A Robust Mutual Authentication Protocol for Wireless Sensor Networks

Tien-Ho Chen and Wei-Kuan Shih

Authentication is an important service in wireless sensor networks (WSNs) for an unattended environment. Recently, Das proposed a hash-based authentication protocol for WSNs, which provides more security against the masquerade, stolen-verifier, replay, and guessing attacks and avoids the threat which comes with having many logged-in users with the same login-id. In this paper, we point out one security weakness of Das' protocol in mutual authentication for WSN's preservation between users, gateway-node, and sensor nodes. To remedy the problem, this paper provides a secrecy improvement over Das' protocol to ensure that a legal user can exercise a WSN in an insecure environment. Furthermore, by presenting the comparisons of security, computation and communication costs, and performances with the related protocols, the proposed protocol is shown to be suitable for higher security WSNs.

Keywords: Wireless sensor networks, hash function, parallel session attack, security, mutual authentication.

I. Introduction

Recently, wireless sensor networks (WSNs) have been applied in many different areas, for instance, the voltage variation monitoring in electric power companies, temperature and humidity remote controlling in museums and human health tracking systems. Normally, the client device needs to obtain authentication from the system where it wants to access.

Considering power consumption and computation capacity, sensor nodes generally do not execute a verification procedure directly but the gateway (GW)-node does [1]. According to IEEE 802.15.4 [2], Sastry and Wagner [3] provided an access restriction on security in that the access control list (ACL) can only contain 255 entries maximally. In other words, not only to verify the client's authentication and arrange the nearest sensor node cooperating with the client, GW node also needs to care for the limitation of ACL. Watro and others [4] proposed an authentication protocol by applying intricate mathematical methods. Afterward, Wong and others [5] presented a simpler hash-based protocol for authentication. However, Das [6] and Tseng and others [7] pointed out that Watro's approach still suffered from the masquerade attack, and that the method of Wong and others could not resist the stolen-verifier, replay, and forgery attacks. Therefore, both approaches [4], [5] are vulnerable to the threat of a multi-user with one login-id.

Das proposed a two-factor user authentication for WSNs, claiming that the mechanism could avoid not only replay and stolen-verifier attacks but also the guessing and masquerade attacks. Unfortunately, we find that Das' protocol fails in mutual authentication. To tackle this problem, this paper provides effective improvements with higher security. Moreover, because there are more threats in WSNs than any other related networking systems, the user authentication in

Manuscript received: Mar. 15, 2010; revised July 1, 2010; accepted Aug. 2, 2010.

Tien-Ho Chen (phone: +886 3 5742808, email: d918325@oz.nthu.edu.tw) and Wei-Kuan Shih (email: wShih@rtlab.cs.nthu.edu.tw) are with the Department of Computer Science, National Tsing Hua University, HsingChu, Taiwan.

doi:10.4218/etrij.10.1510.0134

WSNs needs to consider more in mutual authentication. This is absent in Das' protocol. In this paper, we propose a mutual authentication for preservation in WSN between the user, GW-node, and sensor node, while the power consumption of the sensor node is the same as in Das' protocol and superior to the other related schemes.

The rest of this paper is organized as follows. Section II reviews the concept of Das' protocol, and section III discusses its weakness analysis. Section IV shows the details of our proposed scheme, while section V demonstrates the security analysis of our proposed scheme. Section VI compares the performances of the related protocols. Finally, section VII concludes this paper.

II. Review of Das' Protocol

The notations used throughout this paper are summarized as follows:

- U: the user
- ID: the identity of U
- PW: the password of U
- DID: dynamic login identity of U
- S_n: the sensor node of WSN
- GW-node: the gateway node of WSN
- x_a: the permanent secret key generated by the GW-node and stored in some defined sensor nodes before-deploying the nodes in the field
- h(.): a secure one-way hash function
- ||: string concatenation operation
- K: symmetric key of GW-node shared between the GW-node, users and the sensor nodes
- ⊕: string XOR operation
- ⇒: a secure channel
- →: a common channel

There are three phases in Das' protocol: registration, login, and verification. A description of each follows.

Registration phase. In this phase, user U_i has to submit an identity, ID_i, and a password, PW_i, to the GW-node in a secured way. Then, the GW-node issues a license to U_i. The two detailed steps are depicted as follows:

Step 1. $U_i \Rightarrow$ GW-node: $\{ID_i, PW_i\}$.
U_i selects an ID_i and a password PW_i and then sends $\{ID_i, PW_i\}$ to the GW-node by the secure channel.

Step 2. GW-node $\Rightarrow U_i$'s smart card: $\{ID_i, N_i, h(PW_i), x_a, h()\}$.
After receiving the message from U_i and deciding to accept U_i's request, the GW-node calculates the results and personalizes the smart card with parameters $\{ID_i, N_i, h(PW_i), x_a, h()\}$. Then the GW-node sends the smart card to U_i by the secure channel, where $N_i = h(ID_i \oplus PW_i) \oplus h(K)$.

Login phase. When U_i enters an ID_i and a PW_i in order to deliver some query to or access data from the WSN, the smart card must perform the following steps to validate the legitimacy of U_i:

Step 1. The smart card validates the legitimacy of U_i.
U_i's smart card checks whether the ID_i and PW_i are correct. If they are not correct, it terminates the request.

Step 2. U_i's smart card computes DID_i and C_i.
$DID_i = h(ID_i \| PW_i) \oplus h(x_a \| T)$, where T is the current timestamp of U_i's system, and $C_i = h(N_i \| x_a \| T)$.

Step 3. $U_i \rightarrow$ GW-node: $\{DID_i, C_i, T\}$.
U_i sends the message $\{DID_i, C_i, T\}$ to the GW-node.

Verification phase. After receiving the login request message $\{DID_i, C_i, T\}$ at time T^*, the GW-node executes the following steps to verify the user U_i:

Step 1. Check $T^* - T < \Delta T$?
The GW-node checks whether $(T^* - T) \le \Delta T$ holds, where ΔT is the legal time interval for transmission delay. If the answer is yes, the validity of T can be assured, and the GW-node proceeds to the next step. If no, the GW-node rejects the request.

Step 2. Compute $h(ID_i \| PW_i)^*$ and C_i^*,
where $h(ID_i \| PW_i)^* = DID_i \oplus h(x_a \| T)$, and
$C_i^* = h(h(ID_i \| PW_i)^* \| h(K) \| x_a \| T)$.

Step 3. Check $C_i = C_i^*$?
The GW-node verifies whether $C_i = C_i^*$ is correct. If yes, the GW-node accepts the login request, and the GW-node proceeds to the next step.

Step 4. GW-node $\rightarrow$ S_n: $\{DID_i, A_i, T'\}$.
The GW-node calculates A_i and sends the message $\{DID_i, A_i, T'\}$ to the nearest sensor network S_n through the public channel in order to respond to the request of U_i at current time T', where $A_i = h(DID_i \| S_n \| x_a \| T')$.

Step 5. S_n checks T' and A_i.
After receiving the message $\{DID_i, A_i, T'\}$ at T, S_n executes the following step to verify the request. S_n first checks whether the relation of $(T - T') \le \Delta T$ holds. If yes, then S_n checks whether $A_i = h(DID_i \| S_n \| x_a \| T')$ is correct. If yes, then S_n responds to U_i's request. If no, S_n rejects the request.

III. Cryptanalysis of Das' protocol

In this section, we will discuss the requirements of security in WSNs and describe the flaw of Das' protocol; namely, it omits mutual authentication.

1. Security Requirements in WSNs

Sastry and Wagner [3] investigated several issues regarding IEEE 802.15.4 [2], such as ACL management problems (that is, the same key in multiple ACL entries, loss of ACL state due to power interruptions, key management problems, and insufficient integrity protection), and provided some solutions for these problems. However, the requirements for security authentication protocol in WSNs need to be considered by more appropriate methods in order to resolve the application layer issues, such as impersonation, replay, parallel session, sinkhole and wormhole attacks as well as other kinds of sensor node attacks. Furthermore, mutual authentication, which Das' protocol failed to provide, must be considered in the insecure networks as well.

2. Mutual Authentication

Assume that a malicious user, Allen, wants to attack a WSN. He can accomplish his purpose by eavesdropping and masquerading. A more detailed description of the attack can be stated as follows.

When U_i sends the message $\{DID_i, C_i, T\}$ to the GW-node to access the WSN, the GW-node sends the message $\{DID_i, A_i, T\}$ to S_n asking for the service for U_i. At this point, Allen can provide an S_M (which was not arranged by the GW-node) to impersonate the S_n and get U_i's request data or hold back the request. Since S_M co-works with U_i continuously, U_i will fail the accessing request continuously as well.

As a result, Das' protocol cannot provide mutual authentication. Moreover, in order to achieve the objective of secure authentication in insecure WSNs, we propose an improved protocol to achieve the following goals: (i) the application layer requirement of IEEE 802.15.4 and (ii) the resistance to attacks of impersonation, replaying and parallel session. Such a protocol can also help implementing mutual authentication.

Furthermore, assume that Tom is a legal user of the system. Tom can make a parallel attack the WSN by eavesdropping and masquerading. A more detailed description of Tom's attack can be expressed as follows.

Being a legal user of the system, Tom can login to the WSN at T_1 and T_2 accurately. When another legal user, U_i, wants to login to the WSN at T_1 and T_2 (if Tom has embedded a synchronized Trojan virus into U_i's system), Tom can eavesdrop on the message $\{DID_i, C_i, T_1\}$ and $\{DID_i, C_i, T_2\}$ between the GW-node and U_i at T_1 and T_2. Tom can obtain the following messages:

$$DID_{i(T1)} = h(ID_i \| PW_i) \oplus h(x_a \| T_1) \text{ and}$$
$$DID_{i(T2)} = h(ID_i \| PW_i) \oplus h(x_a \| T_2).$$

And then Tom can forge the dynamic login identity

$$DID_{\text{Tom}(T1)} = h(ID_{\text{Tom}} \| PW_{\text{Tom}}) \oplus h(x_a \| T_1) \text{ and}$$
$$DID_{\text{Tom}(T2)} = h(ID_{\text{Tom}} \| PW_{\text{Tom}}) \oplus h(x_a \| T_2).$$

Tom can use the login phase formula to compute $DID_{i(T2)}$, where $DID_{i(T2)}$ is calculated as

$$DID_{i(T2)} = DID_{i(T1)} \oplus DID_{\text{Tom}(T1)} \oplus DID_{\text{Tom}(T2)}$$
$$DID_{i(T2)} = h(ID_i \| \text{PW}_i) \oplus \boxed{h(x_a \| T_1)} \oplus h(ID_{\text{Tom}} \| PW_{\text{Tom}}) \oplus \boxed{h(x_a \| T_1)} \oplus h(ID_{\text{Tom}} \| PW_{\text{Tom}}) \oplus h(x_a \| T_2)$$

Afterward, Tom obtains the user U_i's $DID_{i(T2)}$ and sends a new session message $\{DID_{i(T2)}, C_i, T_A\}$ at T_A (where $T_A = T_2$ is made by Tom for attack on the WSN) for a new login request. Thus, the GW-node will verify message $\{DID_{i(T2)}, C_i, T_A\}$ from Tom with following steps: $U_{\text{Tom}} \rightarrow$ GW-node: $\{DID_{i(T2)}, C_i, T_A\}$.

Step 1. The GW-node receives $\{DID_{i(T2)}, C_i, T_A\}$ at T^* to check $T^* - T_A < \Delta T$ and the GW-node passes the verification to proceed to the next step ($T^* - T_2 < \Delta T$ is known and $T_A = T_2$ made arbitrarily by Tom).

Step 2. The GW-node calculates $h(ID_i \| PW_i)^* = DID_{i(T2)} \oplus h(x_a \| T)$ and obtains $C_i^* = h(h(ID_i \| PW_i)^* \| h(K) \| x_a \| T)$ ($C_i^* = C_i$) to pass the verification and proceed to the remaining steps.

As a result, Das' protocol cannot resist parallel session attack.

IV. Enhanced Mutual Authentication Protocol

This section presents our enhanced protocol for the two-factor authentication in WSNs, which has three phases: registration, login, and verification.

Registration phase. As in other similar schemes, user U_i has to submit an identity ID_i and a password PW_i to the GW-node in a secured way. Then, the GW-node performs the license to U_i. The steps are stated as follows:

Step 1. $U_i \Rightarrow$ GW-node: $\{ID_i, PW_i\}$.

Step 2. GW-node $\Rightarrow U_i$'s smart card: $\{ID_i, N_i, h(PW_i), x_a, h()\}$.

Login phase. When U_i enters ID_i and PW_i in order to deliver some query to or access data from the network, the smart card must perform the following steps to validate the legitimacy of U_i:

Step 1. The same as step 1 of the login phase in section II.

Step 2. Compute DID_i and C_i.

U_i's smart card generates a random nonce number R_i at T_u and performs the following computations:

$DID_i = h(ID_i \| PW_i) \oplus h(x_a \| T_u \| R_i)$, where T_u is the

current timestamp of U_i's system.

$C_i = h(N_i \| x_a \| T_u \| R_i)$.

Step 3. $U_i \rightarrow$ GW-node: $\{DID_i, C_i, T_u, R_i\}$

U_i sends the message $\{DID_i, C_i, T_u, R_i\}$ to the GW-node.

Verification phase. After receiving the login request message $\{DID_i, C_i, T_u, R_i\}$ at T_g, the GW-node executes the following steps to verify the user U_i:

Step 1. Check $T_g - T_u < \Delta T$?

The GW-node checks whether $(T_g - T_u) \leq \Delta T$ holds, where, similar to Das' scheme, ΔT is the legal time interval for transmission delay. If the condition holds, the validity of T can be assured, and the GW-node proceeds to the next step.

Step 2. Compute $h(ID_i \| PW_i)^*$ and C_i^*,

where $h(ID_i \| PW_i)^* = DID_i \oplus h(x_a \| T_u \| R_i)$ and $C_i^* = h(h(ID_i \| PW_i)^* \| h(K) \| x_a \| T_u \| R_i)$.

Step 3. Check $C_i = C_i^*$?

The GW-node checks whether $C_i = C_i^*$ is correct. If yes, the GW-node accepts the login request, and the GW-node proceeds to the next step.

Step 4. GW-node $\rightarrow$ S_n: $\{DID_i, A_i, T'\}$ and GW-node $\rightarrow$ U_i: $\{C_g, R_c\}$.

The GW-node generates a random nonce number R_c and calculates A_i and sends the message $\{DID_i, A_i, T'\}$ to the nearest sensor network S_n through the public channel to respond to the request of U_i at the current time T', where $A_i = h(DID_i \| S_n \| x_a \| T')$ and $C_g = h(DID_i \| S_n \| x_a \| R_c)$.

Step 5. S_n checks T' and A_i.

After receiving the message $\{DID_i, A_i, T'\}$ at T_n, S_n executes the following procedure to verify the request from U_i. S_n first checks whether $(T_n - T') \leq \Delta T$ holds at time T_n. If yes, then S_n checks whether $A_i = h(DID_i \| S_n \| x_a \| T')$ is correct. If the second yes is granted, then S_n sends $\{S_n\}$ and responds to U_i's request.

Mutual authentication phase. After receiving the message $\{C_g, R_c, S_n\}$, U_i executes the following step to verify the request:

Step 1. U_i checks C_g.

U_i first verifies whether $C_g = h(DID_i \| S_n \| x_a \| R_c)$ is correct. If yes, then U_i co-works with S_n.

V. Security and Performance Analysis

Referring to the security considerations for IEEE 802.15.4 Networks (which Sastry and Wagner proposed), the specification of IEEE 802.15.4, and the requirement for security authentication protocol in WSNs, we provide a mutual authentication for the WSN to protect inside and outside security [8]. There are numerous kinds of attacks in WSNs. In addition to the impersonation attack from sensor nodes, there are side channel [6], replay [9], impersonation, stolen-verifier, guessing, and parallel session attacks. In this section, we discuss the improved security feature and its association with the avoidance of parallel session attack.

1. Mutual Authentication

Mutual authentication is an important method to check mutual validity between the users, the GW-node and the sensor nodes, while the proposed scheme is described as below:

- When the user U_i logs into the WSN, she/he will be verified by $N_i = h(ID_i \oplus PW_i) \oplus h(K)$ from the GW-node.
- When the GW-node sends the message $\{C_g, R_c\}$ to U_i and the message $\{DID_i, A_i, T'\}$ to S_n through step 4 in the verification phase, U_i can verify the GW-node and S_n by $C_g = h(DID_i \| S_n \| x_a \| R_c)$.
- S_n can verify the GW-node by $A_i = h(DID_i \| S_n \| x_a \| T')$.
- The GW-node can verify S_n before all sensor nodes are deployed.

In summary, Das' protocol [6] fails to provide mutual authentication, the protocol of Watro and others [4] can provide mutual authentication but needs a third party to communicate securely, and the protocol of Wong and others [5] cannot achieve mutual authentication between the users and the sensor nodes at all. With our proposed scheme, we can provide a mutual authentication protocol for WSNs.

2. Other Secrecy Issues

The proposed mutual authentication can resist masquerade, stolen-verifier, replay, and guessing attacks. It can also withstand having many logged in users with the same login-id. The attacks are described below with further details.

Masquerade attack. An adversary who wants to impersonate a valid user U_i to log into a WSN must have the DID_i to validate their legitimacy. Since $DID_i = h(ID_i \| PW_i) \oplus h(x_a \| T_u \| R_i)$ and $C_i = h(N_i \| x_a \| T_u \| R_i)$ are calculated by one-way hash function, the adversary cannot decipher a DID_i and C_i without ID_i, x_a, and PW_i. Furthermore, no one can forge the GW-node without $h(K)$, which exists only in the real GW-node that the WSN has verified.

Stolen-verifier attack. An adversary can attack any system which has verifier tables for authentication, but our proposed mutual authentication approach does not need any verification table at all. As a result, there is no possible stolen-verifier attack

within our proposed protocol.

Replay attack. An adversary cannot replay a valid U_i's verification message, $\{DID_i, C_i, T_u, R_i\}$, to the GW-node to succeed in verification because the GW-node will verify whether $T_g - T_u < \Delta T$ when the message is obtained at T_g. Furthermore, the DID_i and C_i are hashed by a user created timestamp, T_u, and the random nonce, R_i, which never generate any duplicates.

Guessing attack. The calculations of the proposed protocol in the registration phase of $N_i = h(ID_i \oplus PW_i) \oplus h(K)$ and login phase $DID_i = h(ID_i \| PW_i) \oplus h(x_a \| T_u \| R_i)$ are by one-way hash function; therefore, the permanent secret key, x_a, and the symmetric key of GW-node K constitute a protection from guessing attack.

Many logged-in users with the same login-id. The proposed protocol provides a dynamic $DID_i = h(ID_i \| PW_i) \oplus h(x_a \| T_u \| R_i)$ that withstands the threat of many logged-in users with the same login-id because DID_i is hashed by the timestamp T_u and the random nonce R_i that never generates duplicates. In addition, the session will be terminated after the user's request is completed.

3. Halevi-Krawczyk Security Game for Enhanced Protocol

Resistance to parallel session attack [10], [11] is another important topic for authentication in WSNs. We shall first show that our proposed protocol resists the parallel session attack, and then illustrate with the Halevi-Krawczyk security game.

Assume that Allen, an animus but legitimate user, wants to parallel attack the WSN by intercepting and masquerading, and therefore, he must obtain another legitimate the user U_i's DID_i. According to the analysis in section III, Allen cannot forge U_i's DID_i since $DID_{i(T1)} = h(ID_i \| PW_i) \oplus h(x_a \| T_1 \| R_{i(T1)})$ and $DID_{i(T2)} = h(ID_i \| PW_i) \oplus h(x_a \| T_2 \| R_{i(T2)})$, where R_i is the new random nonce number at each login time. The attacker Allen cannot get $DID_{i(T2)}$ from $DID_{A(T1)} \oplus DID_{i(T1)} \oplus DID_{i(T2)}$

In 1999, Halevi and Krawczyk [12] proposed a mutual authentication model and a probabilistic game to test the strength of the model [13]. The model has three criteria: (i) there are many users in the model, (ii) one or more users can perform the sessions concurrently or sequentially, and (iii) the active attacker can control the information transmitted over communication lines and corrupt and/or control some of the users of the specific system. The game is parameterized by a security parameter, k, and a public dictionary, D, which contains all possible passwords. The game proceeds as follows.

Set-up phase. The server, S, chooses its cryptographic keys and publishes its public keys. The user, U, then uniformly picks a password, PW, from D and gives it to S while keeping it secretly from the adversary, A, who can also register clients with S at any time (before, during, or after the set-up phase) by picking any pair of identity, U^*, and password PW^* (provided that $U^* \neq U$ and $PW^* \in D$) and giving PW^* to S.

Game running phase. A has full control over all the clients she/he created, as well as the communication between A and S. That means U and S can only communicate through A, and all the messages between U and S sending or receiving just have to be through A. A may prefer to forward messages faithfully or modify messages capriciously. In addition, A can send special "prompt" messages to the parties at any time, causing them to start new authentication sessions, while each session will have a unique transaction called a session identifier (*sid*). This game is run until A decides to halt.

Outputs of parties. To meet the security requirements, U and S will record events related to the security of authentication by giving some special outputs. U outputs a pair of (S, *sid*) whenever U authenticates herself/himself to S under *sid*. S outputs (U, *sid*) whenever a successful authentication by U is completed during a *sid*. If an attempt to authenticate (asserted) U in session *sid* fails, S then outputs (U, *sid*, $\perp$). This is needed so that the "number of failed authentication attempts" can be counted.

Syntactically correct. An authentication protocol (U, S) is said to be "syntactically correct" if whenever all the messages between A and S in a *sid* are passed and remaining unchanged, then S and A output (U, *sid)* and (S, *sid)*, respectively.

Successful impersonation. An event in which S outputs (U, *sid*) but A has never output a pair (S, *sid*) is called a "successful impersonation." Here we assume that the last message is sent by A and A outputs (S, *sid*) only after the last message is sent, while S outputs (U, *sid*) only after receiving the last message sent by A.

Authentication failure. An event in which S outputs (U, *sid*, $\perp$) is called an "authentication failure."

Successful replay. An event in which S outputs a pair (A^*, *sid*) after already outputting some other pair (A^{**}, *sid*) in the past is called a "successful replay." Here A^* and A^{**} are arbitrary clients, and *sid* is the same in both pairs.

(£,m)-run. An (£,*m*)-*run* of the game is a run with the largest number of m active impersonation attempts, and A outputs the largest number of £ pairs of (S, *sid*). The adversary A achieves an (£,*m*)-*win* if in an (£,*m*)-run of the game, there is at least one successful impersonation or replay event.

GEN. There are three algorithms necessary for the game: the key-generation algorithm (GEN), the (probabilistic) encryption algorithm (ENC), and the decryption algorithm (DEC). A ciphertext-verification attack is formally defined via the following Halevi-Krawczyk Security Game [12] which

involves the three algorithms and an adversary A.

- The Halevi-Krawczyk Security Game

The security goal. Let $\Theta(\cdot,\cdot,\cdot)$ be a positive real function and (U, S) a syntactically correct authentication protocol. We say that (U, S) ensures one-way password-based authentication up to Θ, if for any probabilistic polynomial time adversary, A, any finite dictionary, D, any sufficiently large security parameter, k, any $£$, and any integer $m < |D|$, we have

$$\Pr[(£,m)\text{-}win] \leqq \left(\frac{m}{|D|}\right) + \Theta(k, £, m),$$

where Pr denotes the probability of the $(£,m)$-*win* game, the probability is taken over the random coins of S, U, and A in an $(£,m)$-*run* of the game. The security goal is to have $\Theta(k,£,m)$ be a negligible function in k.

Encryption probability. The encryption protocol, $\delta=(GEN, ENC, DEC)$, is said to resist ciphertext-verification c attacks, where GEN, ENC, and DEC are as defined previously, with security $\Theta=\Theta(k)$, if for any feasible adversary A,

$$|\Pr[A \text{ guesses 'encryption of } x_1' | DEC(c)=x_1] - \Pr[A \text{ guesses 'encryption of } x_1' | DEC(c)= x_2]| \leqq \Theta,$$

where the probability is taken over the random coins of GEN, $DEC(c) = x_1$ with 50% probability and $DEC(c) = x_2$ with 50% probability an $(£,m)$-*run* of the game (c denotes the cipher text, x_1 and x_2 represent the plaintexts).

Halevi-Krawczyk's theorem. Let δ be an encryption protocol that resists ciphertext verification attacks with security $\Theta(k)$. Let f be a function that is one-to-one on its components. Then, the encrypted challenge-response protocol, (U, S), with encryption, δ, and function, f, ensures authentication up to

$$\Theta'(k, £, m)=m\cdot£\cdot\Theta(k).$$

Since the proposed protocol in this paper is the same as the Halevi-Krawczyk security model, we will provide the following theorem to prove the security of the proposed protocol.

Theorem 1. The perfect random function can generate any nonce for the adversary. If the adversary has a perfect random function, for example, $\Re_{pfk}()$, then the adversary has only a negligible success probability in the Halevi-Krawczyk security game for two-factor mutual authentication protocol in WSNs.

Proof. The proof is by contradiction. Suppose there exists a probabilistic polynomial-time adversary, Allen, which has probability Θ to win in an $(£,m)$-*run* of the Halevi-Krawczyk game. We construct another adversary, Bob, who explodes the random function.

1) Assume $\Re()_{andom}$ is an ideal random function with a larger space $\{0,1\}^k$. Allen has a negligible success probability $\Theta_{collision}$ getting the same nonce by $\Re()_{andom}$ at the same time in the Halevi-Krawczyk security game, then as in Halevi-Krawczyk's theorem

$$\begin{aligned}\Pr_{Allen}[(£,m)\text{-}win] &= (1-\Theta_{collision})\left(\frac{m}{2^k} + £\cdot m\cdot\Theta(k)\right) + \Theta_{collision}\cdot 1\\ &= \left(\frac{m}{2^k} + £\cdot m\cdot\Theta(k)\right) + \Theta_{collision}\cdot\left(1-\left(\frac{m}{2^k} + £\cdot m\cdot\Theta(k)\right)\right)\\ &\leqq \frac{m}{2^k} + £\cdot m\cdot\Theta(k) + \Theta_{collision}\end{aligned}$$

2) Bob grants access to the authentication service system, S_A, which is either $\Re_{pfk}()$ (with 50% probability or an ideal random function, $\Re()_{andom}$ (with 50% probability). Bob can adaptively query an arbitrarily chosen string $x \in \{0,1\}^k$ to S_A and get the output which is either $\Re_{pfk}(x)$ or a random string uniformly selected from $\{0,1\}^k$ ($\Re(x)_{andom}$). After performing many polynomial queries, Bob finally makes the decision of whether or not S_A is the function $\Re_{pfk}(x)$ or the ideal random function $\Re()_{andom}$. As a result, Bob wins the game if the decision is correct.
3) To win the game, Bob runs a simulation of the Halevi-Krawczyk game and plays the role as the server, S_A. Suppose that the encryption and signature key pairs generated by Bob are (EKS, SKS) and (EKS', EKS'), and Bob invokes an adversary Allen in the game.
4) R_i^* denotes as a random string uniformly selected from $\{0,1\}^k$. Bob generates the random nonce number R_i^* which is obtained from $\Re_{pfk}(x) = R_i^*$ or $\Re(x)_{andom} = R_i^*$ and calculates $DID_i = h(ID_i||PW_i)\oplus h(x_a \,||T_u||\, R_i)$ and $C_i = h(N_i||x_a||T_u||\, R_i)$, where T_u is the current timestamp of U_i's system. The goal is to get the same random nonce number R_i. That means $R_i^*=R_i$ where R_i is generated by S_A at the same time.
5) Bob calculates DID_i^* and C_i^* for S_A enquiries with inputs DID_i and C_i. He then runs the game for $DID_i^*= DID_i$ and $C_i^*=C_i$ until Allen halts. Thus he will run the game in polynomial time. If an $(£,m)$-win occurs in the game, Bob makes a decision that the S_A is $\Re_{pfk}()$. Otherwise, Bob chooses the ideal random function $\Re()_{andom}$ as his decision. Then we have

$$\begin{aligned}&\Pr[\text{Bob wins}]\\ &= \Pr[\text{Bob}\,\Re_{pfk}(x) = R_i^* | S_A\,\Re_{pfk}(x) = R_i^*]\Pr[S_A\,\Re_{pfk}(x) = R_i^*]\\ &\quad + \Pr[\text{Bob}\,\Re(x)_{andom} = R_i^* | S_A\,\Re(x)_{andom} = R_i^*]\Pr[S_A\,\Re(x)_{andom} = R_i^*]\\ &= \frac{1}{2}\Pr[\text{Bob}\,\Re_{pfk}(x) = R_i^* \mid S_A\,\Re_{pfk}(x) = R_i^*]\\ &\quad + \frac{1}{2}\Pr[\text{Bob}\,\Re(x)_{andom} = R_i^* \mid S_A\,\Re(x)_{andom} = R_i^*]\end{aligned}$$

$$= \frac{1}{2} \Pr[\text{Bob}\,\Re_{pfk}(x) = R_i^* | \text{S}_\text{A} \;\; \Re_{pfk}(x) = R_i^*]$$

$$+ \frac{1}{2}(1 - \Pr[\text{Bob} \;\; \Re_{pfk}(x) = R_i^* | \text{S}_\text{A}\, \Re(x)_{\text{andom}} = R_i])$$

$$= \frac{1}{2} \Pr[(£,m)\text{-wins} \mid \text{S}_\text{A}\, \Re_{pfk}(x) = R_i^*]$$

$$+ \frac{1}{2}(1 - \Pr[(£,m)\text{-wins} \mid \text{S}_\text{A}\, \Re(x)_{\text{andom}} = R_i])$$

$$= \frac{1}{2} + \frac{1}{2}(\Pr[(£,m)\text{-wins} | \text{S}_\text{A}\, \Re_{pfk}(x) = R_i^*]$$

$$- \Pr[(£,m)\text{-wins} | \text{S}_\text{A}\, \Re(x)_{\text{andom}} = R_i]).$$

According to the Halevi-Krawczyk game, the successful probability of Allen's attack should be

$$\Pr_{\text{Allen}}[(£,m)\text{-wins} | \text{S}_\text{A} = \Re(x)_{\text{andom}}]$$

$$\leqq \frac{m}{2^k} + £ \cdot m \cdot \Theta(k) + \Theta_{\text{collision}}.$$

However, Bob has

$$\Pr[\text{Bob-win}] \geqq \frac{1}{2} + \frac{1}{2}(\Theta - (\frac{m}{2^k}) - £ \cdot m \cdot \Theta(k) - \Theta_{\text{collision}}).$$

It is a contradiction that Pr[Bob-*wins*] $\geqq$ Pr[Allen-*wins*]. Since [Bob-*wins*] must go through [Allen-*wins*], Pr[Bob-*wins*] $\geqq$ Pr[Allen-*wins*] is impossible. Therefore, it has been proven that our scheme is readily built from the mutual authentication model which Halevi and Krawczyk proposed [12]. □

It is worth recalling that the protocol of Wong and others [5] cannot achieve mutual authentication and has security weaknesses against the replay, forgery attacks, and password exposure to sensor nodes [14]. The approach of Watro and others [4] has the masquerade attack problem, while Das' [6] work uses the hash function for authentication but cannot provide mutual authentication as well. We summarize the comparison of related protocols with ours in Table 1. With reference to attack types and authentication attitudes, our protocol, because it prevents the masquerade, stolen-verifier, replay, and guessing attacks, and avoids the trouble of having too many logged in users with the same login-id, can reach higher security than all others. In addition, this proposed protocol resists the parallel session attack.

Table 1. Comparisons among related protocols.

Item	Ours	Das'	Watro's	Wong's
Avoiding masquerade attack	Yes	Yes	No	Yes
Avoiding many logged in users with the same login-id	Yes	Yes	Yes	No
Avoiding stolen-verifier attack	Yes	Yes	Yes	No
Avoiding replay attack	Yes	Yes	Yes	Yes
Avoiding guessing attack	Yes	Yes	Yes	Yes
Avoiding parallel session attack	Yes	No	Yes	Yes
Mutual authentication	Yes	No	Yes	No

Table 2. Performance comparison among related protocols.

Protocols		Ours	Das'	Watro's	Wong's
Registration	User	-	-	$t_{pu}+t_{pr}$	-
	GW-node	$3t_h$	$3t_h$	t_{pr}	$3t_h$
	Sensor node	-	-	-	-
Authentication (verification and mutual authentication)	User	$1t_h$	-	t_h+2t_{pr}	-
	GW-node	$5t_h$	$4t_h$	-	t_h
	Sensor node	t_h	t_h	t_h+2t_{pu}	$3t_h$
Communication cost	User	$4t_h$	$3t_h$	t_h+2t_{pr}	-
	GW-node	$5t_h$	$4t_h$	-	t_h
	Sensor node	t_h	t_h	t_h+2t_{pu}	$3t_h$

VI. Performance Analysis

In this section, we compare our improved protocol with related ones in terms of computation cost in the registration phase and communication cost in the message exchange phase since these two phases are the main procedures of an authentication protocol. Let's define t_h as the hash computation time, t_{pr} as the private key computation time, and t_{pu} as the public key computation time, as indicated in [7], [14]-[18]. The result is shown in Table 2.

The general goal of the performance suggested in the literature is to minimize the power consumption of the sensor node. It is clear that our protocol parallels Das' because the sensor node requires the least time, $1t_h$, in computing authentication, which is the most important factor of power consumption restriction for sensor nodes. For the time complexity comparison in the different operations, for instance, $t_{pu} >> t_h$ and $t_{pr} >> t_h$, the hash function needs much less time for calculation than t_{pu} and t_{pr}, where t_{pu} and t_{pr} usually need polynomial computation cost to obtain the public and private keys. It means that our protocol, at $1t_h$ (see Table 2), serves better than the Watro and Wong protocols and as well as Das' in terms of power consumption of the sensor node. The GW-node still needs ($5t_h$) for hash function (Das' is $4t_h$, Watro and others' is zero, and Wong and others' is $1t_h$). Although it is higher than others, we consider this as acceptable for the reason of that the GW-node always needs enormous data to encrypt and/or decrypt the user's requests, arranges the sensor node for responding to the requirement, and provides a mutual authentication between the U_i and the WSN. It is the same

reason that U_i needs a mutual authentication. Furthermore, when considering our computation cost in the authentication phase (which includes the verification and mutual authentication phases), note that the sensor node needs $1t_h$, which is the same as Das' protocol, and the GW-node needs $5t_h$, whereas Das' protocol needs $4t_h$. Finally, the user side needs $1t_h$ for mutual verification which Das' protocol cannot provide.

Lastly, when considering the communication cost the proposed protocol displays higher cost than other protocols, as the protocol of Wong and others needs 4 message exchanges, Das' protocol needs 3 and the protocol of Watro and others needs 2. Even though our protocol has the same number of message exchanges as that of Wong and others, the message size of ours is smaller. Das' protocol communication cost is lower than ours, but Das' protocol does not provide mutual authentication. The protocol of Watro and others needs 2 exchanges only but the computation times, for t_{pr} and t_{pu}, are more than the other three related approaches. Consequently, our communication cost is the most worthwhile among the compared protocols.

VII. Conclusion

This paper provides a robust mutual two-factor user authentication protocol for WSNs by applying hash functions. The proposed protocol performs more efficiently in terms of computation cost, communication cost, and security. Compared with the protocol of Wong and others, which is vulnerable to masquerade attack and multi-user with the same login-id, the protocol of Watro and others, which is vulnerable to masquerade attack, and Das' protocol, which cannot provide mutual authentication, the proposed protocol in this paper can prevent all the problems and provide mutual authentication to protect inside security and outside security. Therefore, the proposed protocol is more suited to WSNs environments.

References

[1] I.F. Akyildiz et al., "A Survey on Sensor Networks," *IEEE Commun. Mag.*, vol. 40, no. 8, Aug. 2002, pp. 102-114.

[2] IEEE Standards for 802.15.4, Part 15, Amendment 4, "Wireless Medium Access Control and Physical Layer Specifications for Low-Rate Wireless Personal Area Networks," 2003.

[3] N. Sastry and D. Wagner, "Security Considerations for IEEE 802.15.4 Networks," *Proc. ACM Workshop Wireless Security, ACM Press*, 2004, pp. 32-42.

[4] R. Watro et al., "C. Lynn, and P. Kruus, TinyPK: Securing Sensor Networks with Public Key Technology," *Proc. ACM Workshop Security Ad Hoc Sensor Netw.*, 2004, pp. 59-64.

[5] K. Wong et al., "A Dynamic User Authentication Scheme for Wireless Sensor Networks," *Proc. IEEE Int. Conf. Sensor Netw., Ubiquitous, Trustworthy Computing*, 2006, pp. 244-251.

[6] M.L. Das, "Two-Factor User Authentication in Wireless Sensor Networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 3, 2009, pp. 1086-1090.

[7] H.R. Tseng, R.H. Jan, and W. Yang, "An Improved Dynamic User Authentication Scheme for Wireless Sensor Networks," *IEEE Global Telecommun. Conf.*, 2007, pp. 986-990.

[8] S. Tripathy and S. Nandi, "Defense Against Outside Attacks in Wireless Sensor Networks," *Computer Commun.*, vol. 31, no. 4, 2008, pp. 818-826.

[9] B. Vaidya, M. Chen, and J. Rodrigues, "Improved Robust User Authentication Scheme for Wireless Sensor Networks," *5th IEEE Proc. Wireless Commun. Sensor Networks,* 2009, pp. 1-6.

[10] X. Duan, J.W. Liu, and Q. Zhang, "Security Improvement on Chien Et Al.'s Remote User Authentication Scheme Using Smart Cards," *IEEE Int. Conf. Computational Intell. Secur.*, 2006, pp. 1133-1135.

[11] C.H. Han and W.K. Shih, "Improvement of the Secure Dynamic ID Based Remote User Authentication Scheme for Multi-server Environment," *Comput. Stand. Interfaces*, vol. 31, no. 6, Nov. 2009, pp. 1118-1123.

[12] S. Halevi and H. Krawczyk, "Public-Key Cryptography and Password Protocols," *ACM Trans. Inf. Syst. Secur.*, vol. 2, no. 3, 1999, pp. 230-268.

[13] G. Yang et al., "Two-Factor Mutual Authentication Based on Smart Cards and Passwords," *J. Comput. Syst. Sci.*, vol. 74, no. 7, 2008, pp. 1160-1172.

[14] H.Y. Chien, J.K. Jan, and Y.M. Tseng, "An Efficient and Practical Solution to Remote Authentication: Smart Card," *Computers & Security*, vol. 21, no. 4, 2002, pp. 372-375.

[15] D. Dolev and A.C. Yao, "On the Security of Public-Key Protocols," *IEEE Trans. Inform. Theory,* vol. 29, no. 2, 1983, pp. 198-208.

[16] Hsiang and W.K. Shih, "Weaknesses and Improvements of the Yoon-Ryu-Yoo Remote User Authentication Scheme Using Smart Cards," *Computer Commun.*, vol. 32, no. 4, 2009, pp. 649-652.

[17] P.C. Kocher, J. Jaffe, and B. Jun., "Differential Power Analysis," *Proc. Advances Cryptology*, *LNCS, 1666*, Springer-Verlag, 1999, pp. 388-397.

[18] T.H. Chen, H.C. Hsiang, and W.K. Shih, "Security Enhancement on an Improvement on Two Remote User Authentication Schemes Using Smart Cards," *Future Gen. Comput. Syst.*, Accepted, doi: 10.1016/j.future.2010.08.007. 2010.

Tien-Ho Chen received the MS in information and computer engineering from the Chung Yuan Christian University, Chung-Li, Taiwan, in 1998. He completed his PhD in computer science from the National Tsing Hua University, Taiwan, in 2010. His research activities are mainly focused on cryptography, information security, WSN, wireless communication, electronic commerce, and fuzzy logic control.

Wei-Kuan Shih is a professor at the Department of Computer Science, National Tsing Hua University, Taiwan. He completed his PhD degree from the University of Illinois, Urbana-Champaign, in 1990. His current research interests include real-time systems, wireless, WSN, Internet technology, embedded system, multimedia system, and information security.