be different for the distinct session and cannot be used for different sessions. Thus, $\mathcal{A}$ can compute B_1, M_1, R_i, R_j, R_k and SK_i using SID_k and SIX_k. However, all parameters contain session-specific values and are calculated for the particular session. $\mathcal{A}$ cannot get any secret parameters of U_i, GW_j or any other sensor node and cannot affect other sessions by capturing a sensor node.

5.2j *Sensor node impersonation attack*: $\mathcal{A}$ can impersonate as SN_k if he can compute valid message $\{C_1, C_2, T_3\}$ based on the parameters of the message $\{f_4, f_5, f_6, T_2\}$ transmitted from GW_j to SN_k, where $C_1 = h(SK_i||R_k||SID_k||SIX_k||M_1||T_3)$, $C_2 = R_k \oplus h(SIX_k||R_j)$ and $M_1 = h(V_i||R_j)$ are dependent on secret parameters (SID_k, SIX_k and V_i) and session-specific values (R_i, R_j and R_k), which are impossible to guess in polynomial time and secured by hash function. Therefore, $\mathcal{A}$ cannot impersonate as SN_k.

5.2k *Gateway impersonation attack*: In the proposed scheme, there are two possible cases for $\mathcal{A}$ impersonating as GW_j. Case 1: when GW_j transmits the message $\{f_4, f_5, f_6, T_2\}$ to SN_k, $\mathcal{A}$ can eavesdrop, intercept or alter the message. In this scheme, an attacker $\mathcal{A}$ cannot create another valid message as $f_4 = h(SID_k||M_1||SIX_k||R_i||T_2)$, $f_5 = (R_i||R_j) \oplus h(SIX_k||T_2)$ and $f_6 = M_1 \oplus h(SID_k||SIX_k||R_i)$, calculation of f_4, f_5, and f_6 depends on secret parameters (SID_k, SIX_k and V_i) and session-specific values (R_i, R_j and R_k). Case 2: when GW_j replies to the user with the message $\{f_7, f_8, T_4\}$ where $f_7 = h(SK_i||R_j||R_k||T_4||K_i)$, $f_8 = (R_i||R_k) \oplus h(K_i||R_i)$ and $K_i = h(V_i||G_j)$, to compute a valid message attacker requires the knowledge of secret parameters (V_i and G_j) and session-specific values (R_i, R_j and R_k), which are secured by hash function and impossible to guess with polynomial time complexity. Hence, it is impossible for $\mathcal{A}$ to impersonate as GW_j.

5.3 *Formal security verification using AVISPA tool*

Automated Validation of Internet Security Protocol (AVISPA) [36] is used for simulation and verification of authentication protocols, based on the Dolev and Yao's intruder model [37]. AVISPA has been integrated with GUI called Security Protocol Animator (SPAN) [38].

5.3a *Specification of our scheme*: The specification of our scheme is described briefly for the roles of RC, U_i, GW_j, and SN_k. The role of RC in HLPSL is given in figure 5. In this role, when RC gets a start signal to initiate the registration

```
role regC (Ui, SNk,GWj, RC:agent,
SK1:symmetric_key,
%hashfunction
H: hash_func, Snd, Rcv: channel (dy) )
played_by RC
def=
local State : nat,
IDi,PWi,Bi,Ci,Pi,HBi,F3,B1,T1,T2,T3,T4,
SIXk,Xc,Gj,Ri,Rj,Rk,TGi:text,
GIDj,SIDk,Vi,B2,B3,F4,F5,F6,C1,C2,F7,F8:message,
const u_gw,gw_sn,sn_u,reg_center,sec1,sec2,sec3,sec4,
sec5,sec6,sec7:protocol_id

init State :=0
transition
%Start registration phase of the user
1.State=0/\Rcv(start) =|>
State':=1/\Xc':=new()
/\GIDj':=new()
/\Gj':=h(GIDj'.Xc')
/\SIDk':=new()
/\SIXk':= h(SIDk'.Xc')
/\Snd({GIDj'.Gj'.SIDk'.SIXk'}_SK1)
/\Snd({SIDk'.SIXk'}_SK1)
/\secret({Xc},sec3, RC)
/\secret({Gj},sec4,{RC,GWj} )
/\secret({SIXk},sec5,{RC,SNk,GWj})
end role
```

Figure 5. Role specification of RC.

```
role user (Ui, SNk, GWj ,RC:agent,
SK1:symmetric_key,
%hashfunction
H:hash_func,Snd,Rcv: channel (dy) )
played_by Ui
def=
local State : nat,
IDi,PWi,Bi,Ci,Pi,HBi,D2,F3,L2,B1,T1,T2,
T3,T4,SIXk,Xc,Gj,Ri,Rj,Rk,TGi,SKi,Ki:text,
GIDj,SIDk,Vi,B2,B3,F4,F5,F6,C1,C2,F7,F8:message,
const u_gw,gw_sn,sn_u,reg_center,sec1,sec2,
sec3,sec4,sec5,sec6, sec7:protocol_id
init State:=0
transition
%Start registration phase of the user
1.State=0/\Rcv(start)=|>
State':=1
/\Ci':=h(IDi.Bi)
/\Pi':=h(PWi.Bi)
%Send registration request to the GWj
/\Snd({Ci'.Pi'}_SK1)
/\secret({IDi,PWi,Bi}, sec1, Ui)
2.State=1/\Rcv({D2.F3}_SK1)=|>
%Reeceives smart card information from GWj
State':=2/\HBi':= h(Pi.Ci.Bi)
 %State' : =2
/\Ri':=new()/\T1':=new()
/\L2':=xor(D2,T1')
/\Ki':=xor(F3 ,h(Ci,Pi))
/\B1':=h(Ki'.T1')
/\B2':=h(T1',Ri'.B1')
/\B3':=xor((Ri'),B1')
/\Snd(L2',B2',B3',T1')
%Send login message to the GWj
/\witness(Ui,GWj,u_gw, Ri')
/\secret({Ri'},sec2,{Ui,GWj,SNk})
/\request(Ui,GWj,u_gw, Ri')
3.State=2/\Rcv(F4,F7,xor((Rj.Rk),h(Ki.Ri)),T4 ) =|>
%Receives messages from Gwj
State':=3 /\SKi' := h(Ri.Rj.Rk)
/\F7' := h(SKi'.Rj.Rk.T4.Ki)
end role
```

Figure 6. Role specification of U_i.

of GW_j and SN_k, RC uses $new()$ to generate $GIDj'$ and computes Gj' as secret of GW_j.

To register SN_k, RC generates $SIDk'$ using $new()$ and computes $SIXk'$ as shared secret of GW_j and SN_k. Afterwards, RC sends $\{SIDk', SIXk'\}$ and $\{GIDj', Gj', SIDk', SIXk'\}$ securely to SN_k and GW_j.

The role of U_i is given in figure 6. U_i receives the start signal and sends the registration request (Ci, Pi) to GW_j through private channel. Afterwards, U_i receives $(D2, F3)$ over private channel and stored into the SC.

Afterwards, U_i calculates HBi and uses $new()$ operation to generate Ri' and $T1'$. Then, U_i sends $(L2, B2, B3, T1)$ to GW_j. On receiving $(F4, F7, xor((Rj.Rk), h(Ki.Ri)), T4)$ from GW_j over a public channel, U_i computes $SKi' := h(Ri.Rj.Rk)$.

The role of the gateway is presented in figure 7. In transition 1, GW_j receives $(GIDj, Gj, SIDk, SIXk)$ through secure channel. On receiving registration request from U_i through secure channel, GW_j generates Vi' and computes $F3'$ and $D2'$. Then, it sends $(D2, F3)$ to U_i via a secure channel.

In transition 3, GW_j receives the login request $(L2, B2, B3, T1)$ from the U_i over an unsecure channel and uses $new()$ operation to generate nonce Rj' and time stamp $T2'$ and sends the message $(F4, F5, F6)$ using $Snd()$ operation. In transition 4, after receiving $(C1, xor(Rk, h(SIXk.Rj)), T3)$ from SN_k, it sends $(F7, F8, T4)$ to U_i over an unsecure channel.

```
role gateway (Ui, SNk,GWj , RC: agent,
SK1: symmetric_key ,
%hash function
H: hash_func, Snd, Rcv: channel (dy) )
played_by GWj
def=
local State : nat,
IDi,PWi,Bi,Ci,Pi,HBi,L2,D2,F3,B1,SKi,
T1,T2,T3,T4,SIXk,Xc,Gj,Ri,Rj,Rk,TGi,Ki,M1: text,
GIDj,SIDk,Vi,B2,B3,F4,F5,F6,C1,C2,F7,F8:message,
const u_gw,gw_sn,sn_u,reg_center,sec1,sec2,
sec3,sec4,sec5,sec6,sec7:protocol_id
init State:=0
transition
1.State=0/\Rcv({GIDj.Gj.SIDk.SIXk }_SK1) =|>
State':=1/\secret ({Gj},sec4, {RC, GWj })
2.State=1/\ Rcv ({Ci.Pi}_SK1) =|>
%Receives registration request message from user
State':=2/\Vi':=new()
/\F3':=xor(h(Vi'.Gj),h(Ci.Pi))
/\D2':=xor(Vi',h(Gj))
/\Snd({D2'.F3'}_SK1)
%Send smart card information securely
3.State=2/\Rcv(L2.B2.B3.T1)=|>
%Receives login message from user
State':=3/\Rj':=new()
/\T2':=new()
/\M1':=h(Vi.Rj')
/\F4':=h(SIDk .M1'.SIXk.Ri.T2')
/\F5':=xor((Ri.Rj'),h(SIXk.T2'))
/\F6':=xor(M1',h(SIDk.SIXk.Ri))
/\Snd(F4', F5',F6',T2')
%Send message to the sensor node
/\secret({Rj'},sec6,{Ui,GWj,SNk})
/\request(GWj,SNk,gw_sn,Rj')
4.State=3/\Rcv(C1,xor(Rk,h(SIXk.Rj)),T3)=|>
State':=4/\T4':=new()
/\SKi':=h(Ri.Rj.Rk)
/\F7':=h(SKi.Rj.Rk.T4'.Ki)
/\F8':=xor((Rj.Rk),h(Ki.Ri))
/\Snd(F7',F8',T4')
%Send message to the user
end role
```

Figure 7. Role specification of GW_j.

```
role snode(SNk,Ui,GWj,RC: agent,
SK1:symmetric_key,
%hash function
H:hash_func, Snd, Rcv:channel (dy))
played_by SNk
def=
local State: nat,
IDi,PWi,Bi,Ci,Pi,HBi,F3,B1,T1,T2,T3,T4,
SIXk,Xc,Gj,Ri,Rj,Rk,TGi,SKi,M1:text,
GIDj,SIDk,Vi,B2,B3,F4,F5,F6,C1,C2,F7,F8:message,
const u_gw,gw_sn,sn_u,reg_center,sec1,sec2,
sec3,sec4,sec5,sec6,sec7: protocol_id
init State:=0
transition
%Start node registration phase
1.State=0/\Rcv({SIDk.SIXk}_SK1)=|>
State':=1/\secret({SIXk},sec5,{RC,GWj,SNk})
2.State=1/\Rcv(F4,F5,F6,T2)=|>
%Receives message from GWj
State':=2/\Rk':=new()
/\T3':=new()
/\SKi':= h(Ri.Rj.Rk')
/\C1':= h(SKi'.Rk'.SIDk.SIXk.M1.T3')
/\C2':= xor(Rk',h(SIXk.Rj))
/\Snd(C1,C2,T3')
%Send message to the GWj
/\secret({Rk'},sec7,{Ui,GWj,SNk})
/\witness(SNk,Ui,sn_u,Rk')
/\request(SNk,GWj,gw_sn,Rk')
end role
```

Figure 8. Role specification of SN_k.

The role of SN_k is presented in figure 8. In transition 1, SN_k securely receives $(SIDk, SIXk)$ from the RC. In transition 2, on receiving $(F4, F5, F6, T2)$ from GW_j, SN_k uses

```
role environment()
def=
const ui,snk,gwj,rc:agent,
sk1:symmetric_key,
h:hash_func,
idi,pwi,bi,ci,pi,hbi,d2,f3,b1,t1,t2,t3,
t4,sixk,xc,gj,ri,rj,rk,tgi,gidj,sidk,vi,
l2,b2,b3,f4,f5,f6,c1,c2,f7,f8,ki:text,
u_gw_ri,gw_sn_rj,sn_u_rk,reg_center,
sec1,sec2,sec3,sec4,sec5,sec6,sec7: protocol_id
%Represents Intruder knowledge
intruder_knowledge={ui, snk, gwj ,h, l2, b2, b3,
t1,t2,t3,t4, f4,f5, f6,c1,c2, f7,f8,d2,hbi,f3}
composition
session(ui, gwj, snk, rc, sk1, h)
/\session(ui, gwj, snk , rc, sk1 ,h)
/\session(ui, gwj, snk , rc, sk1 ,h)
/\session(ui, gwj, snk , rc, sk1 ,h)
end role
role session(Ui,SNk,GWj,RC:agent,
SK1:symmetric_key,
H:hash_func)
def=
local SI,SJ,RI,RJ,TI,TJ,PI,PJ:channel (dy)
composition
user(Ui,GWj,SNk,RC,SK1,H,SI,RI)
/\gateway(Ui,GWj,SNk,RC,SK1,H,SJ,RJ)
/\snode(Ui,GWj,SNk,RC,SK1,H,TI,TJ)
end role
goal
%Verifies secrecy of the confidential information
secrecy_of sec1
secrecy_of sec2
secrecy_of sec3
secrecy_of sec4
secrecy_of sec5
secrecy_of sec6
secrecy_of sec7
%Verifies authenticity of the random numbers
authentication_on u_gw_ri
authentication_on gw_sn_rj
authentication_on sn_u_rk
end goal
```

Figure 9. Role specification of environment, session and goal.

$new()$ to generate nonce Rk' and time stamp $T3'$ and sends $(C1, C2, T3)$ to the GW_j over a public channel.

The roles of session, environment and goal are given in figure 9. Session role initializes U_i, GW_j and SN_k roles with actual parameters. The environment role contains global constants, information about intruder knowledge and the combination of one or more sessions.

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/newscheme.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.16s
  visitedNodes: 16 nodes
  depth: 4 plies
```

Figure 10. Simulation results for OFMC back-end.

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL
PROTOCOL
  /home/span/span/testsuite/results/newscheme.if
GOAL
  As Specified
BACKEND
  CL-AtSe
STATISTICS
  Analysed  : 0 states
  Reachable  : 0 states
  Translation: 0.18 seconds
  Computation: 0.00 seconds
```

Figure 11. Simulation results for CL-Atse back-end.

5.3b *Simulation result*: The AVISPA simulation results of the proposed scheme are given in this section. The HLPSL code of the proposed scheme has been simulated with SPAN to examine results. Figures 10 and 11 show simulation results for OFMC and CL-Atse back-end. Results show that the proposed scheme is safe.

6. Performance analysis

A comparison of our scheme to the related schemes regarding security features and computational cost are presented in this section. For comparison of computational cost, we included only the login and authentication phases for comparison as they are executed very frequently than registration and password change phase.

6.1 *Security features*

A comparison of the security features of our scheme with other relevant schemes is shown in table 2. Table 2 clearly shows that the proposed scheme covers all security features, while other schemes lack some of the security features.

6.2 *Computational cost*

The computational costs of schemes are given in table 3. Following notations are used in this table: T_E means time complexity of elliptic curve point multiplication, T_H means time complexity of hash operation, T_f means time complexity of fuzzy extractor operation and T_S means time complexity of symmetric encryption/decryption.

When we compare other schemes with our scheme, the proposed scheme has less computational cost than schemes given in [24, 22, 27, 31, 34]. The schemes are given in [1], and [39] has less computational cost in comparison with our scheme, but their schemes have some major security flaws (see table 2).

Table 2. Security features comparison.

Security features	[39]	[34]	[27]	[24]	[22]	[31]	[1]	Proposed
Provide UA	✔	✗	✗	✗	✗	✔	✔	✔
Provide SA	✗	✔	✗	✔	✗	✗	✔	✔
Provide UT	✔	✗	✔	✗	✗	✗	✔	✔
Resilient to UI attack	✗	✗	✔	✔	✔	✔	✗	✔
Resilient to SNI attack	✔	✔	✔	✔	✔	✔	✔	✔
Resilient to PG attack	✗	✗	✗	✔	✔	✔	✔	✔
Provide MA	✔	✔	✔	✔	✔	✔	✔	✔
Resilient to SSK attack	✔	✔	✔	✔	✔	✔	✔	✔
Resilient to SEE attack	✔	✔	✔	✔	✔	✔	✔	✔
Provide FS	✔	✔	✗	✗	✗	✔	✔	✔
Resilient to Replay attack	✔	✔	✔	✔	✔	✔	✔	✔
Resilient to SV attack	✗	✗	✗	✔	✔	✗	✔	✔
Resilient to SSC attack	✗	✗	✗	✗	✗	✗	✗	✔
Resilient to PI attack	✗	✗	✗	✔	✔	✔	✔	✔
Resilient to DoS attack	✗	✗	✗	✗	✗	✗	✔	✔
Resilient to SNC attack	✔	✔	✔	✗	✔	✔	✗	✔

Table 3. Computational cost.

Schemes	U_i	GW_j	SN_k	Total
Gope and Hwang [39]	$7T_H$	$9T_H$	$3T_H$	$19T_H$
Das [34]	$6T_H + 1T_S + 1T_f$	$6T_H + 2T_S$	$2T_H + 1T_S$	$14T_H + 4T_S + 1T_f$
Xue *et al* [27]	$10T_H$	$14T_H$	$6T_H$	$30T_H$
He *et al*[24]	$4T_H + 3T_S$	$2T_H + 5T_S$	$1T_H + 2T_S$	$7T_H + 10T_S$
Choi *et al* [22]	$12T_H + 3T_E$	$5T_H + 1T_E$	$7T_H + 2T_E$	$24T_H + 6T_E$
Kumari and Om [31]	$10T_H$	$8T_H$	$6T_H$	$24T_H$
Kumar *et al* [1]	$8T_H$	$8T_H$	$4T_H$	$20T_H$
Proposed	$9T_H$	$13T_H$	$6T_H$	$28T_H$

Table 4. Estimated execution time (in ms).

Entities	[39]	[34]	[27]	[24]	[22]	[31]	[1]	Proposed
U_i	3.5	74.78	5	28.1	195.24	5	4	4.5
GW_j	4.5	20.4	7	44.5	65.58	4	4	6.5
SN_k	1.5	9.7	3	17.9	129.66	3	2	3
Total	9.5	104.88	15	90.5	390.48	12	10	14

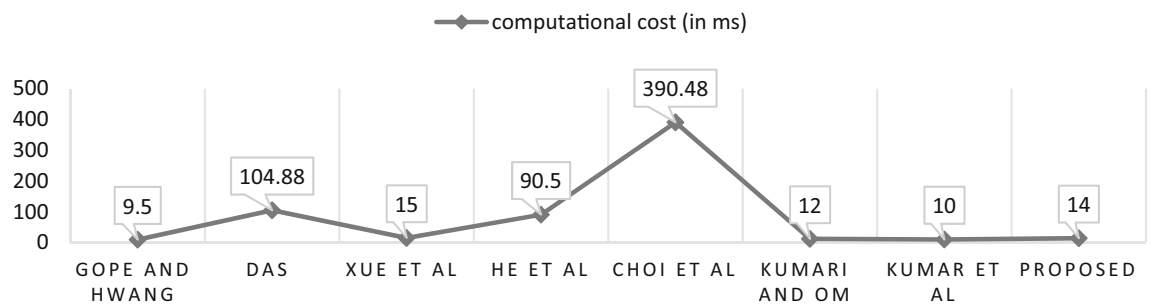

Figure 12. Estimated execution time.

Different cryptographic operations take different amounts of time to execute; we used the approximate time given in [31, 40] for cryptographic operations to estimate the execution time. As $T_H \approx 0.5$ ms, $T_E \approx 63.08$ ms, $T_S \approx 8.70$ ms and $T_f \approx T_E$. Table 4 and figure 12 present comparison of the estimated execution time of the proposed scheme with other related schemes.

7. Conclusion

Coal mining is one of the most dangerous industries. Continuous monitoring of the environmental parameters is needed to ensure safe coal production. Therefore, WSN technology can be used to monitor environmental parameters in coal mines. However, security is a crucial issue for WSN-based safety monitoring.

Recently, Kumar *et al* designed an authentication and key agreement scheme for WSN-based safety monitoring in coal mines. After a thorough analysis of Kumar *et al* scheme, we demonstrated that their scheme is unsafe against SNC attack, SSC attack and user impersonation attack. As a part of our contribution, we provided a biometric-based secure authentication framework for safety monitoring system in coal mines. We simulated the scheme on AVISPA tool. We formally analysed the security using the ROM and also performed informal security analysis. These analyses demonstrate that our scheme is secure and invulnerable to various known attacks. The proposed scheme is lightweight in terms of computations and requires only one two-way communication with sensors to achieve MA and key agreement as well as provides security from all the known attacks. Therefore, it is well suited for the coal mine environment; however, we can also apply it to other applications of WSN.

References

[1] Kumar D, Chand S and Kumar B 2018 Cryptanalysis and improvement of an authentication protocol for wireless sensor networks applications like safety monitoring in coal mines. *J. Ambient Intell. Humaniz. Comput.* 1–20

[2] Zhao G 2011 Wireless sensor networks for industrial process monitoring and control: a survey. *Netw. Protoc. Algorithms* 3: 46–63

[3] Guha A and Kumar K V 2012 Structural controls on coal fire distributions – Remote sensing based investigation in the Raniganj coalfield, West Bengal. *J. Geol. Soc. India* 79: 467–475

[4] Wang J, Liu T, Song G, Xie H, Li L, Deng X and Gong Z 2014 Fiber Bragg grating (FBG) sensors used in coal mines. *Photon. Sens.* 4: 120–124

[5] Kumar A, Kingson T M G, Verma R, Mandal R, Dutta S, Chaulya S K and Prasad G 2013 Application of gas monitoring sensors in underground coal mines and hazardous areas. *Int. J. Comput. Technol. Electron. Eng.* 3: 9–23

[6] Bhattacharjee S, Roy P, Ghosh S, Misra S, and Obaidat M S 2012 Wireless sensor network-based fire detection, alarming, monitoring and prevention system for Bord-and-Pillar coal mines. *J. Syst. Softw.* 571–581

[7] Chehri A, Farjow W, Mouftah H T and Fernando X 2011 Design of wireless sensor network for mine safety monitoring. In: *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, pp. 001532–001535

[8] Chehri A, Fortier P, and Tardif P M 2009 UWB-based sensor networks for localization in mining environments. *Ad Hoc Networks* 7: 987–1000

[9] Mishra P K, Kumar S, Kumar M, and Kumar J 2019 IoT based multimode sensing platform for underground coal mines. *Wirel. Pers. Commun.* 108: 1227–1242

[10] Moridi M A, Kawamura Y, Sharifzadeh M, Chanda E K, Wagner M, Jang H and Okawa H 2015 Development of underground mine monitoring and communication system integrated ZigBee and GIS. *Int. J. Min. Sci. Technol.* 25: 811–818

[11] Al-Karaki J N and Kamal A E 2004 Routing techniques in wireless sensor networks: a survey. *IEEE Wirel. Commun.* 11: 6–28

[12] Wong K H M, Zheng Y, Cao J, and Wang S 2006 A dynamic user authentication scheme for wireless sensor networks. In: *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, vol. 1, (SUTC'06)*, 1: 244–251

[13] Das M L 2009 Two-factor user authentication in wireless sensor networks. *IEEE Trans. Wirel. Commun.* 8: 1086–1090

[14] Tseng H R, Jan R H and Yang W 2007 An improved dynamic user authentication scheme for wireless sensor networks. In: *Proceedings of IEEE GLOBECOM 2007—2007 IEEE Global Telecommunications Conference*, pp. 986–990

[15] Huang H F, Chang Y F and Liu C H 2010 Enhancement of two-factor user authentication in wireless sensor networks. In: *2010 Proceedings of the Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp. 27–30

[16] He D, Gao Y, Chan S, Chen C and Bu J 2010 An enhanced two-factor user authentication scheme in wireless sensor networks. *Ad Hoc Sens. Wirel. Netw.* 10: 361–371

[17] Ko L C 2008 A novel dynamic user authentication scheme for wireless sensor networks. In: *Proceedings of the 2008 IEEE International Symposium on Wireless Communication Systems*, pp. 608–612

[18] Vaidya B, Rodrigues J J and Park J H 2010 User authentication schemes with pseudonymity for ubiquitous sensor network in NGN. *Int. J. Commun. Syst.* 23: 1201–1222

[19] Kim J, Lee D, Jeon W, Lee Y, and Won D 2014 Security analysis and improvements of two-factor mutual authentication with key agreement in wireless sensor networks. *Sensors* 14: 6443–6462

[20] Yeh H L, Chen T H, Liu P C, Kim T H and Wei H W 2011 A secured authentication protocol for wireless sensor networks using elliptic curves cryptography. *Sensors* 11: 4767–4779

[21] Shi W and Gong P 2013 A new user authentication protocol for wireless sensor networks using elliptic curves cryptography. *Int. J. Distrib. Sens. Netw.* 9: 730831

[22] Choi Y, Lee D, Kim J, Jung J, Nam J and Won D 2014 Security enhanced user authentication protocol for wireless sensor networks using elliptic curves cryptography. *Sensors* 14: 10081–10106

[23] Kumar P, Lee S-G, and Lee H-J 2012 E-SAP: efficient-strong authentication protocol for healthcare applications using wireless medical sensor networks. *Sensors* 12: 1625–1647

[24] He D, Kumar N, Chen J, Lee C C, Chilamkurti N and Yeo S S 2015 Robust anonymous authentication protocol for health-care applications using wireless medical sensor networks. *Multimed. Syst.* 21: 49–60

[25] Nam J, Choo K K R, Han S, Kim M, Paik J, and Won D 2015 Efficient and anonymous two-factor user authentication in wireless sensor networks: achieving user anonymity with lightweight sensor computation. *PLoS One* 10: e0116709

[26] Li X, Niu J, Kumari S, Liao J, Liang W and Khan M K 2016 A new authentication protocol for healthcare applications using wireless medical sensor networks with user anonymity. *Secur. Commun. Netw.* 9: 2643–2655

[27] Xue K, Ma C, Hong P and Ding R 2013 A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. *J. Netw. Comput. Appl.* 36: 316–323

[28] Li C T, Weng C Y, and Lee C C 2013 An advanced temporal credential-based security scheme with mutual authentication and key agreement for wireless sensor networks. *Sensors* 13: 9589–9603

[29] Turkanović M, Brumen B and Hölbl M 2014 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

[30] Chang C C and Le H D 2016 A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks. *IEEE Trans. Wirel. Commun.* 15: 357–366

[31] Kumari S and Om H 2016 Authentication protocol for wireless sensor networks applications like safety monitoring in coal mines. *Comput. Netw.* 104: 137–154

[32] Kocher P, Jaffe J and Jun B 1999 *Differential power analysis*. Berlin–Heidelberg: Springer, pp. 388–397

[33] Messerges T S, Dabbish E A and Sloan R H 2002 Examining smart-card security under the threat of power analysis attacks. *IEEE Trans. Comput.* 51: 541–552

[34] Das A K 2017 A secure and effective biometric-based user authentication scheme for wireless sensor networks using smart card and fuzzy extractor. *Int. J. Commun. Syst.* 30: e2933

[35] Chandrakar P and Om H 2017 A secure and robust anonymous three-factor remote user authentication scheme for multi-server environment using ECC. *Comput. Commun.* 110: 26–34

[36] *Automated validation of internet security protocols and applications* [Online]. Available: http://www.avispa-project.org

[37] Dolev D and Yao A C 1983 On the security of public key protocols. *IEEE Trans. Inf. Theory* 29: 198–208

[38] Glouche Y, Genet T, Heen O and Courtay O 2006 A security protocol animator for AVISPA. In: *Proceedings of the ARTIST2 Workshop on Security Specification and Verification of Embedded Systems, Pisa*

[39] Gope P and Hwang T 2016 A realistic lightweight anonymous authentication protocol for securing real-time application data access in wireless sensor networks. *IEEE Trans. Ind. Electron.* 63: 7124–7132

[40] Kilinc H H and Yanik T 2013 A survey of SIP authentication and key agreement schemes. *IEEE Commun. Surv. Tutorials* 16: 1005–1023