**RUASN: A Robust User Authentication Framework for Wireless Sensor Networks**

**Pardeep Kumar**, **Amlan Jyoti Choudhury**, **Mangal Sain**, **Sang-Gon Lee** and **Hoon-Jae Lee**

1. Department of Ubiquitous-IT, Graduate School of Design & IT, Dongseo University, Sasang-Gu, Busan 617-716, Korea; E-Mails: pradeepkhgl@gmail.com (P.K.); choudhuryamlanjyoti@gmail.com (A.J.C.); mangalsain1@gmail.com (M.S.)
2. Division of Computer & Information Eng, Dongseo University. San 69-1, Jurye-2-Dong, Sasang-Gu, Busan 617-716, Korea; E-Mail: hjlee@dongseo.ac.kr

* Author to whom correspondence should be addressed; E-Mail: nok60@dongseo.ac.kr; Tel.: +82-51-320-1730; Fax: +82-51-327-8955.

Received: 25 February 2011; in revised form: 17 April 2011 / Accepted: 20 April 2011 / Published: 4 May 2011

**Abstract:** In recent years, wireless sensor networks (WSNs) have been considered as a potential solution for real-time monitoring applications and these WSNs have potential practical impact on next generation technology too. However, WSNs could become a threat if suitable security is not considered before the deployment and if there are any loopholes in their security, which might open the door for an attacker and hence, endanger the application. User authentication is one of the most important security services to protect WSN data access from unauthorized users; it should provide both mutual authentication and session key establishment services. This paper proposes a robust user authentication framework for wireless sensor networks, based on a two-factor (*password and smart card*) concept. This scheme facilitates many services to the users such as user anonymity, mutual authentication, secure session key establishment and it allows users to choose/update their password regularly, whenever needed. Furthermore, we have provided the formal verification using Rubin logic and compare RUASN with many existing schemes. As a result, we found that the proposed scheme possesses many advantages against popular attacks, and achieves better efficiency at low computation cost.
Keywords: wireless sensor network security; user authentication; user anonymity; session key establishment; confidentiality

1. Introduction

Wireless sensor networks (WSNs) are becoming more and more popular in everyday life as they offer economically viable, real time monitoring solutions. These wireless sensors can be quickly and easily deployed in hostile environments, and WSNs are now widely used in a variety of real-time applications, such as vehicular tracking, habitat monitoring, environment control, military surveillance, healthcare monitoring, wildlife monitoring and traffic monitoring. One recent survey declared that, in the near future, WSNs will become an intelligent and integral part of daily lives [1].

A WSN consists of a discrete group of independent, low cost, low power nodes with limited memory and computation power. They communicate wirelessly over limited frequency and low bandwidth [1]. More specifically, sensor nodes collectively monitor the area and sense substantial amounts of data, which are transmitted to the base-station traversing some nodes via RF signals and routing schemes.

As sensor nodes are resource constrained devices and are often deployed in a hostile environment and have to sense the information properly and efficiently. So the potential deployment of WSNs for any real-time applications has to deal with many challenges, including security, system architecture and protocol functionalities. Providing security to these resource hungry sensor networks is a very tedious task as compared to conventional networks, such as local area networks (LANs) and wide area networks (WANs). Consequently, providing suitable security has emerged as one of the critical issue in wireless sensor networks, and the state-of-art should therefore pay attention to how to deploy user-friendly, reliable and secure WSNs.

In real-time WSNs, sensor data queries are commonly issued from the base-station nodes or the backend application system. Moreover, these sensor networks can be accessed from anywhere in an ad-hoc manner. As sensor nodes provide services to users by themselves, it is necessary to control who is accessing the information and whether the intended user is authenticated to do so. Therefore, access control is a core requirement for WSNs to protect the data from access by unauthorized parties. In general user authentication where each user must verify their legitimacy is considered as one of the basic solutions for the access control issue.

So far, a number of significant schemes that provide adequate security for wireless sensor networks at the link layer [2-6] and the network layer [7] have been proposed. However, secure user authentication at the application layer has not been addressed effectively in order to prevent illegal access to sensor data. A review of current literature on WSNs reveals that only a few user authentication schemes have been adequately addressed [8-22] at the application layer. In [8-14], protocols are based on traditional passwords, where secrets are stored at the base station or the gateway. While, the protocols presented in [18-21] are based on two-factor user authentication with limited functionality (e.g., no mutual authentication, no secure session key, and no confidentiality) at high computation cost. Therefore, one of the primary concerns in wireless sensor network applications
is the design and development of a robust user authentication scheme which is suitable for hostile or unattended environments.

In this paper, we have considered the above challenges and present a robust user authentication framework for wireless sensor networks (RUASN) at the application layer, which uses the two factor approach. The first factor (something you know) refers to something that is known by the user, such as a password, while the second factor (something you have) refers that something that is embedded on a device, such as smart cards, software tokens, digital certificates or biometric identifiers (e.g., fingerprint scans and so on) [23].

The proposed RUASN framework achieves user authentication (access control) for wireless sensor networks, where a user must login with same identity. The proposed scheme is resists many popular attacks, such as replay attack, impersonation attack, insider attack, stolen-verifier attack, password guessing attack, and man-in-the-middle attack. RUASN provides user privacy protection (i.e., user anonymity), mutual authentication, and secure session key establishment. In addition, a user can update his/her password whenever demanded. Our framework uses one-way hash functions along with XOR operations to attain low computational overheads. Moreover, this paper further demonstrated the analysis and verification of the proposed protocol using Rubin Logic [24], which is very close to actual implementation.

The rest of the paper is structured as follows: Section 2 briefly reviews the related literature and the perceived weaknesses of existing schemes. In Section 3 we discuss the design goals, security requirements and system architecture of RUASN. In Section 4, we propose a robust two-factor user authentication framework in detail. Section 5 discusses the nonmonotonic cryptographic protocol and formal verification of proposed protocol using Rubin Logic. Section 6 discusses the security analysis, efficiency evaluation and comparison with existing schemes for wireless sensor networks. Finally, Section 7 conclusions are drawn for RUASN.

2. Literature Review

In this section, we will discuss the literature on user authentication schemes that have been recently proposed to verify the legitimacy of wireless sensor networks users.

Benenson et al. [8] first described several security issues in WSNs, especially the access control problem, and proposed the notion of n-authentication, where users can successfully authenticate with at least (n-t) of n-sensors, where t is the number of sensor nodes that the adversary can compromise. Subsequently, Benenson et al. [9] proposed another solution for the user authentication problem in the face of node capture attacks. The proposed scheme is based on public key cryptography (PKC) and elliptic curve cryptography (ECC). Some major weaknesses were pointed out in the Benenson et al. scheme, such as the fact that impersonation attacks or denial-of-service (DoS) attacks could be mounted by sending many bogus signatures during the authentication phase [10]. Moreover, the computation cost of PKC and ECC is very high for sensor networks.

Wong et al. [10] proposed a dynamic user authentication scheme for wireless sensor networks, which is based on passwords. This scheme imposes a very light computation cost that requires only one-way hash functions and simple XOR operations. The scheme consists of three phases: registration phase, login and authentication phase. Unfortunately, the Wong et al. scheme is
vulnerable to many attacks such as replay attacks, forgery attacks, stolen-verifier attacks and password guessing attacks [11,12,14,18].

Vaidya et al. [14] pointed out some weaknesses of the Tseng et al. [11], Wong et al. [10] and Ko [14] schemes such as replay of account-login attacks, man-in-the-middle (MITM) attacks, forgery attacks and stolen-verifier attack with node capture attacks. They proposed two user authentication schemes for wireless sensor networks, which are based on traditional password schemes and claimed that their proposed scheme provides better security features as compared to the Wong et al., Tseng et al. and Ko et al. schemes.

Recently, Das [18] pointed out some security flaws in Wong et al. [10] scheme, such as the fact this scheme is vulnerable to many logged in users with the same login-id threat and also susceptible to stolen-verifier attacks. Das [18] proposed a two-factor user authentication for wireless sensor networks, where the legitimate users must prove the possession of both a password and a smart card. Das [18] claimed that his scheme is secure against many types of attacks (e.g., user authentication, replay, guessing, impersonation, node compromise, and stolen-verifier attacks).

Nyang and Lee [19] noted that Das’ scheme is not practical and is vulnerable to an offline password guessing attack by insiders, node compromise attacks and does not care about other security issues, i.e., encryption and authenticity verification of query responses. Consequently they proposed an enhanced two-factor user authentication protocol for WSNs, which overcome the Das scheme’s security flaws with some additional security services such as confidentiality and authenticity of user query responses. However, their scheme also does not care about mutual authentication and there is no provision for password updates.

Khan and Alghathbar [20] pointed out that the Das et al. [18] scheme is still not secure and cannot resist many other security attacks, such as gateway-node bypass attacks, and it is vulnerable to insider attacks, as well as not facilitating mutual authentication between the gateway and the sensor nodes and there is no provision for users to change their passwords. Khan and Alghathbar [20] overcome the security weaknesses of Das’ scheme and proposed an improved two-factor user authentication in WSNs, which provides protection against insider attacks, gateway bypass attacks and introduced a password change phase for users. They suggested two secret ($X_a$ and $X_i$) values to be used to overcome the gateway-bypass attacks. For example, $X_a$ is used between the user and the gateway, and $X_i$ is used between the gateway and the sensor nodes. Furthermore, to overcome the room for insider attack, they passed hashed passwords instead of plain passwords. Their scheme also does not however provide mutual authentication between the user and the gateway.

More recently, He et al. [21] have shown that Das’ protocol is susceptible to insider attacks, impersonation attacks and also found design weaknesses (i.e., concerning real identity of user). Later, they proposed an enhanced two-factor user authentication scheme for WSNs that facilitates user anonymity, protection against insider attacks and allows users to change their passwords. Their scheme does not care about mutual authentication between all parties (i.e., user, gateway and sensor). Moreover, the communication cost is quite high as compared with other schemes (see [10-14,18-20]).

As we have seen above, a number of user authentication schemes have been proposed in order to authenticate user legitimacy for sensor networks. These schemes are not designed properly [10-14] (i.e., stored secrets at the gateway) and in [18-21] they do not provide essential services to the users. Consequently, we conclude that the above schemes have many security flaws and provide less security
services. Thus, these security weaknesses and constrained nature of sensor nodes motivated us to design a robust user authentication framework that provides adequate security and provides users with many services, as discussed in the next section.

3. Design Goals, Security Requirements and System Architecture

In this section, we discuss the design goals, security requirements and system architecture for robust user authentication for wireless sensor networks.

3.1. Design Goals and Security Requirements

RUASN provides transparent security service for wireless sensor networks. In this paper another goal is to design a simple and user-friendly framework, which will be suitable for real-time WSNs applications. Overall, RUASN is designed with the following characteristics:

Proper user authentication: A user must prove his/her authenticity, so that only authentic users can access the WSN data.

Mutual authentication: Every entity (user, gateway and sensor) must be mutually authenticated; hence they can ensure the communication is only taking place between authentic entities.

User anonymity: Since all the messages are broadcasted wirelessly, which means that an attacker may simply eavesdrop on the messages and could breach the privacy of a user, thus, the scheme must provide user anonymity.

Session key establishment: A session key should be established between a user and sensor node, so that subsequent communication could take place securely.

Confidentiality: It is desirable that a user authentication protocol facilitate confidentiality of messages; as a result, these confidential messages can only be used by authorized users.

Password update: A password based user authentication scheme should provide users a password update facility so that a user can update his/her password freely.

Low communication and computational cost: Since sensor nodes are resource constrained devices (e.g., MicaZ [25], and Telosb [26]) and, in general, application functions also need room for executing their tasks. So, the scheme must be efficient in terms of communication and computational cost.

Robust against popular attacks: Clearly, the scheme should defend against different popular attacks, such as replay, impersonation, insider, stolen-verifier, password guessing, and man-in-the-middle attacks. As a result, the scheme should be easily applicable to real-world applications.

User-Friendliness: The system architecture should be easy to deploy for the WSN applications as well as user-friendly for non-advanced users so users can update his/her password securely according to their will.

Reliability: From the security point of view, the gateway needs to check the validity of the users and the sensor nodes, so that reliable communication can be established between all the parties.
Moreover, Liao *et al.* [27] identified some security requirements to evaluate a smart card and password based authentication protocol. Their requirements solve most of the problems in smart card oriented schemes. As sensor networks are resource constrained devices, WSN could adopt Liao *et al.* requirements, which are listed below:

- The password tables should not be stored inside the gateway.
- Verification table should not be stored inside the gateway.
- The password should not be transmitted as plain text over the public network.
- Schemes should resist insider-attacks.
- Password length should be sufficient.
- The password is not exposed by the gateway administrator.
- Scheme should resist offline password guessing attacks.

### 3.2. System Architecture

Wireless sensor networks consist of a number N of low-cost sensor devices, which are scattered in a hostile environment. These sensors sense the environmental information, *(e.g., humidity, pressure, and temperature)* and transmit information to the users for further analysis. A user can access real-time WSN data using their mobile devices *(e.g., laptop, PDA or smart-phone)* through wireless communication. WSN data could be easily accessible from anywhere in ad-hoc manner. In real-time environment it is obvious that the gateway nodes and the users are able to access the sensor data directly. The basic system architecture is shown in Figure 1, where a user directly request to the sensor node, upon receiving user request sensor node first verify user authenticity through the gateway node.

**Figure 1.** The basic system architecture for RUASN.
After confirmation of the user’s legitimacy he/she can access the real-time sensor data. Furthermore, the gateway node provides a middle ground between the users and the sensor network.

4. RUASN: Robust User Authentication for Wireless Sensor Network

To solve the potential problems of user authentication for WSNs, we propose RUASN which ensures WSN data are only accessed by legitimate users. Thus, before issuing a query to a sensor node, each user must register with the gateway in a secure manner so that they can access the real time sensors’ data. Upon the successful user registration request, the gateway node personalizes a smart card for every registered user, as shown in Figure 1. Then, a user can submit his/her query in an authentic way and access the sensor network data at any time within an administratively configurable period [10].

In order to execute the proposed framework, we considered that the gateway is a trusted node and it hold two master keys (x and y), which are sufficiently large for the sensor network. Before starting the system, it is assumed that the gateway and the sensor nodes share a long-term common secret key, i.e., \( SK_{gw} = h(Sn||y) \) using any key agreement protocol. For example, [4] demonstrated that, with the careful design, D-H key agreement protocol [28] and RSA public key cryptosystem [29] can be easily deployed on most constrained devices. Here, \( h(.) \) is a collision free one-way hash function (i.e., SHA-1), which has an output length of 160-bits [30] and is used throughout this paper. It is assumed that some identical secure symmetric cryptosystems are publically available and stored on the user device, on the gateway and the sensor node. As a result only the users registered with the gateway have access privilege to the sensors, which share a long-term secret with the gateway. The framework is divided into four phases, namely, user registration phase, login phase, authentication phase and password update phase. For convenience Table 1 provides a list of some notations and symbols will be used throughout the rest of paper.

| Notation | Descriptions |
|----------|--------------|
| \( GW \) node | WSN gateway node |
| \( U_k \) | \( k^{th} \) User to be login |
| \( ID_k \) | Login_ID of \( U_k \) |
| \( PW_k \) | Password of \( U_k \) |
| \( x \) and \( y \) | Gateway master keys |
| \( r \) | Arbitrary random number selected by user |
| \( l \) | Random number generated by the \( GW \) node |
| \( E_{key} \{m\} \) | Message \( m \) is encrypted with symmetric key |
| \( D_{key} \{m\} \) | Message \( m \) is decrypted with symmetric key |
| \( MAC_{key} \{m\} \) [31] | Message authentication code over message \( m \) with secret key |
| \( Sn \) | Sensor Node ID |
| \( X_g \) | Secret parameter generated by the user \( U_k \) |
| \( h(.) \) | Cryptographic hash function |
| \( \oplus \) | Bitwise XOR operation |
| \( \| \) | Concatenation operation |
4.1. Registration Phase (RP)

In the registration phase, initially, each user must register with the GW node. A user $U_k$ chooses his/her identity ($ID_k$). Now user $U_k$ chooses password ($PW_k$) and selects an arbitrary random number $r$ which should be sufficiently large and computes $h(r \oplus PW_k)$. The $ID_k$ and $PW_k$ should include uppercase, lowercase characters and 0–9 numeric characters [32]. Afterward, the user submits a registration request to the GW node over a secure channel. Upon receiving the $U_k$ registration request, the GW node computes:

1. $A_k = h(ID_k \oplus l)$
2. $B_k = E_x[ID_k || l]$
3. $V_k = h(ID_k || h(r \oplus PW_k))$

Thereafter, the GW node personalizes a smart card for the user with the parameters {$A_k, B_k, V_k, h(\cdot), E_x[\cdot], D_x[\cdot]$}. Here, $l$ is random number which is generated by the GW node and $x$ is the gateway secret key and $h(\cdot)$ is a collision free one-way function, e.g., SHA-1 [30]. Subsequently, user $U_k$ enters $r$ into his/her smart card, by doing so, user $U_k$ need not memorize the arbitrary random number. Now the smart card contains {$A_k, B_k, V_k, h(\cdot), E_x[\cdot], D_x[\cdot], r$}. This step completes the registration phase and the process flow is shown in Figure 2.

**Figure 2.** Flow of registration phase.

4.2. Login Phase (LP)

This phase is invoked whenever a user $U_k$ wants to submit his/her query to access the sensor, every time he/she has to complete the login phase. Figure 3 shows both the login phase and the authentication phase. The user $U_k$ inserts his/her smart card into the terminal and inputs his $ID_k$ and $PW_k$. 
Figure 3. Flow of login and authentication phases.

- \( V_k^* = h(ID_k || h(r \oplus PW_k)) \) and verify \( V_k^* = V_k \), if not, then reject the login request.
- \( H_k = h(A_k) \), and generate \( X_g \)
- \( AID_k = E_{TkTu}[h(ID_k || Sn || X_g)] \), here, \( Tk_{Tu} = (Tu || H_k) \)
- \( M1 = <B_k, AID_k, Tu> \)

- \( (Ts - Tu) \geq \Delta T \), if yes, then abort
- \( M2 = <B_k, AID_k, Tu, Ts, Sn> \),
- \( Q = MAC_{SKgs}(B_k || AID_k || Tu || Ts || Sn) \).

\(<M2, Q> \)

- \( (Tg - Ts) \geq \Delta T \), if yes, then abort
- \( Q' = MAC_{SKgs}(M2) \) and verify \( Q' = Q \), if not, then abort
- Obtain \( ID_k' \) and \( l' \) from \( B_k \)
- \( h(ID_k'), A_k' = h(ID_k' \oplus l') \), and \( H_k' = h(A_k') \)
- \( Tk_{Tu} = (Tu || H_k') \) and
- Obtains \( (h(ID_k^*) || Sn^* || X_g) \) from \( AID_k \)
- Verify: \( h(ID_k^*) = h(ID_k^*) \) and \( Sn = Sn^* \), if not, then abort
- \( M3 = <Tg, C> \), here \( C = E_{SKgs}[h(ID_k)||X_g][Tg] \)

- \( (Ts - Tg) \geq \Delta T \), if yes, then abort
- Obtains \( [h(ID_k)||X_g][Tg^*] \) from \( C \)
- Verify: \( Tg = Tg^* \), if not, then abort
- \( Ses_k = h(h(ID_k)||X_g||Sn||Ts||Tu) \)
- \( M4 = <L, Ts> \),
  here \( L = E_{SKgs}[X_g][Ts] \)

- \( (Tu - Ts) \geq \Delta T \), if yes, then abort
- \( Ses_k = h(h(ID_k)||X_g||Sn||Ts||Tu) \)
- Obtains \( [X_g^*][Ts^*] \) from \( L \)
- Verify: \( X_g = X_g^* \) and \( Ts = Ts^* \), if not, then abort

As a result, he/she believes that the sensor node \( Sn \) is authentic, otherwise not.
Now, the smart card performs the following operations:

(LP-1). Compute: $V_k^* = h(ID_k || h(r \oplus PW_k))$.
(LP-2). Check whether $V_k^*$ and $V_k$ are equal or not. If not, then reject the login request, otherwise, the user is a legal user and go to the next step.
(LP-3). Compute: $H_k = h(A_k)$.
(LP-4). Compute: $AID_k = E_{TkTu}(h(ID_k)||Sn||X_g)$; here, $TkTu = (Tu||H_k)$ is short-term key and $Sn$ is the sensor node, which the user $U_k$ wants to access. Here, $X_g$ is a secret random number generated by the user $U_k$ at login time, which is helpful to generate the session key between the user and the sensor node. $Tu$ is denoting the current timestamp of $U_k$’s system, which resist replay attacks.
(LP-5). Send the login message $M1 = <B_k, AID_k, Tu >$ to the sensor node.

Now, this is the end of login phase and the user login request message $<M1>$ is send to the sensor node over a public channel.

4.3. Authentication Phase (AP)

The authentication phase is invoked when a sensor node receives the user login request message $<M1>$ at time $Ts$. The sensor node authenticates users’ requests by the following steps:

(AP-1). The sensor node, $Sn$ validates the $Tu$: Check, if $(Ts−Tu) \geq \Delta T$, if yes, then $Sn$ reject this request and terminates the process. Otherwise, it continues with the next step. Here, $Ts$ is the current timestamp of $Sn$ and $\Delta T$ is the defined time interval for the transmission delay.
(AP-2). Now $Sn$ generates message $M2 = <B_k, AID_k, Tu, Ts, Sn>$ and $Q$. Thereafter, $Sn$ sends $<M2, Q>$ to the $GW$ node over public network. Here, a message authentication code (MAC) [3,33] is computed on the message $M2$ (i.e., $Q = MAC_{SK_g}(B_k||AID_k||Tu||Ts||Sn)$) for the integrity verification by the $GW$ node.
(AP-3). Upon receiving the message $M2$ and $Q$ from the sensor node, the $GW$ node performs the following actions:

1. The $GW$ node validates the time $Ts$: Check if $(T_g−Ts) \geq \Delta T$, if yes, then the $GW$ node rejects this request and terminates the process, otherwise, it continues with the next step. Here, $T_g$ is the current timestamp of the $GW$ node and $\Delta T$ is the defined time interval for the transmission delay.
2. The $GW$ node computes $Q'$ on the message $M2$ using long-term secret key $SK_g$ and verifies $Q' = Q$, if yes, then the $GW$ node considers that this is an original message and proceeds to the next step, otherwise, it rejects the request and terminates further operations.
3. The $GW$ node decrypts $B_k$ using secret key $x$ and obtains $ID_k'$ and $l'$ of $U_k$. Now, the $GW$ node computes $h(ID_k')$, $A_k' = h(ID_k' \oplus l')$ and $H_k' = h(A_k')$. Subsequently, the $GW$ node generates a temporary key $TkTu(Tu||H_k')$. After that, the $GW$ node decrypts the sub-message $AID_k$ by using the $TkTu$ and obtains $[h(ID_k*)||Sn*||X_g]$. Afterwards, the $GW$ node compares $h(ID_k)$ with $h(ID_k')$, if this check is successful then the user is a legal user. Here, $h(ID_k*)$ is sub-message of $AID_k$. At the same time, the $GW$ node verifies whether $Sn$ is equal to $Sn*$, which is included in
sub-message of \( AID_k \) and if yes, then the \( GW \) node considers that \( Sn \) is a legal node that user \( U_k \) wants to access. Otherwise, the \( GW \) node rejects the authentication process.

4. After authenticating the user and the sensor node; the \( GW \) node informs the sensor node that user \( U_k \) is a legitimate user, therefore, the \( GW \) node computes message \( M3 = <Tg, C> \) and sends message \( M3 \) to the sensor node. Here, \( C = E_{SKgs}[h(ID_k)||X_g||Tg] \) and \( Tg \) is the current timestamp of the \( GW \) node.

(AP-4). Upon receiving the message \( M3 \) from the \( GW \) node, the sensor node performs the following:

1. Firstly, \( Sn \) validate the time \( Tg \): Check if \( (Ts−Tg) ≥ ΔT \), if yes, then the sensor node rejects this request and terminates the process. Otherwise, it continues with the next step. Here, \( Ts \) is the current timestamp of the sensor node and \( ΔT \) is the defined time interval for the transmission delay.

2. Now, with the knowledge of \( SK_{gs} \), the sensor node decrypts the sub-message \( C = D_{SKgs}[h(ID_k)||X_g||Tg*] \) and verifies whether \( Tg \) is equal to \( Tg* \) or not. If \( Tg \) is not verified then the process is terminated; otherwise, the sensor node considers the \( GW \) node a legal node and message \( M3 \) is generated by the original \( GW \) node.

3. Thereafter, \( Sn \) computes a session key \( Ses_k = h(h(ID_k)||X_g||Sn||Ts||Tu) \) for secure communication between the sensor node and the user. Now, \( Sn \) generates a message \( M4 = <L, Ts> \), and sends it to the user \( U_k \). Here, \( L = E_{SesK}[X_g||Ts] \) and \( Ts \) is the current time stamp of the sensor node.

(AP-5). While receiving the message \( M4 \) from the sensor node, the user \( U_k \) performs the following:

1. Firstly, \( U_k \) validates the time \( Ts \): Check if \( (Tu−Ts) ≥ ΔT \), if yes, then the sensor node rejects this request and terminates the process. Otherwise, it continues with the next step. Here, \( Tu \) is the current timestamp of the user \( U_k \) and \( ΔT \) is the defined time interval for the transmission delay.

2. Now, the user \( U_k \) computes the session key \( Ses_k = h(h(ID_k)||X_g||Sn||Ts||Tu) \) and decrypts the sub-message \( L \) (i.e., \( D_{SesK}[X_g*||Ts*] \)) and obtains \( X_g* \) and \( Ts* \). The user \( U_k \) checks \( X_g = X_g* \) and \( Ts = Ts* \), if yes, then he/she believes that \( Sn \) is a real sensor node, otherwise not.

3. Moreover, now the user \( U_k \) and \( Sn \) share the symmetric session key \( Ses_k = h(h(ID_k)||X_g||Sn||Ts||Tu) \) for performing further subsequent operation during a session.

As a result, a legitimate user can communicate with real sensor nodes and access the network data.

4.4. Password-Update Phase (PUP)

The password update phase is invoked whenever user \( U_k \) wants to update his/her old password \( (PW_k) \). The password update phase is described below:

(PUP-1). User \( U_k \) inserts his/her smart card into the terminal and enters his/her identity \( (ID_k) \) and password \( (PW_k) \).

(PUP-2). Firstly, the smart card validates the \( U_k \)’s entered \( ID_k \) and \( PW_k \) with the stored values by computing the following: \( V_k = h(ID_k||h(r@PW_k)) \).
(PUP-3). Verify whether $V_k^*$ and $V_k$ are equal or not. If not, then reject the password update request; otherwise, proceed to the next steps.

(PUP-4). Upon receiving the $U_k$’s new password ($PW_{\text{new}}^*$), the smart card computes $V_{\text{new}}^* = h(ID_k || h(r \oplus PW_{\text{new}}^*))$ (PUP-5). Now, the smart card replaces $V_k$ with $V_{\text{new}}^*$.

After performing the above steps, the password update phase takes place successfully. The flow of the password update phase is shown in Figure 4.

**Figure 4.** Flow of password update phase.

| Enter ID$_k$ and PW$_k$ |
|------------------------|
| $V_k^* = h(ID_k || h(r \oplus PW_{\text{new}}^*))$ |
| Check $V_k^* = V_k$, if yes, and then go to the next step. |
| User enter new password $PW_{\text{new}}^*$ |
| $V_{\text{new}}^* = h(ID_k || h(r \oplus PW_{\text{new}}^*))$ |

Replace $V_k$ with $V_{\text{new}}^*$ in the smart card.

5. Nonmonotonic Cryptographic Protocol and RUASN Analysis and Verification

This section presents, first, the nonmonotonic cryptographic protocol (NCP) [24], and second, formal analysis and verification of RUASN using the NCP protocol.

5.1. Nonmonotonic Cryptographic Protocol

This sub-section describes the nonmonotonic cryptographic protocol (NCP), which is also known as Rubin Logic. The NCP logic assumed that involved entities are trusted, and state of knowledge (i.e., current state of the protocol), and describes the authentication logic. In [24-34] the authors state that the NCP logic does not require any idealization step in specifying the protocol and is very close to real implementation. In Rubin logic the roles are assigned to entities, which are considered as independent processes. The nonmonotonic logic consists of two sets, global set and local set. These sets are directly applicable for protocol analysis, as follows. For more details the reader may refer to [24] and [35].

5.1.1. Global Set and Local Set

Global set: The global sets are publically known to each principal in a protocol specification, and consist of principal set, rule set, secret set and observers set. These sets represent the information of the protocol and the contents of these global sets may get updated as the protocol progresses.

- **Principal set:** This set represents the involved entities in a protocol, such as, $Et = \{Et_1, Et_2, Et_3, \ldots, Et_n\}$.
- **Rule set:** This set contain inference rules for deriving new statements from existing statements, as described in Section 5.1.3.
5.1.2. Actions

Actions are operations which play essential roles in the protocol specification. These actions control the state of knowledge and possessions for involved entities (such as, constructs messages, hashing, concatenation, encryption/decryption, secret generation, update, and abort operations, are few examples). For a complete list of actions readers may refer to [24,35]. As per our requirements, we have defined the following actions, as shown below, which are directly adopted from [24-34]. The action is marked with • to show that it has been successful and moved to the next action.

- **Hash** (h(·); X)
  
  Condition: h(·), X ∈ POSS(Ei)
  
  Result: POSS(Ei): = POSS(Ei) U {h(X)}
  
  Description: This action is for hashing the data.

- **XOR** (X₁, X₂, X₃, ..., Xₙ)

  Condition: X₁, X₂, X₃, ..., Xₙ ∈ POSS(Ei)
  
  Result: POSS(Ei): = POSS(Ei) ∈ {X₁, X₂, X₃, .......Xₙ}
  
  Description: This action is for XORing the data.
- **Encrypt**(X, k)
  Condition: X, k ∈ POSS(Ei)
  Result: POSS(Ei) := POSS(Ei) ≮ {X}\k
  Description: This action occurs when a principal encrypts data. If Ei possesses X and knows k then he/she can possess \{X\}_k.
- **Decrypt**(\{X\}_k, k)
  Condition: \{X\}_k, k ∈ POSS(Ei)
  Result: POSS(Ei) := POSS(Ei) ≮ \{X\}
  Description: This action is performed when a principal decrypts data. If Ei possesses X, encrypted under k, and Ei knows k, then he/she (Ei) can possess X.
- **Generate-Secret**(X_g)
  Result: S := S ≮ \{X_g\}, Observers(X_g) = \{Ei\},
  POSS(Ei) := POSS(Ei) ≮ \{X_g, Ei\},
  BEL(Ei) := BEL(Ei) ≮ \#(X_g)
  Description: This action generates a secret for an entity, when needed. Thereafter, a new secret X_g, is added to secret S and the Observers and Possession sets are updated.
- **Concat**(X_1, X_2, X_3, ....X_n)
  Condition: X_1, X_2, X_3, ....X_n ∈ POSS(Ei)
  Result: POSS(Ei) := POSS(Ei) ≮ \{X_1, X_2, X_3, ...., X_n\}
  Description: This action concatenates the sub-messages.
- **Check**(X, Y)
  Condition: X, Y ∈ POSS(Ei)
  Result: Valid if X==Y, otherwise invalid.
- **Split**(X)
  Condition: X contains X_1, X_2, X_3, ...., X_n, X ∈ POSS(Ei)
  Result: POSS(Ei) := POSS(Ei) ≮ \{X_1, X_2, X_3, ...., X_n\}
  Description: This action is used to break the message into sub-messages.
- **Send**(Ej, X)
  Description: This action sends message X to Ej (i.e., Ei to Ej).
- **Receive**(Ej, X)
  Description: This action receives message X from Ej (i.e., Ei receives from Ej) and added to POSS(Ei).
- **Update**(X)
  Description: The purpose of update action is to update the Observers sets of all secrets that are sent on the network.
- **Forget**(X)
  Description: The purpose of forget action, when Ei no longer is in possession of X.
Description: The abort action takes place when a checked action could not satisfy the established conditions. Furthermore, this action aborts the system, if a protocol run is illegal. As a result, analysis reports a failure.

5.1.3. Inference Rules

A procedure which combines known facts and produce new fact is called an inference rule [24]. These inference rules are used to fact about beliefs during the protocol execution and are applied whenever they are relevant to protocol progress. The following inference rules are defined as per our requirements, which are directly adopted from [24,35].

Notation:

- \(X \text{ contains } Y\): \(Y\) appears as a sub-message of \(X\).
- \(S := f(S)\): \(S\) is replaced by the value of \(f(S)\).
- \(X \text{ from } E\): \(X\) is received from \(E\).

1. Message-meaning rule:

   \[
   \{X\}_{k} \text{ from } Ej \in POSS(Ei), \{Ei, Ej\} \subseteq POSS(Ei) \\
   \text{BEL}(Ei) := \text{BEL}(Ei) \cup \{X \in POSS(Ei)\}
   \]

2. Origin rule:

   \[
   X \in POSS(Ei), X \text{ contains } x_{1}, Ej \in Observers(x_{1}) \\
   x_{1} \text{ from } Ej \in POSS(Ei)
   \]

3. Sub-message origin rule:

   \[
   X \in POSS(Ei), X \text{ contains } \{x_{1}, x_{2}\} \text{ from } Ej \\
   x_{2} \text{ from } Ej \in POSS(Ei)
   \]

4. Sub-message freshness rule:

   \[
   \#(x_{1}) \in \text{BEL}(Ei), \{X \text{ contains } x_{1}, Y \text{ contains } x_{2}\} \subseteq POSS(Ei) \\
   \text{BEL}(Ei) := \text{BEL}(Ei) \cup \#(x_{1})
   \]

For a complete list of inference rules, readers may refer to [24] and [35].

5.2. RUASN Analysis and Verification Using Rubin-Logic

This sub-section presents the analysis and verification of the proposed RUASN using well-known Rubin logic [24]. The NCP logic integrates protocol analysis with protocol specification and thus, beliefs of involved entities and current state of knowledge is updated as the protocol run progresses. For RUASN analysis and verification, only three phases are accounted, namely, registration phase, login phase and authentication phase, as follows.

5.2.1. RUASN Specification

For convenience, a list of some additional notations and symbols will be used in the RUASN analysis and verification, as shown in Table 2.
As per the NCP logic, the specifications of RUASN are the following:

**Global sets:** It consists of four sets:

- **Principal set:** \( P = U, GW, \) and \( Sn \). Here, \( U \) is the initiator of RUASN.
- **Rule set:** Inference rules are defined in Section 5.1.3.
- **Secret set:** \( \{ PW_k, r, x, y, l, SK_{gs} \} \)
- **Observers set:**
  - Observers \( (PW_k, r) \): \( \{ U \} \)
  - Observers \( (x, y, l) \): \( \{ GW \} \)
  - Observers \( (SK_{gs}) \): \( \{ GW, Sn \} \)

**Local sets:** The local sets consist of \( U, GW, \) and \( Sn \) and are as shown in Table 3.

### Table 2. Additional notations.

| Notation | Description |
|----------|-------------|
| \( U, GW \) and \( Sn \) | User, Gateway and Sensor, respectively entities |
| \( XI \rightarrow X2 \) | X1 replace by X2 |
| **Phase - I** | Registration phase |
| **Phase - II** | Login phase |
| **Phase - III** | Authentication phase |

#### 1. **Entity U**

- **POSS(U)** = \( \{ PW_k, ID_k, r \} \)
- **BEL(U)** = \( \#(PW_k), \#(r) \)
- **BL(U)** = \( r \) and \( ID_k \)

(1.1) Phase – I

- Hash(h(.); XOR(r, PW_k)) –> Pass
- Send(GW, \( [ID_k, Pass] \))
- Update(\( [ID_k, PW_k, r] \))
- Receive(GW, \( [ID_k, h(.), E_k, D_k] \))

(1.2) Phase – II

- Hash(h(.); Concat(ID_k, Pass)) –> V_k*
- Check(V_k*, V_k)
- Hash(h(.); A_k) –> H_k
- Generate-secret(X_k)
- Concat(Tu, H_k) –> T_k
- Encrypt(\{ Concat(Hash(h(.);ID_k, Sn, X_k)) || Ts \}) –> AID_k
- Send(Sn, \{ B_k, AID_k, Tu \}) –> M1
- Update(ID_k, X_k, Sn, Tu)

### Table 3. Local Sets for RUASN.

| **(1.3) Phase – III** | |
|----------------------|-------------------|
| Receive(Sn, \{ L, Ts \}) –> M4 \[ Here L=E_{Sesk}[X_k][Ts] \] | |
| Split(L, Ts) | |
| Check-freshness (Tu – Ts) ≥ \( \Delta T \), if yes, then aborts | |
| Hash(h(.); Concat(Hash(h(.);ID_k, Sn, Ts, Tu)) –> SesK | |
| Decrypt(\[ L \]_Sesk and obtain (X_k*, T_s*) | |
| Check(X_k*, X_k*) | |
| Check(Ts, T_s*) | |

#### 2. **Entity GW**

- **POSS(GW)** = \( \{ x, y, l, and SK_{gs} \} \)
- **BEL(GW)** = \( \#(s), \#(y), \#(l), \#(SK_{gs}) \)
- **BL(GW)** = \( \} \)

(2.1) Phase – I

- Receive(U, \( [ID_k, Pass] \))
- Generate-secret number(l)
- Hash(h(.); XOR(ID_k, l)) –> A_k
- Encrypt(\{ Concat(ID_k, l) \}) –> B_k
- Hash(h(.); Concat(ID_k, Pass)) –> V_k
- Send(U, \{ A_k, B_k, V_k, h(.) \})
- Update(\{ A_k, B_k, V_k, h(.) Pass) | |
- Forget(\{ A_k, B_k, V_k, Pass) | |
Table 3. Cont.

| (2.2) Phase – II | (3.1) Phase – I |
|------------------|------------------|
| NA               | [Here M1 = (Bk, AIDk, Tu)] |

| (2.3) Phase – III | (3.2) Phase – II |
|-------------------|------------------|
|                  | [Here M2 = (Bk, AIDk, Tu, Ts, Sn)] |

- Receive(U, [M1])
- Split(M1)
- Check-freshness (Ts – Tu) ≥ ΔT, if yes, then aborts.
- MAC({Concat(Bk, AIDk, Tu, Ts, Sn)}SKgs) → Q
- Send(GW, [M2, Q])
- Update(M2, Q)

(3.3) Phase – III

- Receive(GW, [Tg, C])
- Split(Tg, C)
- Check-freshness (Ts – Tg) ≥ ΔT, if yes, then aborts.
- Decrypt({C}SKgs) and obtain [h(ID∗, Xg, Tg)]
- Check(Tg∗, Tg)
- Hash(h(.); Concat(Hash(.); ID∗, Xg, Ts, Tu)) → Sesk
- Encrypt({Concat(Xg, Ts)}skes) → L
- Send(U, [L, Ts]) → M4
- Update(L, Ts)

NA: Not Applicable.

5.2.2. RUASN Analysis and Verification

Once the protocol specification has been completed, the analysis begins. This subsection analyzes the proposed scheme, which is very close to real implementation. We have considered three phases, namely, registration phase, login phase and authentication phase, where three entities are involved in the protocol progress [i.e., user(U), gateway(GW) and sensor(Sn)].

As we can see the Phase-I in Table 3, the entity U is the initiator of the protocol, so user behavior list actions are executed first [i.e., BL(U)]. The first three actions are executed in BL(U) and once the Update action is performed, the next actions have to be executed in the GW behavior list, since the Send operation (i.e., Send(GW, [IDk, Pass])) is specified GW list, as shown below:

- Hash(h(.); XOR(r, PWk)) → Pass
- Send(GW, [IDk, Pass])
- Update([IDk, PWk, r])

Now GW’s Phase-I actions behavior lists take place [i.e., BL(GW)] and first seven actions are executed in BL(GW). After the Update operation, the next operations have to be executed in the user’s behavior list as below, the Send operation [Send(U, [Ae, Be, Ve, h(.), Ee, Dk])] assigned to U. Then the
Forget operation removes all values (i.e., $A_k, B_k, V_k, E_k, D_k$) from the local possession set $[POSS(GW)]$, as shown below:

- **Receive**($U, \{ID_k, Pass\}$)
- **Generate-secret number**($l$)
- **Hash**($h(.)$, XOR($ID_k, l$)) → $A_k$
- **Encrypt**($\{ Concat(ID_k, l) \}$) → $B_k$
- **Hash**($h(.)$, Concat($ID_k, Pass$)) → $V_k$
- **Send**($U, \{ A_k, B_k, V_k, h(.), E_k, D_k \}$)
- **Update**($\{ A_k, B_k, V_k, E_k, D_k, Pass \}$)
- **Forget**($\{ A_k, B_k, V_k, E_k, D_k, Pass \}$)

It is clear that after the end of the user and the gateway Phase-I, there is no significant change in their global sets, whereas, in the local sets of both entities (i.e., *user* and *gateway*) have some changes, as shown:

$POSS(U) = \{ID_k, PW_k, r\{A_k, B_k, V_k, h(.), E_k, D_k\}\}$

$BEL(U) = \{\#(PW_k)\}$

$POSS(GW) = \{x, y, l, SK_{gs}\}$

$BEL(GW) = \{\#(x), \#(y), \#(l), \#(SK_{gs})\}$

This is the end of Phase-I.

Subsequently, Phase-II begins and two entities (i.e., *User* and *Sensor*) are involved in this phase. As, $U$ is the initiator of the protocol, and thus next operations have to be executed in $U$’s behavior list ($BL(U)$), as follows:

- **Hash**($h(.)$, Concat($ID_k, Pass$)) → $V_k^*$
- **Check**($V_k^*, V_k$)
- **Hash**($h(.)$, $A_k$) → $H_k$
- **Generate-secret**($X_g$)
- **Concat**($Tu, H_k$) → $TkTu$
- **Encrypt**($\{ Concat(Hash(h(.),ID_k, Sn, X_g)) \}_{TkTu}$) → $AID_k$
- **Send**($Sn, \{ B_k, AID_k, Tu \}$) → $M1$
- **Update**($ID_k, X_g, Sn, Tu$)

After the *Update* operation, the following changes will take place in $BL(U)$.

$POSS(U) = \{ID_k, V_k, X_g, TkTu, AID_k, A_k, H_k\}$

$BEL(U) = \{\#(X_g), \#(TkTu)\}$

The global set of entity $U$ will change with the following:

$Observers(X_g): \{U\}$
The next operations have to be executed in Sn’s behavior list [BL(Sn)], because the above Send operation (i.e., Send(Sn, {Bk, AIDk, Tu}) → M1) specifies Sn. Upon receiving the Receive operation (i.e., Receive (U, {M1})), Sn does the following:

- **Receive(U, {M1})** [Here M1 = (Bk, AIDk, Tu)]
- **Split(M1)**
- **Check-freshness (Ts − Tu) ≥ ΔT**, if yes, then aborts.
- **MAC(Concat(Bk, AIDk, Tu, Ts, Sn)SKgs) → Q**
- **Send(GW, {M2, Q})** [Here M2 = (Bk,AIDk, Tu, Ts, Sn)]
- **Update(M2, Q)**

The **Update** operation makes the following changes in BL(Sn):

\[ \text{POSS(Sn)} = \{B_k, \text{AID}_k, \text{Tu}, \text{Ts}, \text{Sn}, Q, \text{SK}_{gs}\} \]
\[ \text{BEL(Sn)} = \{\#(\text{SK}_{gs})\} \]

As we can see the next operations will be executed in GW’s behavior list [BL(GW)], as the above Sn Send operation (i.e., Send(GW, {M2, Q})) specifies to GW. Upon receiving the Receive operation [i.e., Receive (Sn, {M2, Q})] from Sn. Now, BL(GW) performs the following operations:

- **Receive(Sn, {M2, Q})[Here M2 = B_k, AID_k, Tu, Ts, Sn]**
- **Split(M2)**
- **Split(M2)**
- **Check-freshness (T_g − Ts) ≥ ΔT**, if yes, then aborts.
- **MAC(\{B_k, AID_k, Tu, Ts, Sn\}SK_{gs}) → Q’**
- **Check(Q, Q’)**
- **Decrypt(B_k) and obtain [ID_k’, l’]**
- **Hash(h(·); ID_k)***
- **Hash(h(·); XOR(ID_k’, l’)) → A_k’**
- **Hash(h(·); A_k’) → H_k’**
- **Concat(Tu, H_k’) → TkTu**
- **Decrypt(AID_k\|TkTu) and obtain[h(ID_k*), Sn*, X_g]***
- **Check(h(ID_k*), h(ID_k’))**
- **Check(Sn*, Sn)**
- **Encrypt(\{Concat(Hash(h(·); ID_k), X_g, T_g)\}SK_{gs}) → C**
- **Send(Sn, { T_g, C })→ M3**
- **Update(T_g, C)**

After applying Update operation the following changes take place in the gateway entity:

\[ \text{POSS(GW)} = \{ID_k, H_k, X_g, SK_{gs}, Sn, l, x, y, } \]
\[ \{B_k, AID_k, Tu, Ts, Sn\} \text{ and } \{\text{MAC}\{B_k, AID_k, Tu, Ts, Sn\}SK_{gs}\}\text{ from Sn} \]
\[ \text{BEL(GW)} = \{\#(\text{SK}_{gs}), \#(x), \#(y), \#(l), \#(T_g)\} \]

Subsequently, the global set of entity GW will change with the following:
Observers($X_g$): \{U, GW\}

The next operations have to be executed in Phase-III of $BL(Sn)$, because the above Send operation \[i.e., \text{Send}(Sn, \{Tg, C\})\] specifies $Sn$ and upon receiving the Receive operation \[i.e., \text{Receive}(GW, \{Tg, C\})\] $Sn$ will do the following:

- **Receive**(GW, \{Tg, C\}).
- **Split**(Tg, C).
- **Check-freshness** ($Ts - Tg \geq \Delta T$), if yes, then aborts.
- **Decrypt**(\{C\}$_{SK_{gs}}$) and obtain \[h(ID_k, X_g, Tg^*)\].
- **Check**(Tg*, Tg).
- **Hash**(h(\_), Concat(Hash(\_), ID$_k$, $X_g$, Sn, Ts, Tu))→Ses$_K$.
- **Encrypt**(\{Concat($X_g$, Ts)\}$_{SesK}$)→L.
- **Send**(U, \{L, Ts\})→M4.
- **Update**(L, Ts).

After applying the **Update** operation on the received message, the following changes will occur in the $Sn$ entity:

\[
\text{POSS}(Sn) = \{ SK_{gs}, SesK, Ts,ID_k, X_g, Tg \text{ from GW} \}
\]
\[
\text{BEL}(Sn) = \#(SK_{gs}), \#(SesK), \#(X_g), \#(Ts)\]

The following changes will occur in the global set of the entity $Sn$:

Observers($X_g$): \{U, GW, Sn\}

Observers(Ses$_K$): \{Sn\}

Thereafter, the next operations have to be executed in Phase-III of $BL(U)$. Since the above Send operation \[i.e., \text{Send}(U, \{L, Ts\})\] specifies $U$ entity and $BL(U)$ perform the following actions:

- **Receive**(Sn, \{L, Ts\})→M4 \[Here L=ESesK[X_g][Ts]\]
- **Split**(\{L, Ts\})
- **Check-freshness** ($Tu - Ts \geq \Delta T$), if yes, then aborts.
- **Hash**(h(\_), Concat(Hash(\_), ID$_k$, $X_g$, Sn, Ts, Tu))→Ses$_K$
- **Decrypt**(\{L\}$_{SesK}$) and obtain ($X_g^*$, Ts*)
- **Check**($X_g$, $X_g^*$)
- **Check**(Ts, Ts*)

In the **Check** operation $U$’s verify $X_g$, which was generated by him/her and verify time-stamp Ts. If the **Check** operations are successful then the following changes will occur in $BL(U)$:

Local set:

\[
\text{POSS}(U) = \{X_g, Sn, SesK\}
\]
\[
\text{BEL}(U) = \#(X_g), \#(SesK)\]

Now finally the global set contains:
The application of Rubin logic in our framework is illustrated above, which closely resembles the structure of real user authentication system in wireless sensor network. Our specifications are designed to resemble actual implementation as much as possible.

6. Evaluation of RUASN

In this section, we present our proposed RUASN evaluation in terms of security analysis (i.e., can it resist against several well-known attacks), and efficiency analysis in terms of computational and communication cost. Finally, we show a functionality comparison with existing schemes.

Before evaluating the RUASN, it is assumed that an adversary may have full control over the network with following capabilities:

- An adversary may intercept all the messages (i.e., $M_1, M_2, M_3$ and $M_4$) at any time.
- He/she may intercept, delete or modify, and insert any message over the public network.
- In addition, we assume that an adversary may hack either passwords or steal user $U_k$’s smart card, extract secrets [36,37], but cannot do both at the same time [38].
- As per the current literature, extracting secrets from the smart card memory is quite difficult and some smart card manufacturer companies provide countermeasures against risk of side channel attacks [18,36]. Furthermore, [39] has proposed countermeasures against power analysis attacks.

Based on above assumptions, an attacker may execute certain attacks to breach the proposed RUASN scheme.

6.1. Security Analysis

In this subsection, we analyze the security of proposed RUASN and further compare with the M.L, Das [18], He et al. [21], Wong et al. [10], and Vaidya et al. [14] schemes. We prove that the presented scheme can resist certain popular attacks that are found in the existing wireless sensor network literature.

**Mutual Authentication.** Our scheme provides mutual authentication, where all entities (i.e., user, gateway and sensor node) are mutually authenticating each other. More specifically, when the GW node receives the message $M_2$ (i.e., $<B_k, AID_k, Tu, Ts, Sn>$) and $Q$, it can make sure that the user message $M_1$ (i.e., $<B_k, AID_k, Tu>$) is included in the sensor node message $M_2$. When the sensor node receives message $M_3$ (i.e., $<Tg, C>$), it ensures that this message is generated by the GW node. Furthermore, when the user receives message $M_4$ (i.e., $<L, Ts>$), he/she can also confirm that this message is generated by the sensor node. Hence, mutual authentication is achieved.

**User anonymity.** In our scheme, user anonymity $U_k$ is preserved at the registration phase by computing $A_k = h(ID_k \oplus l)$ and $B_k = E_{[ID_k]}[l]$. In addition, it is impossible to extract $ID_k$ from the $AID_k$, which is $E_TK_{U}h(ID_k)[Sn][X_g]$, and it is also very difficult to revert the $h(ID_k)$. So, our scheme can preserve user anonymity.
**Session Key Establishment.** The proposed scheme provides session key establishment after the authentication phase. A session key \([i.e., \text{Ses}_K = h(h(ID_u)||X_0||Sn||Ts||Tu)]\) is set up between the user and the sensor node for secure subsequent communication. The \(\text{Ses}_K\) will be different for each login session and cannot be replayed after the time expires. More importantly, the user and the sensor node can securely execute encryptions and decryptions by using of \(\text{Ses}_K\) and hence, achieve confidentiality for the subsequent messages.

**Confidentiality.** Our proposed scheme provides adequate confidentiality to their messages (such as, \(E_{\text{T}_{\text{Tu}}}[h(ID_u)||Sn||X_S]\), \(E_{\text{S}_{\text{gs}}}[h(ID_u)||X_S||Tg]\) and \(E_{\text{S}_{\text{sk}}}[X_S||Ts]\)). More precisely, these messages are confidential from any attacker.

**Replay Attacks.** Our scheme is resistant to replay attacks [42], because the authenticity of messages \(<\text{M}1>, <\text{M}2>, <\text{M}3> \) and \(<\text{M}4>\) are validated by checking the freshness of four timestamps \((Ts - Tu) \geq \Delta T, (Tg - Ts) \geq \Delta T, (Ts - Tg) \geq \Delta T \) and \((Tu - Ts) \geq \Delta T\). Let’s assume an intruder intercepts a login request message \(\text{M}1\) and attempt to access the sensor node by replaying the same message \((\text{M}1)\). The verification of this login attempt fails, since the time difference expires (i.e., \((Ts - Tu) \geq \Delta T\)). Similarly, if an intruder intercepts a valid message \(\text{M}2\) (i.e., \(<B_k, AID_k, Tu, Ts, Sn>\)) and attempts to replay it to the GW node, the verification request will fail at the GW node because of the time difference expires again (i.e., \((Tg - Ts) \geq \Delta T\)). Thus, our framework is secure against replaying of messages.

**User Impersonation Attacks.** An attacker cannot impersonate the user. Suppose an attacker forges a login message \(<B_k, AID_k, Tu>\). Now, he/she will again try to login into the system with the modified message \(<B'_k, AID'_k, Tu>_k\), since, the fake \(AID'_k\) will not be verified at the GW node, and the GW node cannot get the original sub-message \([h(ID'_k)||Sn']\) by decrypting \(AID'_k\). Therefore, it is not possible to impersonate the user.

**Gateway Impersonation Attacks.** As long as an attacker does not possess the secret key \(SK_{gs}\), he/she cannot impersonate the server and cannot cheat the sensor node. Hence, it frustrates attackers to generate the valid message \(\text{M}4\) to the sensor node.

**Insider Attacks.** It is possible in a real-time environment, when the gateway manager or system administrator can use the user password \(PW_k\) (e.g., weak password) to impersonate the user \(U_k\) through any other network gateways [20,21,40]. In this case, our scheme does not give any room for privileged insiders, since, in the registration phase, the user \(U_k\) is passing \(h(r@PW_k)\) instead of the plain password. Thus, the insider of the GW node cannot get \(PW_k\) easily [20,21]. Here, \(r\) is a sufficiently high entropy number, which is not revealed to the GW node. Furthermore, the proposed scheme does not store any verifier table and can resist the insider attacks [41].

**Stolen-Verifier Attacks.** The stolen-verifier attack scenario is not applicable to our scheme, as we are not using any password/verifier table.

**Offline-Password Guessing Attacks.** Our scheme is free from any password verifier table, so password guessing attacks are not feasible. In the login phase, passwords are not simply transmitted, instead, they are transmitted with some other secret (i.e., \(V_k = h(ID_u|| h(r@PW_k))\)), which makes it difficult to guess the user’s password.

**Man-in-the-Middle Attacks.** An attacker may attempt a man-in-the-middle (MIMT) attack by modifying the login message \(<B_k, AID_k, Tu>\) into \(<B'_k, AID'_k, Tu>*\). However, this malicious
attempt will not work, as the false $AID_*'$ will not be verified at the $GW$ node and the $GW$ node cannot get the original sub-message $\{h(ID_k'||Sn*)\}$ by decrypting $AID_*'$. Thus, man-in-the-middle attacks are not applicable to the RUASN scheme.

**Secure Password Update.** In the secure password update phase, our framework first verifies the old $ID_k$, $PW_k$ and only then requests a new password. Otherwise, it rejects all password update requests. Therefore, our framework updates passwords securely.

**Gateway Secret Key Guessing Attacks.** In our scheme, the gateway secret keys ($x$ and $y$) are very long and possess high entropy. In addition, neither $x$ nor $y$ are transmitted in plain text over the public channel, instead $x$ and $y$ are mainly used as a key to encrypt data ($Ex[ID_k||l], SKgs = h(Sn||y)$). Hence, it is very difficult to guess both the gateway master keys, $x$ and $y$.

6.2. Efficiency Analysis

In this subsection, we present an efficiency analysis (i.e., computational cost and communication cost) of our scheme and compare it with existing schemes (Das [18], He et al. [21], Wong et al. [10] and Vaidya et al. [14]) for wireless sensor networks. The evaluation parameters are shown below:

- $H$: performing one-way hash function.
- $S$: symmetric cryptosystem.
- $MAC$: the time for performing a MAC.

**Computation Cost:** RUASN adopts low-cost computations like a one-way hash function and symmetric cryptosystem, which is acceptable for WSNs and provides more security features with reasonable computational costs. As we can see the computation cost comparisons of our scheme and other related scheme are summarized in Table 4. It is easy to see that, in the registration phase (i.e., one-time job) our scheme requires $4H$ and one symmetric cryptosystem, whereas in [21] and [14] $6H$ and $4H$ are required, respectively. Furthermore, in the login and authentication phase the proposed scheme requires $9H$, $6S$ and $2MAC$, whereas, [18,21] and [14] require $9H$, $11H$ and $9H$, respectively. This is due to fact that in order to provide more functionality such as mutual authentication, user anonymity, message confidentiality, and secure session key establishment, more computational costs are incurred.

| Schemes            | Registration | Login and Authentication |
|--------------------|--------------|--------------------------|
|                    | User | Gateway | User | Gateway | Sensor node | Sensor node |
| Das [18]           | -    | 3H      | 4H  | 4H      | 1H           |
| Daojing et al. [21]| 1H   | 5H      | 5H  | 5H      | 1H           |
| Wong et al. [10]   | -    | 3H      | -   | 1H      | 3H           |
| Vaidya et al. [14] | 2H   | 2H      | 3H  | 3H      | 3H           |
| Proposed RUASN     | 1H   | 3H + 1S | 4H + 2S | 4H + 2S + 1MAC | 1H + 2S + 1MAC |

**Communication Cost:** It is easy to visualize from Figure 3 that RUASN requires four message exchanges for the whole communication and confirmation of all entities (i.e., user, gateway and sensor), which is practical for real-time applications.
6.3. Functionality Analysis

From Table 5, it is easy to see that the RUASN has more security functionality as compared to other existing proposed protocols for WSNs. Our scheme has robust security features such as mutual authentication between all entities (i.e., user, gateway and sensor), user anonymity, confidentiality, secure session key establishment, secure password update phase and secure against insider attacks, and it meets all the requirements of Liao et al. [27], which are discussed in Section 3.

Table 5. Functionality comparison of RUASN with existing schemes.

| Security Features                        | Das [18] | He et al. [21] | Wong et al. [10] | Vaidya et al. [14] | Proposed RUASN |
|------------------------------------------|----------|----------------|------------------|--------------------|----------------|
| Provides mutual authentication           | No       | No             | No               | Yes                | Yes            |
| Provide user privacy                     | No       | Yes            | No               | Yes                | Yes            |
| Confidentiality                          | No       | No             | No               | No                 | Yes            |
| Secure Session key agreement             | No       | No             | No               | No                 | Yes            |
| Secure password update phase             | No       | Yes            | No               | Yes                | Yes            |
| Replay attack                            | Yes      | Yes            | Yes              | Yes                | Yes            |
| No password tables stored inside the gateway | Yes   | Yes            | No               | No                 | Yes            |
| No verification table stored inside the gateway | Yes   | Yes            | No               | No                 | Yes            |
| Password is not be transmitted as plaintext | No    | Yes            | No               | Yes                | Yes            |
| Resist insider-attacks                   | No       | Yes            | No               | Yes                | Yes            |
| Password is not exposed to the gateway administrator | No  | Yes            | No               | No                 | Yes            |
| Secure against gateway secret key guessing attack | No  | No             | No               | No                 | Yes            |
| Secure against password guessing attack  | No       | Yes            | No               | Yes                | Yes            |

As we have seen in the above analysis, it is clear that the RUASN is a robust user authentication protocol and provides more security services at less cost.

7. Conclusions

In real-time, as the sensor networks themselves offer services to users; it is necessary to control who is accessing the information and if he/she allowed to do so. Therefore, access control is an
imperative requirement for wireless sensor networks to protect the data access from unauthorized parties.

In this regard, we have proposed a robust user authentication framework for wireless sensor networks, RUASN, which is based on a two-factor approach (i.e., password and smart card) by exploiting the advantages of cryptographic hash functions and cryptosystems. We have shown a security analysis and performance analysis of the RUASN framework and compared it with recent existing schemes. Through analysis, we show that our scheme is more robust against many popular attacks, which are prominent risks for wireless sensor network and that it provides many security services (i.e., mutual authentication, user anonymity, confidentiality, secure session key and allow users to choose/updates their password ) at reasonable computational costs.

Acknowledgments

For the fourth author, this work was supported by 2010 National Research Foundation of Korea, and for the fifth author, this work was supported by Dongseo Frontier 2009 Project and 2010 National Research Foundation of Korea.

References

1. Akyildiz, I.F.; Su, W.; Sankarasubramaniam, Y.; Cayirci, E. A survey on sensor network. IEEE Comm. Mag. 2002, 40, 102-114.
2. Perrig, A.; Szewczyk, R.; Web, V.; Culler, D.; Tygar, J.D. SPINS: Security protocol for sensor networks. In Proceeding of the 7th Annual International Conference on Mobile Computing and Networks (MOBBICOM 2001), Rome, Italy, July 2001.
3. Karlloff, C.; Sastry, N.; Wagner, D. TinySec: A link layer security architecture for wireless sensor networks. In Proceeding of the 2nd ACM Conference on Embedded Networked Sensor System (SenSys 2004), Baltimore, MD, USA, November 2004.
4. Watro, R.; Kong, D.; Cuti, S.F.; Gardiner, C.; Lynn, C.; Kruus, P. TinyPK: Securing sensor networks with public key technology. In Proceedings of the 2nd ACM Workshop on Security of ad hoc and Sensor Networks, SASN ’04, Washington, DC, USA, 25 October 2004.
5. Luk, M.; Mezzour, G.; Perrig, A.; Gligor, V. MiniSec: A secure sensor network communication architecture. In Proceeding of the 6th International Conference on Information Processing in Sensor Networks, IPSN’07, Cambridge, MA, USA, 25–27 April 2007.
6. Kumar, P.; Cho, S.; Lee, D.S.; Lee, Y.D.; Lee, H.J. TriSec: A secure data framework for wireless sensor networks using authenticated encryption. Int. J. Marit. Inf. Commun. Sci. 2010, 8, 129-135.
7. Karlof, C.; Wagner, D. Secure routing in wireless sensor networks: Attacks and countermeasure. Ad Hoc Netw. 2003, 293-315.
8. Benenson, Z.; Gartner, F.; Kesdogan, D. User Authentication in sensor network (extended abstract). In Proceedings of the Informatik 2004, 34. Jahrestagung der Gesellschaft fur Informatik, Workshop on Sensor Networks, Ulm, Germany, September 2004.
9. Benenson, Z.; Gedicke, N.; Raivio, O. Realizing robust user authentication in sensor networks. In Proceedings of the Workshop on Real-World Wireless Sensor Network (REALWSN’05), Stockholm, Sweden, 20–21 June 2005.
10. Wong, K.H.M.; Zheng, Y.; Cao, J.; Wang, S. A dynamic user authentication scheme for wireless sensor networks. In *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC’06)*, Taichung, Taiwan, 5–7 June 2006.

11. Tseng, H.R.; Jan, R.H.; Yang, W. An improved dynamic user authentication scheme for wireless sensor networks. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM’07)*, Washington, DC, USA, 26–30 November 2007; pp. 986-990.

12. Lee, T.H. Simple dynamic user authentication protocols for wireless sensor networks. In *Proceedings of the 2nd International Conference on Sensor Technologies and Application (SENSORCOMM’08)*, Cap Esterel, France, 25–31 August 2008; pp. 657-660.

13. Ko, L.C. A novel dynamic user authentication scheme for wireless sensor networks. In *Proceeding of the IEEE International Symposium on Wireless Communication Systems 2008, ISWCS’08*, Reykjavik, Iceland, 21–24 October 2008; pp. 608-612.

14. Vaidya, B.; Rodrigues, J.J.P.C.; Park, J.H. User authentication schemes with pseudonymity for ubiquitous sensor network in NGN. *Int. J. Commun. Syst.* **2010**, 23, 1201-1222.

15. Banerjee, S.; Mukhopadhyay, D. Symmetric key based authenticated querying in wireless sensor networks. In *Proceedings of the 1st ACM International Conference on Integrated Internet Ad hoc and Sensor Networks (InterSense)*, New York, NY, USA, June 2002; pp. 1278-1287.

16. Yoon, S.J.; Lee, H.; Ji, S.B.; Kim, K. A user authentication scheme with privacy protection for wireless sensor networks. In *Proceedings of the 2nd Joint Workshop on Information Security, Tokyo, Japan, 6–7 August 2007*; pp. 233-244.

17. Li, C.T.; Lee, C.C. A novel user authentication and privacy preserving scheme with smart cards for wireless communications. *Math. Comput. Modelling* **2011**, doi:10.1016/j.mcm.2011.01.010.

18. Das, M.L. Two-factor user authentication in wireless sensor networks. *IEEE Trans. Wireless Comm.* **2009**, 8, 1086-1090.

19. Nyang, D.H.; Lee, M.K. *Improvement of Das’s Two-Factor Authentication Protocol in Wireless Sensor Networks*. Available Online: [http://eprint.iacr.org/2009/631.pdf/](http://eprint.iacr.org/2009/631.pdf/) (accessed on 07 July 2010).

20. Khan, M.K.; Alghathbar, K. Cryptanalysis and security improvement of ‘two-factor user authentication in wireless sensor networks’. *Sensors* **2010**, 10, 2450-2459.

21. He, D.; Gao, Y.; Chan, S.; Chen, C.; Bu, J. An enhanced two-factor user authentication scheme in wireless sensor networks. *Int. J. Ad-Hoc Sensor Wirel. Netw.* **2010**, 0, 1-11.

22. He, D.J.; Mab, M.D.; Zhang, Y.; Chen, C.; Bu, J.J. A strong user authentication scheme with smart cards for wireless communications. *Comput. Commun.* **2011**, 34, 367-374.

23. RSA. Available Online: [http://www.rsa.com/node.aspx?id=1156](http://www.rsa.com/node.aspx?id=1156) (accessed on 12 August 2010).

24. Rubin, A.D.; Honeyman, P. Nonmonotonic cryptographic protocols. In *Proceedings of the Computer Security Foundation Workshop VII*, Franconia, NH, USA, June 1994; pp. 100-116.

25. MICAz. Available Online: [http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf](http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf) (accessed on 10 January 2010).

26. TelosB Datasheet. Available Online: [http://www.willow.co.uk/TelosB_Datasheet.pdf/](http://www.willow.co.uk/TelosB_Datasheet.pdf/) (accessed on 12 January 2010).

27. Liao, I.E.; Lee, C.C.; Hwang, M.S. A password authentication scheme over insecure networks. *J. Comput. Syst. Sci.* **2006**, 72, 727-740.
28. Diffie, W.; Hellman, M.E. New directions in Cryptography. IEEE Trans. Inform. Theory 1976, IT-22, 644-654
29. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. Comm. ACM 1978, 21, 120-126.
30. National Institute of Standards and Technology, FIPS PUB 180-1, Secure Hash Standard. US Department of Commerce: Gaithersburg, MD, USA. Available Online: http://www.techheap.com/cryptography/hash/fip180-1.pdf/ (accessed on 9 August 2009).
31. Capkun, S.; Cagalj, M.; Karame, G.; Tippenhauer, N.O. Integrity Regions: Authentication through Presence in Wireless Networks. IEEE Trans. Mob. Comput. 2010, 9, 1608-1621.
32. Sullivan, B. Preventing a Brute Force or Dictionary Attack: How to Keep the Brutes Away from your Loot. Available Online: http://h71028.www7.hp.com/ERC/cache/568358.pdf/ (accessed on 21 February 2010).
33. Bellare, M.; Kilian, J.; Rogaway, P. The security of the cipher block chaining message authentication code. J. Comput. Syst. Sci. 2000, 61, 362-399.
34. Vaidya, B.; Park, J.H.; Yeo, S.S.; Rodrigues, J.J.P.C. Robust one-time password authentication scheme using smart card for home network environment. Comput. Commun. 2011, 34, 326-336.
35. Das, M.L.; Narasimhan, V.L. Towards a formal verification of an authentication protocol using non-monotonic logic. In Proceedings of the 5th International Conference on Information Technology: New Generation, CA, USA, 2008; pp. 545-550.
36. Kocher, P.; Jaffe, J.; Jun, B. Differential power analysis. In Proceeding of the Advances in Cryptology (CRYPTO’99), Santa Barbara, CA, USA, 15–19 August 1999; pp. 388-397.
37. Messerges, T.S.; Dabbish, E.A.; Sloan, R.H. Examining Smart-card security under the threat of Power Analysis attacks. IEEE Trans. Comput. 2002, 51, 541-552.
38. Yang, G.; Wong, D.S.; Wang, H.; Deng, X. Two-factor mutual authentication based on smart card and passwords. J. Comput. Syst. Sci. 2008, 74, 1160-1172.
39. Popp, T.; Oswald, E.; Mangard, S. Power analysis attacks and countermeasures. IEEE Des. Test. Comput. 2007, 24, 535-543.
40. Ku, W.C.; Chen, C.M.; Lee, H.L. Cryptanalysis of a variant of peyravian-zunic’s password authentication scheme. IEICE Trans. Commun. 2003, E86-B, 1682-1684.
41. Ku, W.C.; Chen, S.M. Weakness and improvements of an efficient password based remote user authentication scheme using smart cards. IEEE Trans. Consum. Electron. 2004, 50, 204-207.
42. Chang, C.C.; Lee, C.Y.; Chiu, Y.C. Enhanced Authentication scheme with anonymity for roaming service in global mobility networks. Comput. Commun. 2009, 32, 611-618.

© 2011 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/3.0/).