Cryptanalysis of Kumar et al.'s Authentication Protocol for Wireless Sensor Networks

Sewan Ha, Jihyeon Ryu, Hyoungshick Kim, Dongho Won and Youngsook Lee

Abstract Wireless sensor networks are popularly used for many applications to monitor environmental changes and track events. In 2019, Kumar et al. proposed a secure and efficient authentication scheme for coal mine monitoring using wireless sensor networking technology. In this paper, we analyze the security issues of Kumar et al.'s authentication scheme. We found that Kumar et al.'s has four security weaknesses: (1) sensor nodes' critical information can be disclosed; (2) session keys can be compromised; (3) user impersonation is possible, and (4) users' identity and password can be leaked with a stolen smart card.

Keywords Wireless sensor network · User authentication · Key agreement · Sensor network

1 Introduction

Wireless sensor networks (WSNs) consist of many compact, low-powered, autonomous sensor nodes and collect environmental and physical data from the surrounding environment. With the deployment of Internet of Things (IoT) applications, WSNs have received more attention recently. WSNs are particularly useful for industrial control applications (e.g., smart city, smart factory, and smart grid). Also, WSNs would be used in hazardous environments (e.g., ocean and coal mine) for safety monitoring because people can periodically check the status of hazardous environments with environmental data collected from wireless sensors that are deployed in such an environment.

Unsurprisingly, it is important to provide the integrity of sensor outputs because modified sensor data may cause significant financial losses and/or affect human

S. Ha (✉) · J. Ryu · H. Kim · D. Won
Sungkyunkwan University, 2066, Seoburo, Suwon, Gyeonggido 16419, Korea
e-mail: hsewan@security.re.kr

Y. Lee
Howon University, Gil, Impi-Myeon, 64 3, Gunsan-Si, Jeonrabuk-Do 54058, Korea
e-mail: ysooklee@howon.ac.kr

K. J. Kim and H.-Y. Kim (eds.), *Information Science and Applications*,
Lecture Notes in Electrical Engineering 621,
https://doi.org/10.1007/978-981-15-1465-4_34

safety. Therefore, an important issue is to provide a secure user authentication mechanism to prevent unauthorized access to sensor nodes. There have been several attempts to provide secure and efficient user authentication schemes [1, 2]. Recently, Kumar et al. [2] also proposed an authentication protocol for WSNs, looking promising. However, we found that Kumar et al.'s authentication protocol has four vulnerabilities. First, sensor nodes' authentication information can be misused. Second, the attacker can compromise all session keys between the user, gateway, and sensor node. Third, the scheme cannot resist user impersonation attack. Last, from a stolen smart card, the user's identification information can be leaked.

1.1 Threat Model

In this paper, we assume that an attacker is a Dolev-Yao attacker [3]. In particular, the attacker has the following capabilities:

1 The attacker can intercept all messages transmitted through public channels.
2 The attacker can modify, delete, and resend the intercepted messages.
3 (Section 4.4 Smart card loss attack) Devices are not tamper-resistant, so the attacker can extract all parameters in the device by applying side-channel attacks.
4 (Section 4.4 Smart card loss attack) User's identity and password are weak secrets, so the attacker can derive them if there are no other values to be changed except for identity or password.

1.2 Organization of the Paper

The remainder of the paper is organized as follows. In Sect. 2, we discuss various related studies about authentication schemes for WSNs. In Sect. 3, we review the Kumar et al.'s scheme in detail. Furthermore, we do cryptanalysis on Kumar et al.'s scheme and demonstrate four vulnerabilities of the scheme in Sect. 4. Lastly, we conclude our study and provide future works of this paper in Sect. 5.

2 Related Work

An authentication for WSNs has begun with Wong et al.'s scheme [4]. In 2006, Wong et al. introduced a symmetric-key based lightweight authentication for WSNs. However, in 2009, Das [5] showed that Wong et al.'s scheme has security weaknesses such as same login identity, replay, and stolen-verifier attacks; thus, Das proposed an enhanced version of Wong et al.'s. Unfortunately, Huang et al. [6] found that

Das' scheme is susceptible to the sensor node impersonation and many logged-in-users attacks, so they proposed an improved version of Das'. However, He et al. [7] reported that Huang et al.'s scheme has impersonation and privileged insider attacks besides the absence of a password change phase. He et al. proposed a new scheme, but Vaidya et al. [8] found security weaknesses of the scheme and presented a two-factor authentication scheme with key agreement for WSNs. In 2014, Kim et al. [9] discussed the security problems of Vaidya et al.'s scheme such as user impersonation and gateway node bypass attacks, and they introduced an improved version of Vaidya et al.'s scheme. Unfortunately, in 2015, Chang et al. [10] showed that Kim et al.'s scheme has security weaknesses (e.g., man-in-the-middle attack, impersonation attack), so they proposed a new two-factor authentication using dynamic identities. However, Park et al. [11] and Jung et al. [12] proved that Chang et al.'s scheme has vulnerabilities and presented new authentication schemes to overcome the weaknesses. In 2019, Shin et al. [13] pointed out that Jung et al.'s scheme is vulnerable to tracing, information leakage, and session key attacks.

In 2016, Kumari and Om [1] proposed an authentication scheme for WSNs for safety monitoring in coal mines. Unfortunately, in 2019, Kumar et al. [2] found that Kumari and Om's scheme is vulnerable to smart card loss, stolen verifier, and denial of service attacks. As a result, they introduced an improved version of Kumari and Om's scheme; however, we found that Kumar et al.'s scheme still has four vulnerabilities such as sensor node's critical information leakage, session key compromise, user impersonation attack, and smart card loss attack.

3 Review of Kumar et al.'s Scheme

In this section, we review the Kumar et al.'s. [2] scheme. The scheme is two-factor key agreement authentication scheme, and there are four entities: registration center, user, gateway, and sensor node. A registration center involves only when a gateway and sensor nodes are set up, and when a user registers. During login and authentication phase, a user logs into a gateway using his/her smart card; then, the gateway and the sensor nodes verify the user using timestamp and secret parameters. Comprehensive notations are included in Table 1.

3.1 Sensor-Gateway Node Registration Phase

A registration center RC deploys gateways GW_j and sensors SN_k as follows:

1. RC chooses MSK_{RC} as a master secret key and assigns an identity GID_j for GW_j.
2. RC gets GW_j's hashed identity by calculating $HGID_j = h(GID_j \| MSK_{RC})$.

Table 1 Notations used in Kumar et al.'s scheme

Notation	Description
RC	A registration center
U_i	A user
GW_j	A gateway
SN_k	A sensor node
MSK_{RC}	A registration center's master secret key
GID_j	GW_j's identity
SID_k	SN_k's identity
ID_i	U_i's identity
PW_i	U_i's password
SK_{ijk}	A session key between U_i, GW_j and SN_k
V_i	Secret nonces for U_i''s authentication
l	The length of fuzzy verifier
$h(\cdot)$	An one-way hash function
$\oplus$, $\|$	Xor and Concatenation operation
$\mathcal{A}$	An attacker

3. RC selects SN_k's identity SID_k and computes the hashed identity $HSID_k = h(SID_k \| MSK_{RC})$.
4. RC stores the gateway GW_j's authentication information and the connected sensors SN_k's authentication information $\{GID_j, HGID_j, SID_k, HSID_k\}$ in GW_j and deploys GW_j in a desirable place.
5. Similarly, RC saves the sensor SN_k's information $\{SID_k, HSID_k\}$ in SN_k, and sets the sensor node SN_k at appropriate place.

3.2 User Registration Phase

In case a user U_i registers to the gateway GW_j, the following steps will proceed:

1. U_i chooses his/her identity ID_i and password PW_i and picks a nonce b_i. Then, U_i calculates hashed identity $HID_i = h(ID_i \| b_i)$ and hashed password $HPW_i = h(PW_i \| b_i)$, and transmits $\{HID_i, HPW_i\}$ to GW_j via a private channel.
2. After receiving the U_i's registration request, GW_j chooses a integer l between 2^4 and 2^8. After that, GW_j generates a nonce V_i and computes $A_{ij} = h(V_i \| HGID_j) \oplus h(HID_i \| HPW_i)$, $B_{ij} = V_i \oplus h(HGID_j)$.
3. GW_j puts parameters and hash function $\{A_{ij}, h(\cdot), B_{ij}, l\}$ into the smart card SC that will be U_i's; then, GW_j securely delivers the smart card to U_i.
4. After obtaining smart card from GW_j, U_i computes and stores the following parameters into the smart card:

$$HB_i = h(HPW_i \| HID_i \| b_i) \mod l$$
$$c_i = b_i \oplus h(ID_i \| PW_i) \mod l$$

5. The U_i's smart card will contain $\{A_{ij}, h(\cdot), B_{ij}, l, HB_i, c_i\}$.

3.3 Login Phase

When a user U_i wants to log into GW_j, U_i performs the following steps:

1. U_i inputs his/her identity ID_i and password PW_i when inserting his/her smart card SC into a card reader.
2. SC computes the following parameters which are needed for the login process:

$$b_i = c_i \oplus h(ID_i \| PW_i) \mod l$$
$$HID_i = h(ID_i \| b_i)$$
$$HPW_i = h(PW_i \| b_i)$$
$$HB_i' = h(HPW_i \| HID_i \| b_i) \mod l$$

3. SC compares HB_i' with the stored parameter HB_i. If two parameters are different, SC terminates the login session. Otherwise, SC generates a nonce R_i.
4. SC calculates the following parameters and transmits $\{L, V_2, V_3, T_1\}$ to GW_j via a public channel:

$$L = B_{ij} \oplus T_1,$$

where T_1 is a current timestamp

$$V_1 = A_{ij} \oplus h(HID_i \| HPW_i)$$
$$V_2 = h(T_1 \| R_i \| V_1)$$
$$V_3 = (R_i \| T_1) \oplus V_1$$

3.4 Authentication and Key Agreement Phase

After receiving login request $\{L, V_2, V_3, T_1\}$ from the user U_i, the gateway GW_j and the sensor node SN_k perform the following steps to establish mutual authentication between U_i, GW_j, and SN_k:

1. When GW_j obtains the login request $\{L, V_2, V_3, T_1\}$ from U_i, GW_j computes the following equations and gets V_i', V_1', and R_i:

$$V_i' = L \oplus h(HGID_j) \oplus T_1$$
$$V_1' = h\left(V_i' \| HGID_i\right)$$
$$\left(R_i' \| T_1'\right) = V_1' \oplus V_3$$

2. GW_j checks the validity of T_1 by checking $T_1' = T_1$. Moreover, GW_j checks the freshness of T_1. If the timestamp T_1 is not valid or not fresh, GW_j terminates the authentication session. Otherwise, GW_j calculates $V_2' = h\left(T_1 \| R_i' \| V_1'\right)$.
3. GW_j checks whether two parameters V_2 and V_2' are equal or not. If two parameters are the same, GW_j can infer that U_i is authenticated; otherwise, GW_j terminates the authentication session.
4. GW_j generates a nonce R_j and calculates the following parameters:

$$C_1 = h\left(SID_k \| V_i' \| HSID_k \| R_j \| T_2\right),$$
$$C_2 = \left(R_i' \| R_j \| T_2\right) \oplus HSID_k$$
$$C_3 = V_i' \oplus h\left(SID_k \| h(R_j) \| R_i'\right)$$

where T_2 is a current timestamp
5. GW_j sends the message $\{C_1, C_2, C_3\}$ to SN_k.
6. When SN_k receives the authentication message $\{C_1, C_2, C_3\}$ from GW_j, SN_k computes $\left(R_i'' \| R_j' \| T_2'\right) = C_2 \oplus HSID_k$ and obtains R_i'', R_j', and T_2'.
7. SN_k validates the freshness of the authentication message using calculated timestamp T_2'. If T_2' fails the freshness test, SN_k terminates the authentication session; otherwise, SN_k computes the following parameters:

$$V_i' = C_3 \oplus h\left(SID_k \| h(R_j') \| R_i''\right)$$
$$C_1' = h\left(SID_k \| V_i' \| HSID_k \| R_j' \| T_2'\right)$$

8. SN_k checks $C_1' = C_1$. If C_1' and C_1 differ, SN_k terminates the authentication session. Otherwise, SN_k can infer that G_j is validated.
9. SN_k selects a nonce R_K, computes parameters, and transmits $\{D_1, D_2\}$ to GW_j:

$$SK_{ijk} = h\left(R_i'' \| R_j' \| R_k\right)$$
$$D_1 = h\left(T_3 \| R_k \| SK_{ijk} \| HSID_k \| SID_k \| T_2'\right)$$
$$D_2 = (R_k \| T_3) \oplus R_j'$$

10. After receiving the message from SN_k, GW_j calculates $\left(R_k' \| T_3'\right) = D_2 \oplus R_j$ and obtains R_k' and T_3'. After that, GW_j performs a freshness test on T_3'. If T_3' is not fresh, GW_j terminates the authentication session.
11. GW_j computes the following parameters and compares D_1' with D_1:

$$SK_{ijk} = h\left(R_i' \| R_j \| R_k'\right)$$

$$D_1' = h\left(T_3' \| R_k' \| SK_{ijk} \| HSID_k \| SID_k \| T_2\right)$$

12. If they are different, GW_j terminates the session; otherwise, SN_k is verified.
13. GW_j calculates the following parameters and sends $\{C_1, C_4, C_5\}$ to U_i:

$$C_4 = h\left(SK_{ijk} \| R_j \| T_4 \| C_1\right),$$
$$C_5 = \left(R_k' \| R_j \| T_4\right) \oplus R_i'$$

where T_4 is current timestamp
14. After acquiring the authentication message $\{C_1, C_4, C_5\}$ from GW_j, U_i's smart card SC computes $\left(R_k'' \| R_j' \| T_4'\right) = C_5 \oplus R_i$ and obtains R_k'', R_j', and T_4'. Then, SC checks the freshness of T_4'. If T_4' is not fresh, SC halts the authentication process; otherwise, SC proceeds to the next step.
15. SC calculates the following parameters and checks $C_4' = C_4$:

$$SK_{ijk} = h\left(R_i \| R_j' \| R_k''\right)$$

$$C_4' = h\left(SK_{ijk} \| R_j' \| T_4' \| C_1\right)$$

16. Lastly, if C_4' and C_4 are equal, SC can prove both GW_j and SN_k are verified; otherwise, SC stops the process. Now, U_i, GW_j, and SN_k can mutually authenticate and securely communicate using the session key $SK_{ijk} = h\left(R_i \| R_j \| R_k\right)$.

3.5 Password Change Phase

When a user U_i wants to change his/her password, U_i performs the following process:

1. After inserting his/her smart card into the card reader, U_i inputs his/her identity ID_i and password PW_i.
2. SC calculates the following parameters:

$$b_i = c_i \oplus h(ID_i \| PW_i) \mod l$$
$$HID_i' = h(ID_i \| b_i)$$
$$HPW_i' = h(PW_i \| b_i)$$
$$HB_i' = h\left(HPW_i' \| HID_i' \| b_i\right) \mod l$$

3. SC checks if HB_i' is the same as HB_i, which is stored in SC. If both parameters are different, SC terminates the session; otherwise, SC proceeds to the next step.
4. SC requests a new PW_{new} and a nonce b_{new}; then, U_i inputs PW_{new} and b_{new}.
5. SC computes the following new parameters:

$$HID_{new} = h(ID_i \| b_{new})$$

$$HPW_{new} = h(PW_{new} \| b_{new})$$

$$HB_{new} = h(HPW_{new} \| HID_{new} \| b_{new}) \mod l$$

$$A_{new} = A_{ij} \oplus h(HID_i \| HPW_i) \oplus h(HID_{new} \| HPW_{new})$$

$$c_{new} = b_{new} \oplus h(ID_i \| PW_{new}) \mod l$$

6. Finally, *SC* replaces $HB_i \leftarrow HB_{new}$, $c_i \leftarrow c_{new}$, and $A_{ij} \leftarrow A_{new}$.

3.6 Sensor Node Addition Phase

Whenever a new sensor node SN_k is required to be added after setting a WSNs, the registration center *RC* follows the steps listed below:

1. *RC* chooses SN_k's identity SID_k and calculates the hashed identity $HSID_k = h(SID_k \| MSK_{RC})$; then, *RC* stores $\{SID_k, HSID_k\}$ in SN_k's memory and deploys it at the appropriate position.
2. *RC* sends SN_k's identification information $\{SID_k, HSID_k\}$ to the closest gateway GW_j for registration.
3. After receiving $\{SID_k, HSID_k\}$, GW_j saves the parameters in its database.

4 Cryptanalysis of Kumar et al.'s Scheme

In this section, we analyze Kumar et al.'s scheme and prove that the scheme is still vulnerable. Overall, the scheme has four security weaknesses: critical information leakage, session key compromise, user impersonation, and smart card loss attack.

4.1 Critical Information Leakage

In Kumar et al.'s scheme, sensors nodes' secret parameter can be leaked. Specifically, an attacker $\mathcal{A}$ can collect hashed identity $HSID_k$ of all sensors SN_k connected to a gateway GW_j and determine SN_k from the parameter C_2 using the collected $HSID_k$ and timestamp T_1. The attack progress is as follows:

1. Let suppose the attacker $\mathcal{A}$ is registered user and sends his/her login request $\{L, V_2, V_3, T_1\}$ to the victim gateway GW_j with his/her nonce R_i.
2. During authentication phase, $\mathcal{A}$ overhears $\{C_1, C_2, C_3\}$ between GW_j and SN_k.
3. After authentication process, $\mathcal{A}$ can acquire GW_j and SN_k's nonces R_j and R_k.
4. $\mathcal{A}$ computes the possible hashed identity $HSID'_k = C_2 \oplus \left(R_i \| R_j \| T_2^*\right)$ using R_i, R_j and T_2^*, where T_2^* is T_2 candidate, and T_2 is the next timestamp of T_1 $(T_1 \le T_2^* \le T_1 + \Delta T)$.
5. $\mathcal{A}$ resends another his/her login request $\{L', V_2', V_3', T_1'\}$ to GW_j, and eavesdrops the authentication message $\{C_1', C_2', C_3'\}$ between GW_j and SN_k'.
6. Same as step 3, using R_i' and the obtained R_j', $\mathcal{A}$ can partially determine whether SN_k' and SN_k are different by checking prefix of $HSID''_k (= C_2' \oplus (R_i' \| R_j' \| 0 \cdots 0)$, where $0 \ldots 0$ is the same length as T_2. If SN_k and SN_k' are expected to be the same, $\mathcal{A}$ calculates other possible $HSID'''_k = C_2' \oplus (R_i' \| R_j' \| T_2^{**})$ using R_i', R_j' and T_2^{**}.
7. $\mathcal{A}$ can get potential $HSID_k^*$ for SN_k by intersecting $\{HSID'_k\}$ and $\{HSID'''_k\}$.
8. After repeating the process several times, $\mathcal{A}$ can find a desirable $HSID_k^*$ of SN_k.

Note (1) Timestamp T_2 can be inferred easily because the base timestamp T_1 is known, and the maximum transmission delay (ΔT) is usually small. If ΔT is big, then the system cannot resist against a replay attack, so the system has to set ΔT small; thus, the number of T_2^* candidates is very small, which can be easily predicted.

4.2 Session Key Compromise

Kumar et al.'s scheme is vulnerable to session key compromise attack. Especially, if the length of the data type is fixed, an attacker $\mathcal{A}$ can easily forge the session key.

4.2.1 The Length of Variable Is the Same

Usually, nonces are set to 32-bit or 64-bit. However, if we set R_i, R_j, and R_k to the same bits, crucial information is leaked, and an attacker can use this information to compromise session key. This is because a concatenation operation increases the parameter's size. The detailed attack trace is described as follows:

1. The attacker $\mathcal{A}$ overhears overall login request of U_i; then, $\mathcal{A}$ can acquire at least $\{L, V_2, V_3, T_1, C_1, C_4, C_5\}$.
2. Since all nonces have fixed length, C_5 can be interpreted as $C_5 = (R_k \| R_j \| T_4) \oplus R_i = (R_k \| R_j \| T_4 \oplus R_i)$. This is because the timestamp is usually 32-bit or 64-bit unsigned integer. Therefore, $\mathcal{A}$ can acquire R_k and R_j very easily.
3. Let assume $\mathrm{T_r} = T_4 \oplus R_i$. Now, $\mathcal{A}$ can efficiently calculate R_i because $T_1 \le T_4 \le T_1 + 3\Delta T$, where ΔT is the maximum transmission delay. For each T_4 candidate T_4', $\mathcal{A}$ can test whether $h\big(h(\mathrm{T_r} \oplus T_4' \| R_j \| R_k) \| R_j \| T_4' \| C_1\big) = C_4$. If $\mathcal{A}$ finds a desirable T_4', $\mathcal{A}$ can derive R_i by computing $R_i = \mathrm{T_r} \oplus T_4'$.

4. Finally, $\mathcal{A}$ can forge a session key $SK_{ijk} = h\left(R_i \| R_j \| R_k\right)$ between U_i, GW_j, and SN_k, so $\mathcal{A}$ can eavesdrop all secret messages among them by using the forged session key.

4.2.2 The Length of Variable Is Set Properly

Even if all nonces' sizes (R_i, R_j, R_k) are set to the appropriate length, the attacker $\mathcal{A}$ can obtain these nonces according to the following attack method:

1. Through the same method as the Sect. 4.1, the attacker $\mathcal{A}$ obtains SN_k's hashed id $HSID_k$; then, $\mathcal{A}$ overhears the login request between U_i, GW_j, and SN_k.
2. $\mathcal{A}$ acquires T_2' by calculating $\left(R_i' \| R_j' \| T_2'\right) = C_2 \oplus HSID_k$, where $HSID_k$ is candidate of the possible hashed ID of SN_k. By checking $T_1 \leq T_2' \leq T_1 + \Delta T$, $\mathcal{A}$ finds a desirable R_i' and R_j'.
3. Then, $\mathcal{A}$ can find R_k by computing $\left(R_k \| R_j' \| T_4\right) = C_5 \oplus R_i'$.
4. Finally, with the computed R_i', R_j', and R_k, $\mathcal{A}$ forges a session key $SK_{ijk} = h\left(R_i' \| R_j' \| R_k\right)$ between U_i, GW_j, and SN_k, so $\mathcal{A}$ can eavesdrop all secret messages among them by using the forged session key.

4.3 User Impersonation Attack

During Sect. 4.2, after obtaining U_i's nonce R_i, $\mathcal{A}$ can fabricate a fake login request $\{L', V_1', V_2', V_3'\}$ and impersonate U_i. The attack procedure is as below:

1. The attacker $\mathcal{A}$ overhears overall login request of U_i; then, $\mathcal{A}$ can acquire at least $\{L, V_2, V_3, T_1, C_1, C_4, C_5\}$, and by implementing session key compromise attack, $\mathcal{A}$ can obtain R_i. Then, $\mathcal{A}$ computes and acquires $V_1 = V_3 \oplus (R_i \| T_1)$.
2. Now, $\mathcal{A}$ can impersonate U_i because $\mathcal{A}$ can make $L' = L \oplus T_1 \oplus T_1'$, $V_1' = V_1$, $V_2' = h\left(T_1' \| R_i' \| V_1\right)$, and $V_3' = h\left(R_i' \| T_1'\right) \oplus V_1$, where R_i' is chosen by $\mathcal{A}$ and T_1' is a current timestamp. Then, $\mathcal{A}$ transmits the login request $\{L', V_2', V_3', T_1'\}$ to the gateway GW_j.
3. GW_j considers the message from $\mathcal{A}$ as a login request from the U_i because T_1' is fresh and the message is authenticated by the shared information V_1.
4. When authentication message $\{C_1', C_4', C_5'\}$ comes, $\mathcal{A}$ can create a session key $SK_{ijk}' = h(R_i' \| R_j' \| R_k')$, where R_j' and R_k' can be derived from $\left(R_k' \| R_j' \| T_4'\right) = C_5' \oplus R_i'$, and successfully play the role of U_i.

4.4 Smart Card Loss Attack

When an attacker acquires a U_i's smart card, $\mathcal{A}$ can eventually learn U_i's identity ID_i and password PW_i. To be specific, after obtaining U_i's smart card, $\mathcal{A}$ extracts B_{ij} and finds a message in the log of all login-authentication messages whose result of $L \oplus T_1$ matches B_{ij}. After that, $\mathcal{A}$ performs a session key compromise attack to get R_i and V_1. With the derived V_1, $\mathcal{A}$ can find U_i's identity ID_i and password PW_i by performing the following steps:

1. After obtaining U_i's smart card (SC), $\mathcal{A}$ extracts $\{A_{ij}, c_i, l, HB_i\}$ from SC.
2. $\mathcal{A}$ chooses a random $ID_{\mathcal{A}}$ and $PW_{\mathcal{A}}$ and computes the following:

$$b_{\mathcal{A}} = c_i \oplus h(ID_{\mathcal{A}} \| PW_{\mathcal{A}}) \mod l$$
$$HID_{\mathcal{A}} = h(ID_{\mathcal{A}} \| b_{\mathcal{A}})$$
$$HPW_{\mathcal{A}} = h(PW_{\mathcal{A}} \| b_{\mathcal{A}})$$
$$HB_{\mathcal{A}} = h(HPW_{\mathcal{A}} \| HID_{\mathcal{A}} \| b_{\mathcal{A}}) \mod l$$

3. $\mathcal{A}$ compares $HB_{\mathcal{A}}$ with HB_i. If two parameters are different, then chooses another id $ID'_{\mathcal{A}}$ and password $PW'_{\mathcal{A}}$; otherwise, $\mathcal{A}$ inputs $ID_{\mathcal{A}}$ and $PW_{\mathcal{A}}$ after inserting U_i's SC into the card reader.
4. After inputting $ID_{\mathcal{A}}$ and $PW_{\mathcal{A}}$, SC requests a new password and nonce. Then $\mathcal{A}$ inputs the new password $PW^*_{\mathcal{A}}$ and a nonce $b^*_{\mathcal{A}}$, and SC performs password change phase.
5. After password change phase is done, $\mathcal{A}$ extracts a new parameter $A^*_{ij} = A_{ij} \oplus h(HID_{\mathcal{A}} \| HPW_{\mathcal{A}}) \oplus h(HID_{new} \| HPW_{new})$
6. Since $\mathcal{A}$ knows $HID_{\mathcal{A}}$, $HPW_{\mathcal{A}}$, $PW^*_{\mathcal{A}}$, and $b^*_{\mathcal{A}}$, $\mathcal{A}$ can find $h(HID_{new} \| HPW_{new}) = A^*_{ij} \oplus A_{ij} \oplus h(HID_{\mathcal{A}} \| HPW_{\mathcal{A}})$
7. $\mathcal{A}$ can find U_i's identity ID_i by applying identity guessing attack on $h(HID_{new} \| HPW_{new}) = h\big(h(ID_i \| b^*_{\mathcal{A}}) \| h(PW^*_{\mathcal{A}} \| b^*_{\mathcal{A}})\big)$, where $PW^*_{\mathcal{A}}$ and $b^*_{\mathcal{A}}$ are chosen by $\mathcal{A}$.
8. Now, $\mathcal{A}$ can also find U_i's password PW_i by applying password guessing attack on $h(HID_i \| HPW_i) = h(h(ID_i \| b_i) \| h(PW_i \| b_i)) = A_{ij} \oplus V_1$, where ID_i is already derived, V_1 is derived from the previous log of SC, and b_i is relatively small ($2^4 \leq l \leq 2^8$).

Note (2) Normally, this attack cannot be realized. This can be done in Kumar et al.'s scheme because the size of fuzzy verifier l is very small. If l is small, the probability of finding a collision $HB_{\mathcal{A}}$ of HB_i is very high. In addition, $b_i \in Z_l$ is also small, so $\mathcal{A}$ can apply password guessing attack on HPW_i (which is up to 256 times slower than original password guessing attack); thus, $\mathcal{A}$ can derive ID_i and PW_i. For this reason, it is required to increase the size of l sufficiently.

5 Conclusion

In this paper, we examined Kumar et al.'s two-factor user authentication scheme for WSNs and found that Kumar et al.'s scheme has four security vulnerabilities such as critical information leakage, session key compromise, user impersonation attack, and smart card loss attack. As a result of these attacks, sensor nodes' identification information is disclosed, the session key can be intentionally exposed, the user can be impersonated, and the user's identity and password can also be stolen. Therefore, we do not recommend using the current Kumar et al.'s authentication scheme.

In future work, we plan to develop an improved version of Kumar et al.'s scheme that is resistant to these attacks. Kumar et al.'s scheme is vulnerable due to the predictability of timestamp and the leakage of critical information. Therefore, our future scheme will be based on nonce-based and added additional technique on hiding critical information.

References

1. Kumari S, Om H (2016) Authentication protocol for wireless sensor networks applications like safety monitoring in coal mines. Comput Netw 104:137–154
2. Kumar D, Chand S, Kumar B (2019) Cryptanalysis and improvement of an authentication protocol for wireless sensor networks applications like safety monitoring in coal mines. J Ambient Intell Humaniz Comput 10(2):641–660
3. Dolev D, Yao A (1983) On the security of public key protocols. IEEE Trans Inf Theory 29(2):198–208
4. Wong KH, Zheng Y, Cao J, Wang S (2006) A dynamic user authentication scheme for wireless sensor networks. In: IEEE international conference on sensor networks, ubiquitous, and trustworthy computing
5. Das ML (2009) Two-factor user authentication in wireless sensor networks. IEEE Trans Wireless Commun 8(3):1086–1090
6. Huang HF, Chang YF, Liu CH (2010) Enhancement of two-factor user authentication in wireless sensor networks. In: Sixth international conference on intelligent information hiding and multimedia signal processing. IEEE, pp 27–30
7. He D, Gao Y, Chan S, Chen C, Bu J (2010) An enhanced two-factor user authentication scheme in wireless sensor networks. Ad hoc Sens Wireless Netw 10(4):361–371
8. Vaidya B, Makrakis D, Mouftah H (2016) Two-factor mutual authentication with key agreement in wireless sensor networks. Secur Commun Netw 9(2):171–183
9. Kim J, Lee D, Jeon W, Lee Y, Won D (2014) Security analysis and improvements of two-factor mutual authentication with key agreement in wireless sensor networks. Sensors 14(4):6443–6462
10. Chang IP, Lee TF, Lin TH, Liu CM (2015) Enhanced two-factor authentication and key agreement using dynamic identities in wireless sensor networks. Sensors 15(12):29841–29854
11. Park Y, Park Y (2016) Three-factor user authentication and key agreement using elliptic curve cryptosystem in wireless sensor networks. Sensors 16(12)
12. Jung J, Moon J, Lee D, Won D (2017) Efficient and security enhanced anonymous authentication with key agreement scheme in wireless sensor networks. Sensors 17(3)
13. Shin S, Kwon T (2019) A lightweight three-factor authentication and key agreement scheme in wireless sensor networks for smart homes. Sensors 19(9)