Unlinkable User Authenticated Key Agreement for Multi-Gateway Wireless Sensor Networks

Solomon Kuonga Nkhoma¹, Patrick Ali¹, Levis Eneya¹, and Hyunsung Kim¹,²,*

¹Department of Mathematical Sciences, Chancellor College, University of Malawi P.O.Box 280, Zomba, Malawi
²Department of Cyber Security, Kyungil University Kyungsan, Kyungbuk 38428, Korea

Background

Wireless sensor network (WSN) has wide potential application in various fields such as military, agricultural and healthcare. WSNs need effective security mechanisms because they are deployed in hostile unattended environments. Various user authentication schemes were proposed for WSNs security. However, there are many previous schemes that have various security vulnerabilities including masquerading, password guessing attack and traceability. This paper proposes an unlinkable user authenticated key agreement scheme (UAKA) for multi-gateway WSNs that could achieve desirable security and privacy attributes. It preserves all the original merits of the related schemes and the security of UAKA is analyzed using the BAN logic. Furthermore, the performance of UAKA is comparable to the related existing schemes.

Publication History: Received: 21 September 2018 | Revised: 26 October 2018 | Accepted: 31 October 2018

Keywords: Wireless sensor network, Information security, Multi-gateway, Unlinkability, Anonymity, Authentication, Key agreement.

1. INTRODUCTION

Wireless sensor networks (WSNs) consisted of many small devices each with sensing, processing, and communication capabilities to monitor the targeting environment. There are numerous fields of application of WSNs like environmental monitoring, agriculture, military, health care and so on. WSN is basically consist of numerous sensor nodes with the wireless channel to connect the nodes. Number of nodes in any network varies from hundreds to thousands which makes it different from other wireless networks [1-3]. Since, most of the time, deploying a WSN in a hostile environment is done by a random distribution, it is difficult to know the topology of WSNs a priori. Generally, there are three entities in WSN, sensor, a gateway node (GWN), and user. Sensors collect data from the deployed environment and send it to GWN. GWNs are responsible for conveying the sensed data to users and the request data to sensors. Registered users can access data obtained by the sensors after registering to GWN [4]. Such users need to be authorized and, if done positively, allowed to gather data from or send commands to the sensor node. Since the most important characteristic of WSNs is their resource constrained feature, a lightweight security solution is required. A key challenge is how to establish a shared cryptographic key in a secure manner between the sensor node and the user.

Numerous user authentication and key agreement schemes for enhancing the security of WSNs have been proposed [5-14]. Wong et al. proposed a user authentication scheme for WSNs, which requires only the computation of hash functions [5]. However, it was proved to be vulnerable to stolen verifier attack and many logged-in users with the same login identity (ID) attack. Das proposed an efficient password-based user authentication to solve the weaknesses in Wong et al.’s scheme, which uses the temporal credentials for verification [6]. Das’s scheme is also shown to be vulnerable to denial-of-service attack and node capture attack. Khan and Alghathbar presented an improvement of Das’s scheme [7]. Vaidya et al. identified the security pitfalls in Khan and Alghathbar’s scheme [8]. To overcome these security pitfalls, Vaidya et al. proposed an improved version of Khan and Alghathbar’s scheme [9]. Independently, Das et al. proposed authentication and key agreement schemes for WSNs using smart cards, which supports a user to viably and securely connect to the nodes of a WSN [10]. Turkanovic et al. proposed a user authentication and key agreement model to overcome the security flaws of the earlier designed schemes [11]. Farash et al. shown that Turkanovic et al.’s scheme is insecure and inefficient for various security drawbacks such as
a session key agreement, mutual authentication between all parties, traceability, preservation of user anonymity, privileged insider attack and password guessing attack [12]. To overcome these shortcomings, Amin and Biswas presented a secure lightweight scheme for user authentication and key agreement in multi-gateway based WSNs [13]. However, Srinivas et al. showed that Amin and Biswas’s scheme has leakages of sensors secret keys and the system key, and is weak against server spoofing attack, user impersonation attack, stolen smart card attack, off-line password guessing attack and identity guessing attack [14].

The purpose of this paper is to design an unlinkable user authenticated key agreement scheme (UAKA) for multi-gateway WSNs that could achieve desirable security and privacy attributes and can also be applicable for practical applications. The contributions are listed below:

- To overcome the security weaknesses of the previous schemes, we have proposed an efficient and more secure WSNs authentication scheme that can preserve all the original merits of the related schemes and withstands the possible known attacks.

- To strengthen UAKA, the security analysis using the BAN logic has been presented. Using the informal security analysis, we have also shown that UAKA can resist numerous security attacks which include the attacks found in the previous schemes.

- Furthermore, the performance of UAKA is comparable to the related existing schemes.

2. PRELIMINARIES

This section provides preliminaries for the targeting WSN environment and threat model. They are important to understand UAKA.

2.1. WSN Model

Fig. (I) shows our proposed target WSN model. The proposed model consists of three types of entities, sensor nodes, GWNs and users. Their roles are defined as follows:

- Sensor nodes: They are responsible for sensing the real-time data and forward them to the nearest GWN node directly.

- GWNs: They are responsible for receiving and forwarding the relevant data to the user and sensor node. Furthermore, they keep a database of sensor nodes to be related among GWNs.

- Users: They can access the sensed data of the sensor node through GWN after performing mutual authentication and key agreement.

GWN and sensor nodes are stationary after deployment, which is shown in Fig. (I). As mentioned in [15-17], the receiver end can measure the distance based on the received signal strength. Therefore, it is our valid assumption that all the deployed sensor nodes execute registration phase to the nearest GWN. In order to access the desired sensor node, the user can execute registration phase to any one of GWNs of our WSN model. While a user completes the registration procedure to any one of GWNs, called as home GWN (HGWN) and rest of the others are foreign GWNs (FGWN) with respect to that user. It is our effortful contribution that the user can access all GWNs of WSNs, although he (or she) has performed registration to only one HGWN. There are two scenarios in Fig. (I), which are for users A and B. The first case is for the situation when he (or she) wants to access the data from SN1. However, if the GWN could not find the target sensor node in its own database, it forwards the

![Multi GWN based WSN model.](image-url)
3. UNLINKABLE USER AUTHENTICATED KEY AGREEMENT FOR MULTI-GATEWAY WSNs

This section proposes an unlinkable user authenticated key agreement for multi-gateway WSNs, which is denoted as UAKA. UAKA uses a similar concept with Srinivas et al.’s scheme in [14] but provides further security and privacy aspects. UAKA has six phases, system setup phase, registration phase, system environment phase, login authentication and key.

3.1. System Setup Phase

It is an off-line mode, where a system administrator (SA) generates the identity and security parameters for each sensor node (SNs). First, SA generates the identities \{ID_{SN1}, ID_{SN2}, \ldots, ID_{SNn}\} for each SN \{SN1, SN2, \ldots, SNn\} such that no two distinct SNs will get same identity. Then, SA computes \(P_j = h(ID_{SNj} \oplus S_{\text{sec}})\) for \(1 \leq j \leq m\), where \(S_{\text{sec}}\) is a random secret known to all GWNs. SA stores \(<ID_{SNj}, P_j>\) for \(1 \leq j \leq m\) into the memory of SNs before their deployment.

3.2. Registration Phase

It is divided into two sub-phases, SN registration and user registration. SN should be registered to its HGWN only once right after the deployment for the security reason. Also, for a user to get the services from any SN, any user needs to be registered. Once the user gets registered, he (or she) will be able to connect with the opted SN. Both users and SNs undergo this registration process, respectively.

3.2.1. Sensor Node Registration Phase

Soon after the deployment, SN has to be one of GWNs realm by applying this phase. This is done through an open channel. SN registration phase is outlined in Fig. (2) and the detailed phase is as follows

SR1. SN generates a random number \(r_{\text{UD}}\) and computes \(UD_1 = h(P_j || ID_{SNj} || T_i || r_{\text{UD}})\) and \(UD_2 = h(P_j || ID_{SNj} || T_i || r_{\text{UD}})\), where \(T_i\) is the time stamp of SNj. SNj sends a registration request message \(M_l = <ID_{SNj}, UD_1, UD_2, T_i>\) to the nearest HGWN through an open channel.

SR2. HGWN checks \(|T_2 - T_1| \leq \Delta T\), where \(\Delta T\) is the predefined permitted transmission delay. Only if it holds, HGWN computes \(P_j’ = h(ID_{SNj} \oplus S_{\text{sec}})\) and \(r’_{\text{UD}} = UD_1 \oplus h(P_j || ID_{SNj} || T_i)\). After that HGWN verifies if \(UD_2\) equals to \(h(P_j’ || ID_{SNj} || T_i || r’_{\text{UD}})\). HGWN terminates the session if the verification fails.
Whenever a registered user, HGWN, generates a random number \( \# \) and computes \( TID = h(ID_s || \#) \) and \( RPW^r = h(PW^r || \#) \), the message \( M_1 = <TID, RPW^r> \) is sent to HGWN. HGWN generates a random number \( n_0 \) and computes \( D_ID = E_{HGWN}(TID) || n_0 \). Then HGWN issues a smart card \( SC \) to \( U_i \). 

\[
SC = <Y_s, DID_s, h(c_s), ID_{SN_{ij}}>
\]

Otherwise, HGWN stores \( ID_{SN_j} \) in its database and computes \( UD_3 = h(ID_{SN_j} || Y_{HGWN}) \), \( UD_4 = h(PW^r || ID_{SN_j} || ID^4) \), \( UD_5 = h(PW^r || ID_{SN_j} || T_2) \), and the message \( M_1 = <TID, RPW^r> \) is sent to HGWN via a secure channel.

SR3. Upon receiving the message, \( SN_j \) checks if \( |T_3 - T_2| \leq \Delta T \). Only if it satisfies, \( SN_j \) computes \( UD_3^j' = UD_3 \oplus h(PW^r || ID_{SN_j} || r_{TD^j}) \) and verifies if \( UD_3^j' \) equals to \( h(PW^r || ID_{SN_j} || UD_5) \). Only if the verification holds, \( SN_j \) changes its identity to \( ID_{SN_j}^j \) and sends it to \( U_i \).

This phase has two major functions, to update the secret parameter \( S_\text{sec} \) in \( P \), so that every HGWN knows which \( SN_j \) is in its region.

### 3.2.2. User Registration Phase

User completes his (or her) registration at HGWN, and achieves personalized security parameters to access \( SN \). The registration phase is discussed in Fig. (3) and the detailed description is as follows:

**UR1.** \( U_i \) selects his (or her) identity \( ID_s \), password \( PW^r \), and a random number \( \# \). \( U_i \) computes \( TID = h(ID_s || \#) \) and \( RPW^r = h(PW^r || \#) \), and then submits a registration message \( M_1 = <TID, RPW^r> \) to HGWN.

**UR2.** On receiving the registration request, HGWN generates a random number \( n_0 \) and computes \( D_ID = E_{HGWN}(TID) || n_0 \) and \( K_r = h(D_ID || X_{HGWN}) \). HGWN issues a smart card \( SC \) for \( U_i \) such that \( SC = <Y_s, DID_s, h(c_s), ID_{SN_{ij}}> \) and sends it to \( U_i \).

**UR3.** Upon receiving \( SC \), \( U_i \) computes \( C_s = h(ID_s || PW^r) \) and \( V = h(ID_s || PW^r || c_s) \). \( U_i \) stores \( C_s \) and \( V \) into \( SC \).

### 3.3. System Environment Phase

HGWN maintains the public directory of all the SNs. So, whenever a registered user, \( U_i \), wants to get services from an SN, \( SN_j \), \( U_i \) can pick the appropriate \( SN_j \)'s identity in the service environment of WSNs. In order to access the services, \( U_i \) first initiates the login session using his (or her) smart card. The authenticity of \( U_i \) is verified in the smart card authentication. Once the legitimacy of \( U_i \) is verified, a login message is forwarded to the HGWN which includes \( SN_j \)'s identity, \( ID_{SN_j} \), where the existence of \( ID_{SN_j} \) is checked in its database.

#### 3.4. Login and Authenticated Key Agreement Phase

HGWN maintains the public directory, which comprises the SN's identities in WSN. This enables any user to select a SN as per his (or her) requirement. To get services from \( SN_j \), \( U_i \) extracts \( ID_{SN_j} \) from the public directory of HGWN. The registered \( U_i \) inserts his (or her) smart card into a card reader to initiate the login and authenticated key agreement phase. This phase can be seen as two phases. Once the legitimacy of \( U_i \) is verified, a login message is forwarded to the proper HGWN through the public channel in order to login to the desired \( SN_j \). The procedure of the login and authenticated key agreement phase is described in the following sub-sections.

#### 3.4.1. Login Phase

First, the smart card needs to verify the legitimacy of \( U_i \). For the valid \( U_i \)'s smart card processes the login request. \( U_i \) executes the login request as follows:

**LG1.** To start the login process, \( U_i \) inserts \( SC \) into the terminal and inputs his (or her) identity \( ID' \) and \( PW^r \). Then \( SC \) computes \( h' = C_s \oplus h(ID' || PW^r) \) and checks if \( V \) equals to \( h(ID' || PW^r || c_s) \). \( SC \) inquires for sensor identity as per requirement to HGWN upon the successful verification, by observing the user requirement and sensors availability, HGWN sends the available sensor's identity \( ID_{SN_j} \) to \( U_i \). Only if the verification holds, \( SC \) generates a random number \( r_0 \) and computes \( K'_r = Y_s \oplus h(PW^r || ID_{SN_j} || ID' || TID) \) and \( D_1 = h(K'_r || ID_{SN_j} || TID) \). After that, \( U_i \) sends a login message \( M_1 = <ID_{SN_j}, D_1, D_2, T_1> \) to HGWN.
3.4.2. Authenticated Key Agreement Phase

After receiving the login message from $U_i$, $HGWN$ checks whether the requested $SN_j$ is in the registered sensor list or not by checking its database. Only if $SN_j$ is in its database, $HGWN$ executes the authenticated key agreement phase. Otherwise, it forwards the message to the appropriate GWN. The message exchange of login and authenticated key agreement is discussed in Fig. (4) and the details of this phase are as follows:

AK1. On receiving the login message $<DID_a, ID_{SN_j}, D_1, D_2, T_i>$ at $T_3$, $HGWN$ checks the freshness of the message as $|T_2 - T_1| \leq \Delta T$. Only if the verification passes, $HGWN$ computes $TID_i || m = D_{SN_j}^{\bot}(DID_i)$, computes $K'' = h(TID_i || X_{HGWN})$ and retrieves $r' = D_3 \oplus h(K'' || |DID_i| |ID_{SN_j}|)$ from its data base. $HGWN$ checks $|T_1 - T_2| \leq \Delta T$ and computes $K' = Y \oplus h(PW') || d'$. $D_2 = h(K' || |r_i| |T_1| |DID_i| |ID_{SN_j}| |TID_i|)$ and $M_1 = <DID_a, lD_{SN_j}, D_3, D_5, T_i>$.

Checks if $|T_3 - T_2| \leq \Delta T$

Computes $K'' = h(DID_3 || X_{HGWN})$

$TID_3 || m' = D_{SN_j}^{\bot}(DID_3)$

$r'' = D_3 \oplus h(K'' || |DID_i| |ID_{SN_j}|)$

Verifies $D_3 = h(DID_3 || P_i || r'' || |T_2|)$

Generates a random number $r_i$

Computes $D_4 = h(P_i || t_i || T_3) \oplus r_i$

$D_5 = h(DID_3 || P_i || |T_3| || r_i)$

$M_2 = <DID_3, D_4, D_5, T_2>$.

Fig. (4). Login phase and authenticated key agreement phase.
the verification holds, $HGWN$ authenticates $U$. Otherwise, the connection is terminated. $HGWN$ generates a random number $r_i$ and computes $i'=h(ID_{SNi} \oplus U_{GWN})$, $D_i=h(PW_i || ID_{SNi} || T_i) \oplus r_i$ and $D_{i'}=h(DID_i || PW_i || r_i' || |ID_{SNi}||T_i)$. $HGWN$ contracts the login message $M_2=<DID_i, D_{i'}, D_{i''}, T_2>$ and sends it to $SN_i$.

AK2. On receiving the message at $T_3$, $SN_i$ checks the freshness of the message as $|T_3 - T_2| \leq \Delta T$. Only if the verification is valid, $SN_i$ extracts $r_i'=D_i \oplus h(PW_i || ID_{SNi} || T_2)$ and $r''_i=D_{i'} \oplus h(PW_i || r'_i || |ID_{SNi}||T_2)$. Then, $SN_i$ verifies if $D_{i''}$ equals to $h(DID_i || PW_i || r''_i || |ID_{SNi}||T_2)$. If the verification does not hold, the connection is aborted. Otherwise, $SN_i$ generates a random number $r_i$ and computes $D_5=h(PW_i || r'_i || T_3) \oplus r''_i$ and $D_6=h(PW_i || r''_i || r_i'')$. Then, $SN_i$ sends the message $M_3=<D_5, D_6, T_3>$ to $HGWN$.

AK3. On receiving the message at $T_4$, $HGWN$ checks $|T_4 - T_3| \leq \Delta T$. Only if the verification is valid, $HGWN$ computes $r'_i'=D_6 \oplus h(PW_i || |ID_{SNi}||T_3)$ and verifies if $D_5$ equals to $h(PW_i || r'_i || |ID_{SNi}||T_3)$. If the verification does not hold, the connection is aborted. Otherwise, $HGWN$ computes $D_8=h(K' || |ID_{SNi}||T_4) \oplus r''_i$ and $D_9=h(K' || |ID_{SNi}||T_4) \oplus r'_i \oplus D_{10}=E_{\text{GWN}}(TID_i || r_i \oplus D_{11}=h(K' || |ID_{SNi}||T_4) \oplus r'_i \oplus D_{12}=h(K' || |ID_{SNi}||T_4) \oplus r''_i$ and sends a message $M_4=<D_8, D_9, D_{10}, D_{11}, D_{12}, T_4>$ to $U$.

AK4. On receiving the message at $T_5$, $U_i$ checks $|T_5 - T_4| \leq \Delta T$. Only if the verification is valid, $U_i$ computes $r''_i'=D_{12} \oplus h(K' || |ID_{SNi}||T_4)$ and $D_{13}=D_{11} \oplus h(K' || |ID_{SNi}||T_4)$ and then verifies if $D_9$ equals to $h(K' || |ID_{SNi}||T_4)$. If the verification does not hold, $U_i$ changes $DID_i$ with $D_9'$. Hence, it is confirmed that $SN_i$ is authentic. But if not, the connection is aborted. On the success of mutual authentication, a session key $SK=h(DID_i || r_i || r''_i || r_i'' || |ID_{SNi}|)$ is constructed by involved entities in the system.

3.5. Dynamic Node Addition Phase

It may happen that a new sensor node needs to be added over the target field as and when required, after the establishment of WSNs. So, $N/A$ deploys the new sensor node over the target region by performing the system setup phase in off-line mode. Then after, the newly added sensor node under-goes the sensor node registration phase and introduces the new sensor node into the setup network model.

3.6. Password Change Phase

In smart card based authentication, protocols should be able to address password related attack so that user with valid smart card and personal credentials can initiate the password change phase. Additionally, user should be able to choose and change the password without interaction with $N/A$ or $HGWN$, which is to provide user-friendly password selection and change. The proposed password change phase requires user can change the password without interaction with the other network entity.

A user $U_i$ with valid credentials and smart card can initiate the password change by inputting $ID'_i$ and $PW'_i$. $SC$ computes $i'=C_{\text{GWN}}(ID'_i || PW'_i)$. To resist password related attack, $SC$ verifies if $V_i$ equals to $h(ID'_i || PW'_i || i')$. Using this condition, $SC$ identifies the correctness of user credentials. If verification holds, $SC$ asks for the new password $PW_{new}$ to $U_i$. On receiving $PW_{new}$, $SC$ computes $YPW_{new}=h(PW_{new} || i')$ and updates $Y_i'=Y_i \oplus P_{\text{GWN}} \oplus P_{\text{PW_{new}}} C=C_{\text{GWN}}(ID_i || PW_i || ID'_i || ID''_i || PW_{new})$ and $V_i'=h(ID'_i || PW_{new} || i')$ on $SC$.

4. ANALYSIS

This section provide security analysis and performance analysis. First of all, we provide BAN logic analysis and informal security analysis to show the security and privacy of UAKA. Performance analysis is focused on computational and communicational overheads and provides comparisons of UAKA with the related schemes.

4.1. BAN Logic Analysis

In this section, we provide a formal protocol analysis of our proposed UAKA using the BAN logic method [21]. The BAN logic is used to verify the correctness of the authentication scheme with key agreement. UAKA correctness refers to the communication parties: a legal user $U_i$, GWN and an accessed sensor node $J_i$ who share a fresh shared session key with each other after the scheme is achieved. The formal analysis of UAKA using BAN logic involves following steps:

1. Converting original scheme statements to their idealized form.
2. Determining the assumptions about the initial state of the system.
3. Representation of the state of the system after executing each statement as logical assertions by attaching logical formulas to each statement.
4. Application of logical postulates to assumptions and assertions.

The following notations are used in formal security analysis using the BAN logic:

- $Q \models X$: Principal $Q$ believes the statement $X$.
- $\#(X)$: Formula $X$ is fresh.
- $Q \models \rightarrow X$: Principal $Q$ has jurisdiction over the statement $X$.
- $\models \rightarrow Q$: Principal $Q$ has a public key $K$.
- $Q \models \neg X$: Principal $Q$ sees the statement $X$.
- $Q \models \neg X$: Principal $Q$ once said the statement $X$.
- $(X, Y)$: Formula $X$ or $Y$ is one part of the formula $(X, Y)$.
\( (P) \) Q: Formula P combined with the formula Q.

- \( Q \leftarrow R \): Principal Q and R may use the shared session key, SK to communicate among each other.
  
In addition, the following four BAN logic rules are used to prove that UAKA provides a secure mutual authentication among U, HGWN and Sj:

**Rule 1. Message-meaning rule:**
\[
\frac{R | \exists R - S. R \equiv \langle X \rangle^y}{R | \exists S | \equiv X}
\]

**Rule 2. Nonce-verification rule:**
\[
\frac{R | \exists S | \equiv X}{R | \exists S | \equiv \langle X \rangle^y}
\]

**Rule 3. Jurisdiction rule:**
\[
\frac{R | \exists X}{R | \exists \#(XY)}
\]

**Rule 4. Freshness-concatenation rule:**
\[
\frac{R | \exists \#(X)}{R | \exists \#(XY)}
\]

In order to show that UAKA provides secure mutual authentication between among U, HGWN and Sj in UAKA to the idealized forms is as follows:

**Goal 1:** \( U_i \equiv (U \leftarrow S_j) \)

**Goal 2:** \( S_j \equiv (S_j \leftarrow U) \)

**Goal 3:** \( U_i \equiv S_j \equiv (S_j \leftarrow U) \)

**Goal 4:** \( U_i \equiv U_j \equiv (U_i \leftarrow S_j) \)

**Idealized form:** The arrangement of the transmitted messages among U, HGWN and Sj in UAKA to the idealized forms is as follows:

**Message 1:** \( U_i \rightarrow HGWN: \langle D|D_i\rangle_{\text{HGWN}}, \langle ID|ID_i\rangle, \langle D|D_i\rangle_{\text{HGWN}} \)

**Message 2:** \( HGWN \rightarrow S_j: \langle D|D_j\rangle_{\text{HGWN}}, \langle D|D_j\rangle_{\text{HGWN}} \)

**Message 3:** \( S_j \rightarrow HGWN: \langle D|D_j\rangle_{\text{HGWN}} \)

**Message 4:** \( HGWN \rightarrow U_i: \langle D|D_i\rangle_{\text{HGWN}} \)

**Assumptions:** The following are the initial assumptions of UAKA:

- A1: \( U_i \equiv \#(r_1, T_1) \)
- A2: \( HGWN \equiv \#(r_2, T_2) \)
- A3: \( S_j \equiv \#(r_3, T_3) \)
- A4: \( U_i \equiv (U_i \leftarrow HGWN) \)
- A5: \( HGWN \equiv (HGWN \leftarrow U) \)
- A6: \( HGWN \equiv (HGWN \leftarrow S_j) \)
- A7: \( S_j \equiv (S_j \leftarrow HGWN) \)
- A8: \( U_i \equiv S_j \equiv (U_i \leftarrow S_j) \)
- A9: \( S_j \equiv U_i \equiv (S_j \leftarrow U) \)

**Proof:**

In the following, we prove the test goals in order to show the secure authentication using the BAN logic rules and the assumptions.

Based on Message 1, we could derive:

**Step 1.** \( HGWN \leftarrow <D|D_i \rangle_{\text{HGWN}}, <ID|ID_i >, <D|D_i >_{\text{HGWN}}, <D|T_1 > \)

According to assumption A4 and the message meaning rule, we could get:

**Step 2.** \( HGWN \equiv U_i \equiv \#(D|D_i \rangle_{\text{HGWN}}, <ID|ID_i >, <D|D_i >_{\text{HGWN}}, <D|T_1 >) \)

According to assumption A1 and the freshness concatenation rule, we could get:

**Step 3.** \( HGWN \equiv \#(<D|D_i \rangle_{\text{HGWN}}, <ID|ID_i >, <D|D_i >_{\text{HGWN}}, <D|T_1 >) \)

According to Step 2, Step 3 and the nonce verification rule, we could get:

**Step 4.** \( HGWN \equiv U_i \equiv \#(<D|D_i \rangle_{\text{HGWN}}, <ID|ID_i >, <D|D_i >_{\text{HGWN}}, <D|T_1 >) \)

According to Step 4, assumption A4 and the believe rule, we could get:

**Step 5.** \( HGWN \equiv U_i \equiv \#(<D|D_i \rangle_{\text{HGWN}}, <ID|ID_i >, <D|D_i >_{\text{HGWN}}, <D|T_1 >) \)

According to the jurisdiction rule, we could get:

**Step 6.** \( HGWN \equiv (HGWN \leftarrow U) \)

Based on Message 2, we could derive:

**Step 7.** \( S_j \equiv <D|D_i \rangle_{\text{HGWN}}, <D|T_2 >_{p_1}, <D|T_2 >_{p_2}, <D|T_2 >_{p_3} \)

According to assumption A7 and the message meaning rule, we could get:

**Step 8.** \( S_j \equiv HGWN \equiv \#(<D|D_i \rangle_{\text{HGWN}}, <D|T_2 >_{p_1}, <D|T_2 >_{p_2}, <D|T_2 >_{p_3}) \)

According to assumption A2 and the freshness concatenation rule, we could get:

**Step 9.** \( S_j \equiv \#(<D|D_i \rangle_{\text{HGWN}}, <D|T_2 >_{p_1}, <D|T_2 >_{p_2}, <D|T_2 >_{p_3}) \)

According to Step 8, Step 9 and the nonce verification rule, we could get:

**Step 10.** \( S_j \equiv HGWN \equiv \#(<D|D_i \rangle_{\text{HGWN}}, <D|T_2 >_{p_1}, <D|T_2 >_{p_2}, <D|T_2 >_{p_3}) \)

According to Step 10, assumption A6 and the believe rule, we could get:

**Step 11.** \( S_j \equiv HGWN \equiv (HGWN \leftarrow S_j) \)

According to the jurisdiction rule, we could get:

**Step 12.** \( S_j \equiv (S_j \leftarrow HGWN) \)

According to Step 8, Step 9, Step 10 and the nonce verification rule, we could get:

**Step 13.** \( S_j \equiv U_i \equiv \#(U_i \leftarrow S_j) \)

According to assumption A8 and the jurisdiction rule, we could get:

**Step 14.** \( S_j \equiv \#(S_j \leftarrow U) \)

Based on Message 3, we could derive:

**Step 15.** \( HGWN \leftarrow <D|T_2 >_{p_1}, <D|T_2 >_{p_2}, <D|T_2 >_{p_3} \)
According to assumption A6 and the meaning rule, we could get:
Step 16. \(HGWN|\equiv \Sigma_j \Circ <D_{i_0}>_{p_j} <D_{i_1}>_{p_j} <T_j>\)
According to assumption A3 and the freshness concatenation rule, we could get:
Step 17. \(HGWN|\equiv \#<D_{i_0}>_{p_j} <D_{i_1}>_{p_j} <T_j>\)
According to Step 16, Step 17 and the nonce verification rule, we could get:
Step 18. \(HGWN|\equiv \Sigma_j \Circ <D_{i_0}>_{p_j} <D_{i_1}>_{p_j} <T_j>\)
According to Step 18, assumption A7 and the believe rule, we could get:
Step 19. \(HGWN|\equiv \Sigma_j \Circ <D_{i_0}>_{p_j} \circ HGWN\)
According to Step 16, Step 17, Step 18 and the nonce verification rule, we could get:
Step 20. \(HGWN|\equiv \Sigma_j \Circ <D_{i_0}>_{p_j} \circ HGWN\)
According to assumption A10 and the jurisdiction rule, we could get:
Step 21. \(HGWN|\equiv (HGWN \circ \Sigma_j)\)
Based on Message 4, we could derive
Step 22. \(U_i \Circ <D_{i_0}>, <D_{i_1}>_{k_0}, <D_{i_1}>, <D_{i_2}>_{k_0}, <T_2>\)
According to assumption A4 and the meaning rule, we could get:
Step 23. \(U_i |\equiv HGWN|\equiv \Sigma_j <D_{i_0}, <D_{i_1}>_{k_0}, <D_{i_2}>_{k_0}, <T_2>\)
According to assumption A2 and the freshness concatenation rule, we could get:
Step 24. \(U_i |\equiv \#<D_{i_0}, <D_{i_1}>_{k_0}, <D_{i_2}>_{k_0}, <T_2>\)
According to Step 23, Step 24 and the nonce verification rule, we could get:
Step 25. \(U_i |\equiv HGWN|\equiv <D_{i_0}, <D_{i_1}>_{k_0}, <D_{i_2}>_{k_0}, <T_2>\)
According to Step 25, assumption A5 and the believe rule, we could get:
Step 26. \(U_i |\equiv HGWN|\equiv HGWN|\equiv (HGWN \circ \Sigma_j)\)
According to Step 23, Step 24, Step 25 and the nonce verification rule and the jurisdiction rule, we could get:
Step 27. \(U_i |\equiv \Sigma_j \Circ <D_{i_0}, <D_{i_1}>_{p_j} \circ HGWN\)
According to assumption A8 and the jurisdiction rule, we could get:
Step 28. \(U_i |\equiv (U_i \circ \Sigma_j)\)
According to Steps 14 and 28, UAKA successfully achieves both goals (Goals 1 and 2). Both \(U_i\) and \(\Sigma_j\) believe that they share a common session key \(SK=\circ b(ID_{NJ} \circ \Sigma_j)\).

4.2. Informal Security Analysis

Although it is important to provide a formal security proof on any cryptographic protocol, the formal security proof of protocols remains one of the most challenging issues for cryptography research. Until now, a simple, efficient and convincing formal methodology for correctness analysis on security protocols is still an important subject of research and an open problem. Because of these reasons, most protocols have been demonstrated with a simple proof. This section follows the security analysis approaches used in [22]. As shown in Table 1, the security analysis is focused on verifying the overall security requirements for UAKA, including passive and active attacks, as follows.

<table>
<thead>
<tr>
<th>Table 1. Comparison of Security Features.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Attributes</td>
</tr>
<tr>
<td>Masquerading attack</td>
</tr>
<tr>
<td>Replay attack</td>
</tr>
<tr>
<td>Trace attack</td>
</tr>
<tr>
<td>Insider attack</td>
</tr>
<tr>
<td>Password guessing attack</td>
</tr>
<tr>
<td>DoS attack</td>
</tr>
<tr>
<td>Anonymity and unlinkability</td>
</tr>
</tbody>
</table>

Proposition 1. UAKA is secure against HGWN masquerading attack.

Proof: By definition this is the attack in which an attacker pretends to be a legitimate HGWN and plays in between user and sensor node with the assumption that the attacker could obtain any messages transmitted in the previous sessions. In UAKA, the attacker could try to form \(M_1=<D_{i_0}, D_{i_1}, D_{i_2}, T_1>\) or \(M_2=<D_{i_0}, D_{i_1}, D_{i_2}, D_{i_3}, D_{i_4}, T_2>\) right after receiving \(M_3=<D_{i_0}, D_{i_1}, D_{i_2}, D_{i_3}, D_{i_4}, T_1>\) from \(U_i\) for the trial of this attack. However, they are impossible to the attacker in UAKA because they require knowledge of the important secret key, \(X_{HGWN}\) of HGWN. Again to \(U_i\), the attacker needs to form a correct \(M_1=<D_{i_0}, D_{i_1}, D_{i_2}, D_{i_3}, D_{i_4}, T_1>\) which requires the knowledge of \(K'\) where \(K'=<b(ID_{NJ} \circ X_{HGWN})\). Without the knowledge of \(X_{HGWN}\), the attacker could not form the proper message \(M_1\). In the other hand, against to \(SN_j\), the attacker needs to form \(M_2=<D_{i_0}, D_{i_1}, D_{i_2}, T_2>\), which requires the knowledge of \(P\) where \(P=b(ID_{NJ} \circ X_{HGWN})\). The attacker could not do anything to form the proper message with the same reason for \(U_i\). There is no feasible way the attacker knows \(X_{HGWN}\) or \(P\). Hence we can confirm that UAKA resists HGWN masquerading attack.

Proposition 2. UAKA is secure against \(SN_j\) masquerading attack.

Proof: With the similar definition of the attack on HGWN and the assumption, to masquerade as a legitimate sensor node \(SN_j\), an attacker needs to form a proper response message \(M_3=<D_{i_0}, D_{i_1}, D_{i_2}, T_1>\) to \(X_{HGWN}\). For the attacker to do this, he (or she) must have the knowledge of \(P=b(ID_{NJ} \circ X_{HGWN})\). However, it is not possible in UAKA as the attacker does not have the knowledge.
of the secret key $X_{HGWN}$ of the involved parties. Thus it will be impossible for him (or her) to compute the message $M_3$ correctly. Therefore, UAKA can resist the sensor node masquerading attack.

**Proposition 3.** UAKA is secure against replay attack,

Proof: By definition replay attack is an attack where the attacker captures the previously transmitted messages and uses them during UAKA execution to make the receiver of the message believe that the transmitted message is from a legal entity. In order to justify UAKA resist from the replay attack, we assume that the attacker has captured the previous session messages of UAKA and later tries to transmit the same message to the targeted. In a replay attack, it does not matter if the attacker who intercepted the original message can read or decipher the key. All he (or she) has to do is capture and resent the entire thing — message and key — together. To counter this possibility, UAKA uses random session key, which is a type of code that is only valid for one transaction and cannot be used again. For an example, when preparing message one $M_i = <DID, ID_{SN_i}, D_i, D_s, T_i>,$ UAKA uses $r_i$ as its random number while in when preparing message two $M_t = <D_t, D_s, D_{SN_t}, T_t>,$ HGWN generates $r_t$ as its random number and $SN_t$ generates $r_t$ as its random number to send a message to HGWN. Another preventative measure for this type of attack is using time stamps on all messages as you can see each and every message has a time stamp as it is sent, this prevents hackers from resenting messages sent longer ago than a certain length of time, thus reducing the window of opportunity for an attacker to eavesdrop, siphon off the message, and resent it. In specific, a time stamp and random number mechanisms are used to guarantee the freshness of each message. Following this we can conclude that UAKA’s scheme is strong against replay attack.

**Proposition 4.** UAKA could withstand trace attack

Proof: Trace attack is an attack where the attacker can distinguish the messages communicated between entities by eavesdropping on a communication. For an attacker to achieve this, he (or she) intercepts two or more messages from two or more different sessions and checks whether they have something in common that can be computed by the attacker. If it happens, the attacker believes that these two messages belong to the same source, either from $U_i$ or HGWN. However, the attacker cannot trace $U_i$ HGWN, $SN_t$ after intercepting the communicating messages because UAKA’s scheme updates $DID$ and $K$ apart from that he uses the one-way hash function which is infeasible for an attacker to compute important parameters such as $X_{HGWN}$

**Proposition 5.** UAKA could withstand privileged insider attack

Proof: An insider attack is defined as a malicious attack perpetrated on a network or computer system by a person with the authorized system access. Practically, in UAKA, it is assumed that the HGWN is trusted. So, the HGWN provides confidentiality to the user’s credentials, where leakage of any confidential parameters of the user is not permitted. But, it is observed that due to the presence of an insider, systems can get hacked. Therefore, the user’s information such as identity and password should be kept secret such that the insider of the HGWN cannot gain control over the user’s information. In UAKA, during the user’s registration phase, instead of the original $ID$ and $PW$, we have transmitted the masked identity $TID = h(ID||a)$ and password $RPW = h(PW||a)$ to HGWN. Hence, extracting the user’s password or identity by the insider of the HGWN is computationally infeasible due to the non-invertible property of the cryptographic one-way hash function. Therefore, UAKA can resist privileged insider attack.

**Proposition 6.** UAKA could withstand password guessing attack

Proof: A password guessing attack is an attack that consists of an attacker trying many passwords or pass phrases with the hope of eventually guessing correctly. We suppose that the user $U_i$’s smart card was stolen by an attacker, then the attacker can extract the information stored on the smart card $<Y_d, DID, b(\cdot), ID_{SN_i}, C_i, V>$. By using the method of power analysis, where $V = h(ID||PW||a)$, $C_i = h(ID||PW)$ and $V = K \oplus R PW$. The attacker needs to know $a$, $ID$, and $PW$, where this information is known only to the user, and both user $ID$ and $PW$ are unknown to the attacker because they are well protected by the one way hash function. So, the attacker has no way to guess or extract user $ID$ and $PW$, at the same time, as it is computationally infeasible to guess the two parameters at the same time and also, due to the non-invertible one-way hash function property. Hence, there is nowhere for an attacker to update the $PW$ of the user $U$. Therefore, UAKA is free from the stolen smart card attack.

**Proposition 7.** UAKA could withstand denial-of-service (DoS) attack diagram.

Proof: DoS attack is an attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. In UAKA, we have three possibilities where a registered user could encounter DoS. However, UAKA is efficient to resist DoS attack in all scenario as follows: In first situation when a user inputs incorrect credentials (identity or password) unknowingly during login phase, however, the smart card can correctly verify the login parameters such as $ID_{SN_i}$ and $K$ apart from that he uses the one-way hash function $h$ which is infeasible for an attacker to compute important parameters such as $X_{HGWN}$. Therefore, UAKA can resist DoS at all scenario.
attack is different from an entire network attack. However this is not feasible in our scheme since our scheme involves the use of one way-hash function which very difficult for an attacker to compute it.

**Proposition 8.** UAKA provides anonymity and unlinkability

Proof: Anonymity is a property of network security. An entity in a system has anonymity if no other entity can identify the first entity, nor is there any link back to the first entity that can be used, nor any way to verify that any two anonymous acts are performed by the same entity.

As shown in proposition 6, it is clear from UAKA that an attacker has no way to obtain or guess the identity \( ID_u \) of the user \( U \), as it is protected by the non-invertible cryptographic one way hash function not only that but also the use of pseudo-identity makes the system anonymous thus UAKA provides anonymity and also unlikability.

4.3. Performance Analysis

In this section, we compare the performance and functionality features of UAKA with the related existing schemes proposed for WSNs. This evaluation gives an insight into the effectiveness of UAKA.

Table 2. Comparison of Communication and Performance Overhead.

<table>
<thead>
<tr>
<th>Features</th>
<th>[12]</th>
<th>[13]</th>
<th>[14]</th>
<th>UAKA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total number of messages</td>
<td>4</td>
<td>4/8</td>
<td>4/7</td>
<td>4</td>
</tr>
<tr>
<td>Computational overhead</td>
<td>( 32Ti )</td>
<td>( 20Ti / 20Ti )</td>
<td>( 29Ti / 35Ti )</td>
<td>( 2Ti_{c} + 30Ti )</td>
</tr>
</tbody>
</table>

Table 2 shows the communication overhead required during the login and authenticated key agreement phases between UAKA and the related schemes. The length of message has also effects to the communication overhead. However, the number of messages is much important factor to be compared. UAKA only requires 4 messages in any cases. Thereby, UAKA has a good property in the communication overhead concern compared to the other schemes. However, UAKA has a bit more computational overhead due to provide anonymity and unlikability, which requires two symmetric key cryptosystem operations. Based on MIRACL library with 32-bit Windows 7 operating systems and Visual C++ 2008, symmetric key cryptosystem operation and hash function require 0.1303 ms and 0.0004 ms if advanced encryption standard and secure hash algorithm 1 are used [23].

5. CONCLUSION

In this paper, we have proposed an unlikable user authenticated key agreement scheme for multi-gateway WSNs. Security validation of the proposed scheme has analyzed using BAN logic and informal cryptanalysis proofs the resilience of relevant security and privacy attacks. Even if WSNs are resource constrained, they should consider and provide privacy provision to their entities. To solve this requirement, the proposed scheme adopted the symmetric key cryptosystem operation, which requires a bit overhead than the other related schemes. However, it has a good property in communication overhead concern as shown in Table 2.

**ACKNOWLEDGEMENTS**

The results in this paper are part of Solomon Kuonga Nhoma’s Master degree thesis. Hyunsung Kim is the corresponding author. This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2017R1D1A1B04032598).

**CONFLICT OF INTEREST**

The author declares that there are no conflicts of interest.

**REFERENCES**


**Unlinkable User Authenticated Key Agreement**

**10**


