Quan *et al. Journal of Cloud Computing: Advances, Systems and Applications* (2015) 4:6
DOI 10.1186/s13677-015-0030-z

Journal of Cloud Computing
a SpringerOpen Journal

RESEARCH Open Access

A secure user authentication protocol for sensor network in data capturing

Zhou Quan[1,2*], Tang Chunming[1,2], Zhen Xianghan[3] and Rong Chunming[4]

Abstract

Sensor network is an important approach of data capturing. User authentication is a critical security issue for sensor networks because sensor nodes are deployed in an open and unattended environment, leaving them possible hostile attack. Some researchers proposed some user authentication protocols using one-way hash function or using biometric technology. Recently, Yel et al. and Wenbo et al. proposed a user authentication protocols using elliptic curves cryptography. However, there are some security weaknesses for these protocols. In the paper, we review several proposed user authentication protocols, with a detail review of the Wenbo et al.'s user authentication protocol and a cryptanalysis of this protocol that shows several security weaknesses. Furthermore, we propose a secure user authentication protocol using identity-based cryptography to overcome those weaknesses. Finally, we present the security analysis, a comparison of security, computation, and performance for the proposed protocols, which shows that this user authentication protocol is more secure and suitable for higher security WSNs.

Keywords: Data capturing; Wireless sensor networks; User authentication; Identity-based cryptography

Introduction

With the application of big data, there are some base manipulation processes: data capturing, data transport, data storage, data extraction & integration, data analysis & interpretation and data application. In the data capturing, using all kinds of devices and methods to collect data, such as smart devices, sensors, Web. So there are three important approaches of data capturing: Internet, Internet of Things (IoT) and sensor network [1]. Wireless Sensor networks (WSNs) is an open environment distributed network, which is an important approach of data capturing for big data. Nevertheless, with the application of dig data, the requirement of real-time data from WSNs is increasing highly. In some situations the gateway impossibly does force a user to access the sensor node directly. In such case the security and reliability to inquire and data disseminate are very important. Only when every client (remote sensor node, remote user) in the WSNs proves his/her identity can he/she be allowed to join the WSNs and access to resource, such as real-time data. Thus, a key security requirement for WSNs is user authentication [2-5].

In 2004, Sastry et al. [2] proposed a security scheme using access control lists (ACL) for IEEE 802.15.4 networks in the gateway node. An ACL would be maintained in gateway node and sensor nodes. Watro et al. [6] proposed a user authentication protocol using RSA and Differ-Hellman algorithm, but which was open to hostile attack by a user masquerading.

In 2005, Benenson et al. [7] proposed a user authentication protocol based on elliptic curve discrete logarithm problem (ECDLP) to handle the sensor node capture attack, which relied on a trusted third party.

In 2006, Wong et al. [8] proposed a dynamic user authentication scheme for WSNs based on a light-weight strong password using hash function, which included three phases: registration phase, login phase and authentication phase. Nonetheless, Tseng et al. [9] and Das [10] pointed out that this protocol had some weaknesses in protecting against replay attack, forgery attack, stolen-verifier attack, sensor node revealing and exposing the password to the other node and no updating user's password. In 2007, Tseng et al. [9] proposed an enhanced user authentication protocol by adding an extra phase (password changing phase) on Wong et al.'s phases. However,

---

* Correspondence: zhouqq@gzhu.edu.cn
[1]Key Laboratory of Mathematics and Interdisciplinary Sciences of Guangdong Higher Education Institutes, Guangzhou University, Guangzhou, China
[2]School of Mathematics and Information Science, Guangzhou University, Guangzhou, China
Full list of author information is available at the end of the article

in 2008 Ko [11] showed the Tseng et al.'s protocol was still insecure and did not provide mutual authentication.

In 2009, Das [10] proposed a two-factor user authentication protocol based on password and smart card against stolen-verifier attack. Nevertheless, Nyang et al. [12] showed there were some security weaknesses in offline-password guessing attacks.

In 2010, Vaidya et al. [13] demonstrated the Tseng et al.'s protocol, Wong et al.'s protocol and Ko's protocol were still not strong enough to protect again replay attack, stolen-verifier attack and man-in-the-middle attack. Khan et al. [14,15] pointed out the Das's protocol did not provide mutual authentication, and against by passing attack and privileged insider attack. Moreover, Chen et al. [16] also demonstrated the Das's protocol did not provide mutual authentication between the gateway node and the sensor node. And Chen et al. proposed a more secure and robust two-factor user authentication scheme for WSNs.

In 2011, Yeh et al. [17] found that the Chen et al.'s protocol failed to provide a secure method for updating password and insider attack. And Yeh et al. proposed a new user authentication scheme for WSNs using elliptic curve cryptography (ECC). Unfortunately, Han [18] found this protocol still had some weaknesses: no mutual authentication, no key agreement between the user and the sensor node, and no prefer forward security. Meanwhile, Yuan et al. [19] proposed a biometric-based user authentication for WSNs using password and smart card in 2010. Unfortunately, in 2011 Yoon et al. [20] showed the integrity problem of the Yuan et al.'s protocol and proposed a new biometric-based user authentication scheme without using password for WSNs.

In 2012, Ohood et al. [21] pointed out Yoon et al.'s scheme still had some drawbacks, such as no key agreement, no message confidentiality service, no providing against DoS and node compromise attack. Moreover, Ohood et al. [22] proposed an efficient biometric authentication protocol for WSNs.

Recently, Wenbo et al. [23] in 2013 proposed a new user authentication protocol for WSNs using elliptic curve cryptography to overcome the security weaknesses of Yeh et al.'s protocol. Although they suggested security improvements of Yeh et al.'s protocol, there were some security weaknesses in their protocol, e.g. no mutual authentication between the user and sensor node, no protecting against insider attack, forgery attack and DoS (denial of service) attack.

To address all of the issues raised in the above studies, we propose a secure user authentication protocol using identity-based cryptography on the basis of our previous studies to trusted management and trusted architecture of WSNs [24-26]. Our proposal addresses the key security issues.

The remainder of this paper is organized as follows: in Section Related works, we review the Wenbo et al.'s protocol and a detail cryptanalysis; next we present our user authentication protocol based on identity-based encryption in Section Proposed protocol; in Section Security and performance analysis, a security and performance analysis of the related protocol is presented; in Section Conclusion, we provide some conclusion remarks.

Related works

Notation

In Table 1, some notations used throughout this paper and their corresponding definitions are shown.

Review of Wenbo's scheme

In the Wenbo's protocol, the gateway GW held two master keys (x and y). And it was assumed that the gateway and the sensor nodes shared a long-term common secret key, $\mathrm{SK}_{\mathrm{GS}} = h(S_n||y)$. The Wenbo's protocol involves the registration phase, login phase, authentication phase and password update phase, which can be briefly described as follows.

Registration phase

In this phase, a user U submits his/her ID_u and a hash of his/her password to GW via a secured channel. Then, GW issues a license to U. The steps are described as follows.

Step 1: $U \rightarrow GW$: $\{ID_u, PS'\}$.

U enters an identity, selects a random number br and a password PS. And U computes $PS' = h(PS \oplus br)$. Then U sends message $\{ID_u, PS'\}$ to GW via a secured channel.

Step 2: $GW \rightarrow$ a smart card of U: $\{Bu, Wu, h(.)\}$.

GW computes $Ku = h(ID_u||x) \times P$, $Bu = h(ID_u \oplus PS')$, and $Wu = Bu \oplus Ku$, where x is a master key of GW. Then the GW stores (Bu, Wu) into a smart card and sends it to U.

Login phase

When U access Sn, U needs enter his ID_u and PS. And the smart card must confirm the validity of U via the following steps.

Step 1: Validate U.

The smart card check whether $Bu = h(ID_u \oplus h(PS \oplus br))$ hold. If the answer is no, the U's identification validation fails and the smart card will terminate this request. Otherwise, the smart card continues to execute the next step.

Step 2: U's smart card generates a random number r_u, calculates X and a. $X = r_u \times P$, $X' = r_u \times (Bu \oplus Wu)$, and $a = h(ID_u||X||X'||T_u)$, where T_u is the curren time of U's system.

Step 3: $U \rightarrow Sn$: $\{ID_u, X, T_u, a\}$.

The $\{ID_u, X, Tu, a\}$ is submitted to Sn via public channel.

Table 1 Notations

Symbol	Define
p	A big prime number
Fp	A finite field
E	An elliptic curve in F_p with a large order
P	A point on elliptic curve E with order q that is a big prime number
U	A remote user
ID	An identity
PS	A user password
GW	Gateway of WSNs
S_n	Sensor node of WSNs
Q_{id}	Public key of id
d_{id}	Private key of id
P_{set}	A system parameter set of PKG
$h(.)$	A public secure one-way hash function
$H_1(.)$	A public function: $\{0,1\}^* \rightarrow G_1$, the G_1 is a group $G_1 = \{NP \mid n \in \{0,1 \ldots q-1\}\}$
$H_2(.)$	A public function $G_2 \rightarrow \{0,1\}^*$, G_2 is subgroup with an order q of $GF(p^2)^*$
$f(.)$	A public function: $G_1 \rightarrow \{0,1\}^*$
$\hat{e}(.)$	An admissible pairing: $G_1 \times G_1 \rightarrow G_2$
$E_k(m)$	Encrypt message m with key k
$D_k(c)$	Decrypt message c with key k
$\|\|$	A string concatenation operation
$\oplus$	A XOR operation

Authentication phase

The authentication phase includes: *Sn* checking the validity of the request message of *U*, *GW* authenticating *Sn* and *U, Sn* authenticating *GW* and *U, U* authenticating *Sn* and *GW*.

Sn* checks the validity of the request message of *U When receiving the login message $\{ID_u, X, T_u, a\}$ at time T', *Sn* checks and generates request message which is sent to *GW* for authentication. *Sn* executes the following steps.

Step 1: Checks T_u.

Sn checks if $(T'\text{-}T_u) \leqq \Delta T$ holds, where ΔT denotes the expected time interval for transmission delay. If the answer is yes, the validity of T_u can be assured, and *Sn* executes the next step. Otherwise *Sn* rejects the login request.

Step 2: Picks a random number r_s and calculates Y and b.

$Y = r_s \times P$, $b = h(\mathrm{SK}_{GS}||ID_u||X||T_u||a||ID_{Sn}||Y||T_s)$, where T_s denotes the current request time of the *Sn* system.

Step 3: $Sn \rightarrow GW$: $\{ID_u, X, T_u, a, ID_{Sn}, Y, T_s, b\}$.

The $\{ID_u, X, T_u, a, ID_{Sn}, Y, T_s, b\}$ is submitted to *GW* via public channel.

GW* authenticates *Sn* and *U When receiving the request message that sent by *Sn* at time T'', *GW* checks and validates *Sn* and *U*, and generates the response message that will be sent to *Sn. GW* executes the following steps.

Step 1: Validates if T_s and T_u.

GW checks whether $(T''\text{-}T_s) \leqq \Delta T$ and $(T''\text{-}T_u) \leqq \Delta T$ hold. If the answer is yes, the validity of T_s and T_u can be assured and *GW* executes the next step. Otherwise *GW* rejects this request message.

Step 2: Calculates b^*.

$$b^* = h(SK_{GS}||ID_u||X||T_u||a||ID_{Sn}||Y||T_s).$$

Step 3: Confirms whether $b = b^*$ and validates *Sn*.

GW checks if $b = b^*$ holds. If the answer is yes, *GW* accepts this request message and executes the next step. Otherwise, *GW* rejects this request message.

Step 4: Calculates X' and a^*.

$X' = h(ID_u||x) \times X$, $a^* = h(ID_u||X||X'||T_u)$, where x denotes a master key of *GW*.

Step 5: Confirms whether $a = a^*$.

GW checks if $a = a^*$ holds. If the answer is yes, *GW* accepts this request message and executes the next step. Otherwise, *GW* rejects the request message.

Step 6: Calculates y and l.

$$y = h(SK_{GS}||ID_u||X||T_u||a||ID_{Sn}||Y||T_G),$$

$$l = h(ID_u||X||X'||T_u||Y||T_s),$$

where T_G denotes the current response time of *GW*.

Step 7: $GW \rightarrow S_n$: $\{T_G, y, l\}$

The $\{T_G, y, l\}$ is submitted to *Sn* via public channel.

Sn* authenticates *GW When receiving the response message that sent by *GW* at time T''', *Sn* checks and validates *GW*, and generates the message that will be sent to *U. Sn* executes the following steps.

Step 1: Validates T_G .

Sn checks if $(T'''\text{-}T_G) \leqq \Delta T$ holds. If the answer is yes, the validity of T_G can be assured and *Sn* executes the next step. Otherwise *Sn* rejects the response message.

Step 2: Calculates y^*.

$$y^* = h(SK_{GS}||ID_u||X||T_u||a||ID_{Sn}||Y||T_G).$$

Step 3: Validates y.

Sn checks if $y = y^*$ holds. If the answer is yes, *Sn* accepts this response and executes the next step. Otherwise, *Sn* rejects this response message.

Step 4: Calculates K_{SU}, g and session key sk.

$$K_{SU} = r_s \times X, g = h(Y||T_s||l||K_{SU}), \ \ sk = h(X||Y||K_{US}).$$

Step 5: $S_n \rightarrow U$: $\{Y, T_s, l, g\}$

The $\{Y, T_s, l, g\}$ is submitted to *U* via public channel.

U* authenticates GW and *Sn When receiving the response message that sent by *Sn* at time T'''', *U* checks and validates *GW* and *Sn*. *U* executes the following steps.

Step 1: Validates *Ts*.

U checks if (T''''-*Ts*)≦Δ*T* holds. If the answer is yes, the validity of T_S can be assured and *U* executes the next step. Otherwise, *U* rejects the response message.

Step 2: Calculates K_{US}, l^* and g^*.

$$K_{SU} = r_u \times Y,\ \ l^* = h(ID_u||X||X'||T_u||Y||T_s)),\ \ and g^* = h(Y||T_s||l||K_{SU}).$$

Step 3: Confirms $l\ and\ g$.

U checks if $l = l^*$ and $g = g^*$ hold. If the answer is yes, *U* accepts the response message and executes the next step. Otherwise, *U* rejects the response message.

Step 4: Calculates session key sk.

$$sk = h(X||Y||K_{US}).$$

Password update phase

When *U* wants to update his/her old password, *U* and the smart card execute the following steps.

Step 1: *U* inserts his/her smart card into the smart terminal and enters his/her identify ID_u, the old password *PS* and the new password *PSn*.

Step 2: The smart card calculates $PS' = h(PS \oplus br)$, and checks whether $Bu = h(ID_u \oplus PS')$ holds. If it does not hold, the smart card stops *U*'s request. Otherwise, the smart card continues to compute $Ku = h(ID_u||PS') \oplus Wu$, $PSn' = h(PSn \oplus br)$, $Bu' = h(ID_u \oplus PSn')$ and $Wu' = Bu' \oplus Ku$. Finally, the smart card replaces (*Bu*, *Wu*) with (*Bu*', *Wu*').

Cryptanalysis of Wenbo's protocol

Security requirements in WSNs

(1) Secure user authentication in WSNs should be based on full mutual authentication.
(2) Secure user authentication in WSNs should resist masquerade, replay, forgery and DoS attacks.
(3) Secure user authentication in WSNs should resist internal attack (compromise attack).
(4) Secure user authentication in WSNs with smart card should reject Virus Injection attack.

No full mutual authentication

Because Wenbo's protocol does not authenticate *U* during the authentication phase (***Sn checks the validity of the request message of U***), a malicious user can attack *Sn* and *GW* by means of forging. The attack could be accomplished as follows:

(1) The attacker sends a forging message {ID_a, X_a, Tu_a, a_a} to *Sn*.
(2) *Sn* sends a message {ID_a, X_a, Tu_a, a_a, ID_{Sn}, *Y*, T_s, *b*} to *GW* for authenticating the user when receiving the forging message.

During the above process, since *Sn* does not authenticate the user, *Sn* directly generates authenticating request message for *GW* to authenticate the user. When *GW* receives this request message, *GW* can finish the process from Step 1 to Step 4 of authentication phase (***GW authenticates Sn and U***). This is because there is no mechanism for *Sn* to be assured that *U* is real user of WSNs. Thus, the Wenbo's protocol does not provide mutual authentication between *U* and *Sn*. There is no full mutual authentication between *Sn* and *U*. This protocol cannot reject DoS attack to *Sn* and *GW*.

No protection against forgery attack

Because the confidential information (*Bu*, *Wu*) is not encrypted to be stored, the attacker can masquerade as a legal user *U*. In the case that an attacker steals the (*Bu*, *Wu*) from the smart card via some a Virus or a Trojan in the user terminal, he/she maybe try to impersonate user *U* to access resource in WSNs. The attack can be accomplished via the following means.

(1) The attacker steals the (*Bu*, *Wu*)} via some methods, such as Virus software, Trojan.
(2) The attacker could compute $Ku = Bu \oplus Wu$ and gain the secret *Ku*.
(3) The attacker picks a random number R_u.
(4) The attacker could computes $X_a = R_u \times P$, $X_a' = R_u \times Ku$, and $a_a = h\ (ID||X_a||X_a'||T_a)$ because the point *P* on elliptic curve *E* is public.
(5) The attacker sends the request message {ID_u, X_a, T_a, a_a} to the *Sn* via public channel.
(6) *Sn* can finish the authentication phase processes. And *GW* also can accomplish the authentication phase processes.

After *GW* and *Sn* finish to authenticate, the attacker can gains the session key sk. The attacker continues to access *Sn*. Thus, the Wenbo's protocol does not provide sufficient protection against forgery attack.

No protection against insider attack

In the Wenbo's protocol, *U* uses a single password for accessing *Sn*. It is convenient for a user. Nevertheless, if the system manager or a privileged user of *GW* obtains (*Bu*, *Wu*) of *U* during *U* registration phase, he/she maybe try to impersonate *U* to access the resource in WSNs. The attacking processes are the same as the forgery attack. Thus, the Wenbo's protocol does not provide sufficient protection against an insider attack on *GW* by a privileged user.

No protection against compromise attack

In the Wenbo's protocol, the gateway and the sensor nodes shared a long-term common secret key SK_{GS}. If an attacker captures some a sensor node, he/she can attain the shared secret key SK_{GS} via some methods since the SK_{GS} is not encrypted. So it is very easy to impersonate a sensor node in WSNs. Even the attacker may make many sensor nodes to impersonate the sensor nodes of in WSNs.

Proposed protocol

To solve the security weaknesses of the Wenbo's protocol, we propose a new user authentication protocol for WSNs using identity-based cryptography. First, we review the fundamentals of identity-based cryptography, and then survey the identity-based cryptography which is suitable for our design of a secure authentication protocol for WSNs. In the proposed protocol, *GW* integrates the trusted and reputation scheme [24,26]. The proposed five phases are described in detail later.

Identity-based cryptography

Identity-based cryptography is a kind of public-key based scheme. The public key is the unique identity of the user. The private key is generated by a third party called a Private Key Generator (PKG) with its master secret and user's identity. In the identity-based cryptography system, firstly, the PKG must create a master public key and a master private key. Then any user may use this master public key and also use the user's identity to generate the user's public. The user's private key is created by the PKG with the user's identity.

For every two parties using in identity-based cryptography, it is easy to calculate a shared secret session key between them using its own private key and public key of another party. For example, a sensor node *Sn* with public key Q_{Sn} and private key d_{Sn}, and a user *U* with public key Q_u and private key d_u can calculate their shared secret session key by computing key $= \hat{e}(Q_u, d_{Sn}) = \hat{e}(d_u, Q_{Sn})$.

In the proposed protocol, *GW* is the PKG. *GW* selects a random number $s \in Z_q^*$ that is kept secret. *GW* computes $K_{pub} = s \times P$. This public-private key pair $< K_{pub}, s >$ is the master key pair of *GW*. And *GW* computes $Q_{GW} = H_1(ID_{GW})$, $d_{GW} = s \times Q_{GW}$. Q_{GW} is the authentication public key of *GW*. d_{GW} is the authentication private key of *GW*.

Registration phase

In the registration phase, *Sn* and *U* register to *GW*. The processes are the follow as.

Sensor node registration

In the WSNs, all sensor nodes must register to *GW* before being deployed. *GW* creates a private key for every sensor node. And the system parameters P_{set}, the public functions and the private key are stored in the sensor node. *GW* completes the following steps.

Step 1: Creates the public key Q_{Sn}.

GW uses the identity ID_{Sn} of *Sn* to generate the public key Q_{Sn}, $Q_{Sn} = H_1(ID_{Sn})$.

Step 2: Generates the private key d_{Sn}.

GW uses the master key *s* and the public key Q_{Sn} to create the private key d_{Sn}, $d_{Sn} = s \times Q_{Sn}$.

Step 3: Installs system parameters, public functions and private key of *Sn*.

GW installs the system parameters P_{set}, d_{Sn} and other public functions into *Sn*. That is to say, $\{P_{set}, d_{Sn}, h(.), f(.), H_1(.), e(.)\}$ is stored into the *Sn*.

User registration phase

Before accessing a sensor node in WSNs, any user must register to *GW* and gains a set P_{set} and other parameters. The registration phase is shown in the Figure 1.

Step 1: $U \rightarrow GW$: $\{ID_u, \text{Reg-inf}, T_1\}$.

U sends the register request message $\{ID_u, \text{Reg-inf}, T_1\}$ to *GW* at the time T_1.

Step2: *GW* $\rightarrow U$: $\{ID_{GW}, P, xP, h(.), a_1, T_2\}$.

When receiving the register request message of *U* at the time T′, firstly *GW* checks whether $(T'-T_1) \le \Delta T$ holds. If the answer is no, *GW* rejects the register request message of *U*. Otherwise, *GW* selects a random number $x \in Z_q^*$ and computes $xP = x \times P$. Then *GW* calculates $a_1 = h(ID_{GW}||ID_u||xP||T_2)$, where T_2 is the current time of *GW*. Finally, *GW* sends the register response message $\{ID_{GW}, P, xP, h(.), a_1, T_2\}$ to *U*.

Step 3: $U \rightarrow GW$: $\{ID_u, E_k(PS'), yP, b, T_3\}$.

When receiving the register response message $\{ID_{GW}, P, xP, h(.), a_1, T_2\}$ at the time T′, *U* checks whether $(T'-T_2) \le \Delta T$ holds. If the answer is no, *U* rejects the register response message. Otherwise, *U* computes $a_1' = h(ID_{GW}||ID_u||xP||T_2)$ and checks whether $a_1' = a_1$ holds. If the answer is no, *U* rejects the register response message. Otherwise, *U* picks a random number $y \in Z_q^*$ and computes $yP = y \times P$. And *U* selects a password $PS \in Z_q^*$ and a random number $br \in Z_q^*$. *U* calculates $PS' = h(PS \oplus br)$ and $k = h(y \times xP)$. Then *U* encrypts PS' with the session key *k*, $E_k(PS')$. Finally, *U* computes $b = h(ID_u||ID_{GW}||E_k(PS')||yP||T_3)$, where T_3 is the current times of *U*. And *U* sends a message $\{ID_u, E_k(PS'), yP, b, T_3\}$ to *GW*.

Step 4: $GW \rightarrow U$: $\{ID_{GW}, P_{set}, E_{\Theta}(\Theta, \mathrm{M}), a_2, T_4\}$.

Receiving the message $\{ID_u, E_k(PS'), yP, b, T_3\}$ at the time T′, *GW* firstly checks whether $(T'-T_3) \le \Delta T$ holds. If the answer is no, *GW* rejects this message. Otherwise, *GW* computes $b' = h(ID_u||ID_{GW}||E_k(PS')||yP||T_3)$ and

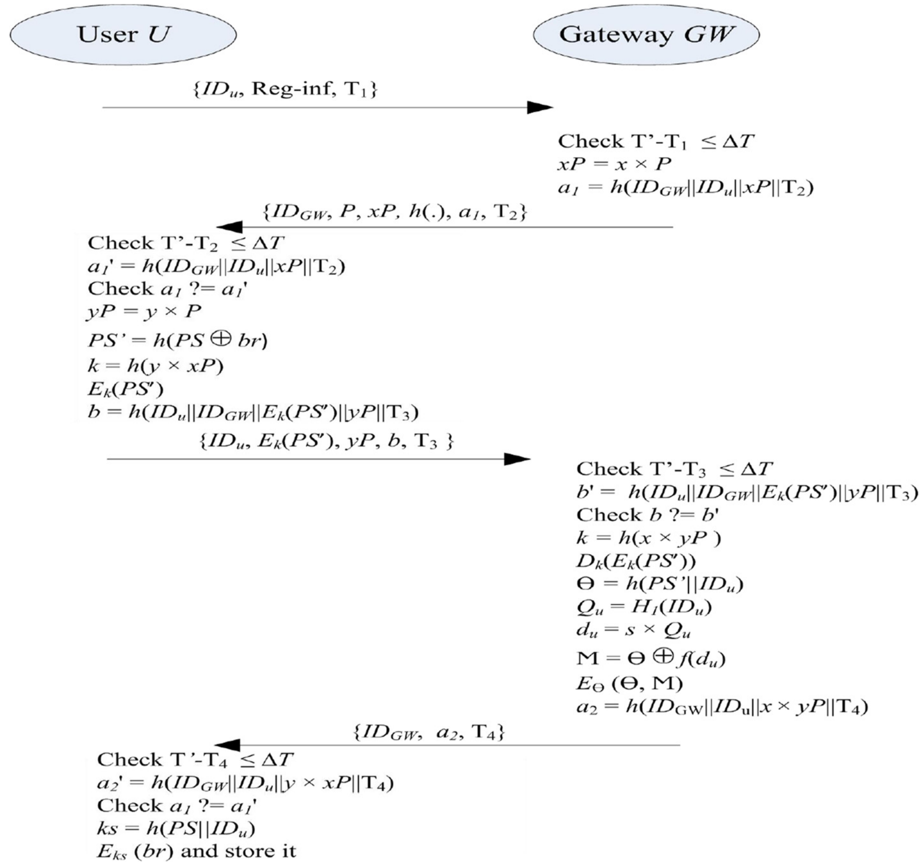

Figure 1 Registration phase of the user.

checks whether $b' = b$ holds. If the answer is no, *GW* rejects this message. Otherwise, *GW* generates the session key k and decrypts $E_k(PS')$, $k = h(x \times yP)$, $D_k(E_k(PS'))$ to gain PS'. Then *GW* computes $\Theta = h(PS'||ID_u)$, $Q_u = H_1(ID_u)$ and $d_u = s \times Q_u$. And *GW* also calculates $\mathrm{M} = \Theta \oplus f(d_u)$. *GW* encrypts the (Q_u, M), $E_\Theta(\Theta, \mathrm{M})$ and computes $a_2 = h(ID_{GW}||ID_u||xyP||T_4)$. At last *GW* stores ($P_{set}$, $E_\Theta(\Theta, \mathrm{M})$, $h(.)$, $f(.)$, $H1(.)$, $\hat{e}(.)$) into a smart card that is sent to *U*. Moreover *GW* sends the register acknowledge message $\{ID_{GW}, a_2, T_4\}$ to *U*.

Step 5: *U* encrypts and stores *br*.

When receiving the register acknowledge message $\{ID_{GW}, a_2, T_4\}$ at the time T', *U* firstly checks whether $(T'-T_4) \le \Delta T$ holds. If the answer is no, *U* rejects this message. Otherwise *U* computes $a_2' = h(ID_{GW}||ID_u||yxP||T_4)$ and checks whether $a_2' = a_2$ holds. If the answer is no, *U* rejects this message. Otherwise, *U* computes $ks = h(PS||ID_u)$ and encrypts *br*, $E_{ks}(br)$. Finially *U* stores $E_{ks}(br)$.

Login phase and authentication phase

Accessing the data in *Sn*, *U* must login *Sn* and be authenticated by *GW* and *Sn*. And *U* must complete the login phase and authentication phase. Login phase and authentication phase are shown in Figure 2.

Login phase

U must enter his ID_u and password *PS* firstly. Then, after the smart card validates *U* via the following steps, the smart card sends the access request message to *Sn*.

Step 1: Gains *br*.

U enters his identity ID_u and password *PS* to the smart terminal. And the smart terminal computes $ks = h(PS||ID_u)$, and $D_{ks}(E_{ks}(br))$ to gain *br*.

Step 2: Validate *U*.

The smart card computes $PS' = h(PS \oplus br)$, $\Theta' = h(PS'||ID_u)$ and $D_{\Theta'}(E_\Theta(\Theta, \mathrm{M}))$ to gain the (Θ, M). The smart card checks whether $\Theta = \Theta'$ holds. If the answer is no, the smart card stops and alarms. Otherwise, the smart card continues to execute the next step.

Step 3: Computes Q_{Sn}, Q_{GW}, d_u, X and Y.

$$Q_{Sn} = H_1(ID_{Sn}),\ \ Q_{GW} = H_1(ID_{GW}),\ \ d_u = H_1(\mathrm{M}\oplus\theta),$$
$$X = \hat{e}(d_u, Q_{Sn}) and Y = \hat{e}(d_u, Q_{GW}).$$

Step 4: Generates a, b and encrypts (a, b).

The smart card calculates $a = h(ID_u||ID_{GW}||Y||T_u)$, $b = h(ID_u||ID_{Sn}||X||a||T_u)$ and $E_X(a, b)$, where T_u is the current time of the smart terminal system.

Step 5: $U \rightarrow Sn$: $\{ID_u, ID_{Sn}, E_X(a, b), T_u\}$.

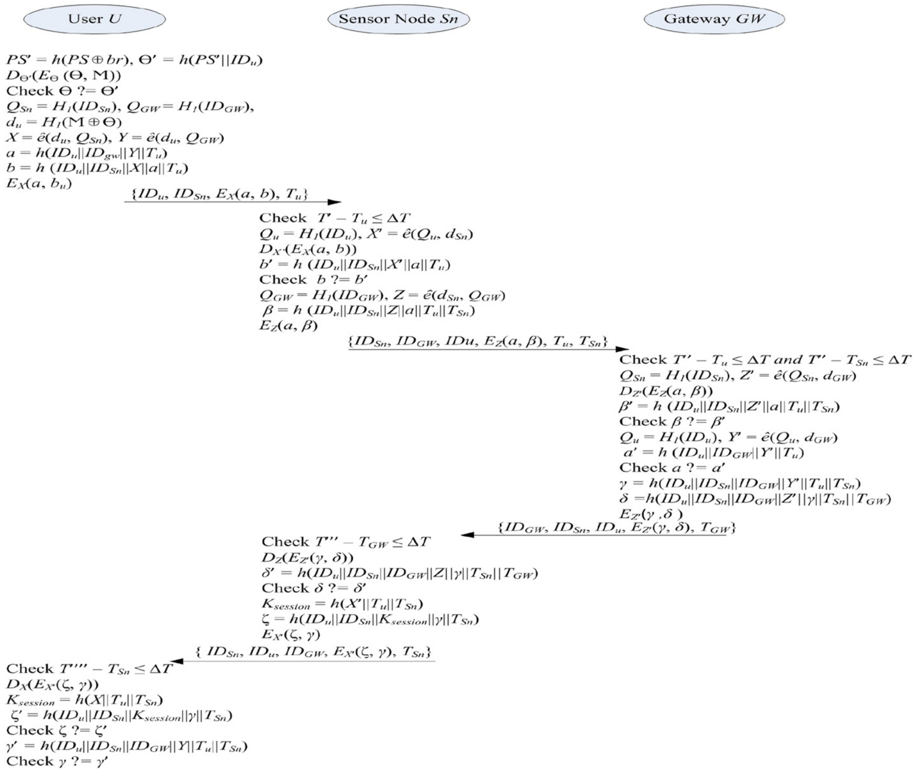

Figure 2 Login phase and authentication phase.

The smart card sends the login request message $\{ID_u, ID_{Sn}, E_X(a, b), T_u\}$ to the *Sn*.

Authentication phase

The authentication phase includes *Sn* authenticating *U* and *GW*, *GW* authenticating *Sn* and *U*, *U* authenticating *Sn* and *GW*. The authentication phase can complete the mutual authentication.

Sensor node *Sn* authenticates user *U* When receiving the login request message $\{ID_u, ID_{Sn}, E_X(a, b), T_u\}$ sent by *U* at time T', *Sn* firstly checks the validity of the request message. Then *Sn* authenticates *U*.

Step 1: Validates login request message.

Sn checks whether $(T'-T_u) \le \Delta T$ holds. If the answer is no, *Sn* rejects the login request of *U*. Otherwise, it continues to perform the next step.

Step 2: Decrypts $E_X(a, b)$.

S_n computes $Q_u = H_1(ID_u)$, $X' = \hat{e}(Q_u, d_{Sn})$ and $D_{X'}(E_X(a, b))$ to gain (a, b).

Step 3: Computes $b' = h(ID_u||ID_{Sn}||X'||a||T_u)$.

Step 4: Validates *U*.

S_n checks if $b = b'$ holds. If the answer is yes, the validity of *U* can be assured and *Sn* continues to perform the next step. Otherwise, it rejects the login request message of *U*.

Step 5: Computes Q_{GW}, Z, β and encrypts.

$Q_{GW} = H_1(ID_{GW})$, $Z = \hat{e}(d_{Sn}, Q_{GW})$ and $\beta = h(ID_u||ID_{Sn}||Z||a||T_u||T_{Sn})$, where T_{Sn} is the current time of *Sn* system. And *Sn* encrypts (a, β), $E_Z(a, \beta)$.

Step 6: $S_n \rightarrow GW$: $\{ID_{Sn}, ID_{GW}, IDu, E_Z(a, \beta), T_u, T_{Sn}\}$

Sn sends a request message $\{ID_{Sn}, ID_{GW}, ID_u, E_Z(a, \beta), T_u, T_{Sn}\}$ to *GW*.

Gateway *GW* authenticates sensor node *Sn* When receiving the request message $\{ID_{Sn}, ID_{GW}, ID_u, E_Z(a, \beta), T_u, T_{Sn}\}$ at time T'', *GW* checks the validity of this message firstly. And *GW* authenticates *Sn* and *U*. Finally, *GW* creates a response message for *Sn* and *U*.

Step 1: Validates request message of *Sn*.

GW checks whether $(T''-T_u) \le \Delta T$ and $(T''-T_{Sn}) \le \Delta T$ hold. If the answer is no, *GW* rejects the request message. Otherwise, *GW* continues to perform the next step.

Step 2: Computes Q_{Sn}, Z' and gains (a, β).

GW computes $Q_{Sn} = H_1(ID_{Sn})$, $Z' = \hat{e}(Q_{Sn}, d_{GW})$ and $D_{Z'}(E_Z(a, \beta))$ to gain (a, β).

Step 3: Computes $\beta' = h(ID_u||ID_{Sn}||Z'|a||T_u||T_{Sn})$.

Step 4: Validates *Sn*.

GW checks if $\beta' = \beta$ holds. If the answer is yes, the validity of *Sn* can be assured and *GW* continues to perform the next step. Otherwise, it rejects the request message.

Step 5: Computes Q_u, Y' and a'.

GW computes $Q_u = H_1(ID_u)$, $Y' = \hat{e}(Q_u, d_{GW})$ and $a' = h(ID_u||ID_{GW}||Y'||T_u)$.

Step 6: Validates U.

GW checks if $a' = a$ holds. if the answer is yes, the validity of U can be assured and GW continues to perform the next step. Otherwise, GW rejects the request message.

Step 7: $GW \rightarrow Sn$:$\{ID_{GW}, ID_{Sn}, ID_u, E_{Z'}(\gamma, \delta), T_{GW}\}$.

GW generates the response message for Sn and U. GW calculates: $\gamma = h(ID_u||ID_{Sn}||ID_{GW}||Y'||T_u||T_{Sn})$ and $\delta = h(ID_u||ID_{Sn}||ID_{GW}||Z'||\gamma||T_{Sn}||T_{GW})$, where T_{GW} is the current time of GW's system. And GW encrypts (γ, δ) with the key Z', $E_{Z'}(\gamma, \delta)$, and sends the response message $\{ID_{GW}, ID_{Sn}, ID_u, E_{Z'}(\gamma, \delta), T_{GW}\}$ to Sn.

Sensor node *Sn* authenticates gateway *GW* When receiving the response message $\{ID_{GW}, ID_{Sn}, ID_u, E_{Z'}(\gamma, \delta), T_{GW}\}$ sent by GW at time T''', Sn checks and authenticates GW via the following steps.

Step 1: Validates the response message.

Sn checks if $(T'''\text{-}T_{GW}) \leq \Delta T$ holds. If the answer is no, Sn rejects this response message. Otherwise, Sn continues to perform the next step.

Step 2: Gains (γ, δ).

Sn decrypts the $E_{Z'}(\gamma, \delta)$ with the key Z, $D_Z(E_{Z'}(\gamma, \delta))$, to gain (γ, δ).

Step 3: Computes δ'.

$$\delta' = h(ID_u||ID_{Sn}||ID_{GW}||Z||\gamma||T_{Sn}||T_{GW}).$$

Step 4: Validates GW.

Sn checks if $\delta' = \delta$ holds. If th answer is yes, the validity of GW can be assured and S_n continues to execute the next step. Otherwise, it rejects the response message.

Step 5: Generates $K_{session}$, ζ and encrypts.

Sn computes $K_{session} = h(X||T_u||T_{Sn})$,

$$\zeta = h(ID_u||ID_{Sn}||K_{session}||\gamma||T_{Sn}) \text{ and } E_{X'}(\zeta, \gamma).$$

Step 6: $Sn \rightarrow U$: $\{ID_{Sn}, ID_u, ID_{GW}, E_{X'}(\zeta, \gamma), T_{Sn}\}$.

Sn sends the response message $\{ID_{Sn}, ID_u, ID_{GW}, E_{X'}(\zeta, \gamma), T_{Sn}\}$ to U.

User *U* authenticates sensor node *Sn* When U receives Sn's response message $\{ID_{Sn}, ID_u, ID_{GW}, E_{X'}(\zeta, \gamma), T_{Sn}\}$ at time T'''', U checks this message and authenticates Sn and GW. U performs the following steps.

Step 1: Validates the response message.

U checks whether $(T''''\text{-}T_{Sn}) \leq \Delta T$ holds. If the answer is no, U rejects this response message. Otherwise, it continues to perform the next step.

Step 2: Gains (ζ, γ).

U computes $D_X(E_{X'}(\zeta, \gamma))$ to decrypt $E_{X'}(\zeta, \gamma)$ with the key X to gain (ζ, γ).

Step 3: Generates $K_{session}$ and ζ'.

U computes $K_{session} = h(X||T_u||T_{Sn})$,
and $\zeta' = h(ID_u||ID_{Sn}||K_{session}||\gamma||T_{Sn})$

Step 4: Validates Sn.

U checks whether $\zeta = \zeta'$ holds. If the answer is yes, the validity of Sn can be assured and U continues to execute the next step. Otherwise, U rejects the response message.

Step 5: Computes $\gamma' = h(ID_u||ID_{Sn}||ID_{GW}||Y||T_u||T_{Sn})$.

Step 6: Validates GW.

U checks whether $\gamma' = \gamma$ holds. If the answer is yes, the validity of GW can be assured and U accepts this response message. Otherwise, U rejects this response message.

After U authenticates Sn and GW, U will access the data of the Sn with the session key $K_{session}$.

Password update phase

When U updates his password, U enters his ID_u, old password PS and news password PSn to the smart terminal or a update password program. The smart card must compute a new password value, which is encrypted and stored in the smart card. The user password update phase includes the following steps.

Step 1: U enters his ID_u, old password PS and news password PSn to the smart terminal or a update password program.

Step 2: The smart terminal computes $ks = h(PS||ID_u)$ and $D_{ks}(E_{ks}(br))$ to gain br firstly. Then it computes $PS' = h(PS \oplus br)$, $PSn' = h(PSn \oplus br)$. The smart terminal sends $\{ID_u, PS', PSn'\}$ to the smart card.

Step 3: The smart card computes $\Theta' = h(PS'||ID_u)$ and $D_{\Theta'}(E_{\Theta}(\Theta, \mathrm{M}))$ to gain (Θ, M).

Step 4: The smart card checks whether $\Theta' = \Theta$ holds. If the answer is no, the smart card rejects the password update and alarms. Otherwise, the smart card continues to perform the next step.

Step 5: The smart card calculates $\Theta_n' = h(PSn'||ID_u)$ and $\mathrm{M}' = \Theta_n' \oplus (\Theta \oplus \mathrm{M})$.

Step 6: The smart card encrypts the new sensitive password value (Θ_n', M') with the key Θ_n', $E_{\Theta n'}(\Theta_n', \mathrm{M}')$, and replaces the $E_{\Theta}(\Theta, \mathrm{M})$ with $E_{\Theta n'}(\Theta_n', \mathrm{M}')$.

Security and performance analysis

The proposed protocol provides message confidentiality service

Proof

Message confidentiality service against eavesdropping attack is performed by data encryption service. Our proposed protocol can provide sufficient confidentiality for sensitive data stored and transmitted with encrypting data (e.g. $E_k(PS')$, $E_{\Theta}(\Theta,\mathrm{M})$, $E_X(a, b)$, $E_Z(a, \beta)$, $E_{Z'}(\gamma, \delta)$ and $E_{X'}(\zeta, \gamma)$. More specifically, these sensitive information are confidential against the attacker. If the sensitive data is stored or transmitted without encryption in the public channel , the attacker maybe view the plaintext data. This attack maybe occur in Wenbo's protocol and Yoon and Yoo's protocol [15]. Moreover, in Wenbo's protocol the sensitive (*Bu*, *Wu*) that was not encrypted was stored in the smart card and the long-term shared secret key SK_{GW} was not also encrypted in the *Sn*. In the [15] *Sn*'s response message that was not encrypted was sent to *U* by a public channel directly.

The proposed protocol resists an integrity attack

Proof

The data integrity attack includes data modification attack, data corruption attack and data insertion attack. The integrity service assures the transmitted data is not modified by an unauthorized entity.

In our proposed protocol, *Sn* can guarantee the login request message $\{ID_u, ID_{Sn}, E_X(a, b), T_u\}$ from *U* has not been modified by an unauthorized entity via decrypting $E_X(a, b)$, recomputing and checking *b*. *GW* can also guarantee the authentication request message $\{ID_{Sn}, ID_{GW}, IDu, E_Z(a, \beta), T_u, T_{Sn}\}$ from *Sn* has not been modified by an unauthorized entity via decrypting $E_Z(a, \beta)$, recomputing and checking a, β. Similarly, *Sn* can guarantee the authentication response message $\{ID_{GW}, ID_{Sn}, ID_u, E_{Z'}(\gamma, \delta), T_{GW}\}$ from *GW* has not been modified by an unauthorized entity via decrypting $E_{Z'}(\gamma, \delta)$, recomputing and checking δ. Moreover, *U* uses the same way to guarantee the authentication response message $\{ID_{GW}, ID_{Sn}, ID_u, E_{X'}(\zeta, \gamma), T_{Sn}\}$ from *Sn* has not been modified.

The proposed protocol resists a denial attack

Proof

This type of attack is that the participating entity denies in all of the operations or part of its. However, in our proposed protocol, we assume that *GW* is a trusted party. And *GW* creates the unique private key for every entity (sensor node, user) . Although *GW* does not store the private key of an entity, it can trace the entity operations with the entity's public key and HMAC. Therefore, the entity cannot deny that he/she performed all participation.

The proposed protocol resists a DoS Attack

Proof

The DoS attack can be occurred by the attacker who transmitting the large number of request messages to *Sn* or *GW* in the login phase or in the authentication phase. In our proposed protocol, since every message associates with a timestamp *T* and is authenticated, the unauthenticated message or the timeout message is rejected. So the proposed protocol can reject DoS attack.

The proposed protocol resists a sensor node compromise attack

Proof

Since WSNs is normally deployed in an open environment, the attacker is easy to capture a sensor node and may attempt to get some information stored in the sensor node. When the attacker gets the secret from the capturing sensor node, he/she can attack the WSNs. If the authenticating user and data access from the sensor node are allowed directly to the user without the license of gateway, this attack is very high, which occurs in Watro et al.'s scheme [19].

In our proposed protocol, And *U* does not access data from *Sn* until it is authorized by *GW* and *Sn*. And *U*'s request message must be authenticated by *Sn* firstly, and the request message must be authenticated by *GW*. After that *GW* sends the license of *U*'s to *Sn* and *U*. Only *U* can access the data of sensor node when his/her license from *GW* is the same as *Sn*'s from *GW*. Moreover, in our proposed protocol *GW* can monitor whether a sensor node is captured with the trusted and reputation management scheme [24,26]. If some a sensor node is captured by an attacker, *GW* can detect and isolate it.

The proposed protocol resists a replay attack

Proof

The replay attacks are impossible if the previous information is not reused again. In our proposed protocol, the login message and the authentication message are validated by checking timestamps. When an attacker eavesdrops the communication between *U* and *Sn* or between *Sn* and *GW*, he/she does not reusable again. We assume if an adversary intercepts a login request message $\{ID_u, ID_{Sn}, E_X(a, b), T_u\}$ and attempts replaying the same message for login to *Sn*. The verification of the login request fails because of $(T_a\text{-}T_u) > \Delta T$, where T_a denotes the time when *Sn* receives the replaying message. Similarly, if an adversary intercepts $\{ID_{Sn}, ID_{GW}, ID_u, E_Z(a, \beta), T_u, T_{Sn}\}$and attempts to replay it to *GW*, he/she cannot pass the verification of *GW* because the time expires (i.e. $(T_b\text{-}T_{Sn}) > \Delta T$ and $(T_b \text{-}T_u) > \Delta T$), where T_b denotes the time when the replaying message is received by *GW*. Also if an adversary intercepts $\{ID_{GW}, ID_{Sn}, ID_u, E_{Z'}(\gamma, \delta), T_{GW}\}$ and attempts replaying the same message to *Sn*, he/she cannot pass the verification of *Sn* because of $(T_c\text{-}T_{GW}) > \Delta T$, where T_c denotes the time when *Sn* receives the replaying response message. Moreover, if an adversary intercepts $\{ID_{GW}, ID_{Sn}, ID_u, E_{X'}(\zeta, \gamma), T_{Sn}\}$ and attempts replaying the same message to *U*, he/she also cannot pass the verification of *U* because of $(T_d\text{-}T_{Sn} > \Delta T)$, where T_d denotes the time when *U* receives the replaying response message.

The proposed protocol resists an impersonation attack

Proof

In our proposed protocol, all sensitive information that is transmitted is encrypted with some a key. Additionally, the messages are validated and authenticated. Only when an attacker knows the master key *s* or solves

Bilinear Differ-Hellman Problem can he/she attain the private key. It is impossible for an attacker.

In the login phase, only when an attacker knows *U*'s private key d_u can he/she generate a legal login request message $\{D_u, ID_{Sn}, E_X(a, b), T_u\}$ to impersonate the *U*. Moreover it is impossible that an attacker gains the sensitive key material (Θ, M) that is encrypted to only be stored in the smart card without the user *U*'s password. Thus it is not possible to compute X without d_u for an attacker. And as long as an attacker does not possess *Sn*'s private key d_{Sn}, he/she cannot generate a legal authentication request message $\{ID_{Sn}, ID_{GW}, IDu, E_Z(a, \beta), T_u, T_{Sn}\}$ and $\{ID_{GW}, ID_{Sn}, ID_u, E_{X'}(\zeta, \gamma), T_{Sn}\}$ to impersonate *Sn*. This is because that the attacker cannot compute the key Z and the key X' without d_{Sn}. Similarly, an attacker also cannot generate a legal response message $\{ID_{GW}, ID_{Sn}, ID_u, E_{Z'}(\gamma, \delta), T_{GW}\}$ to impersonate *GW*. This is due to that an attacker does not know the private key d_{GW} of *GW*.

The proposed protocol resists a stolen verifier attack

Proof

An attacker who has stolen *U*'s private key materials $E_\Theta(\Theta, \mathrm{M})$ from the smart terminal or the smart card via the Trojan or other intruding methods cannot obtain any useful information. This is due to that the private key materials are encrypted. The attacker cannot decrypt $E_\Theta(\Theta, \mathrm{M})$ to gain (Θ, M) without *U*'s password *PS*. And the attacker also cannot attain any useful private key information of *U* from *GW* because *U*'s private key materials are not stored in the *GW* database.

The proposed protocol resists a stolen smart card attacks

Proof

The attacker who has stolen *U*'s smart card cannot impersonate this user to access S_n. Because the attacker does not know *U*'s password, the smart card does not validate the login request and rejects the access request of the attacker.

The proposed protocol resists an insider attack

Proof

The insider attack is intentionally misused by authorized entities. In our proposed protocol, the gateway manager or system administrator cannot attain *U*'s password *PS* because in the registration phase *U* transmits $E_k(PS')$ to *GW* instead of the plain password *PS*, and any sensitive key material information of *U* and any verifier table are not stored in *GW*. Additionally, the smart terminal manager or administrator also cannot attain the useful information of *U*'s key from the smart card and the smart terminal because of the sensitive key material encrypted. Therefore, the proposed protocol can resist the privileged insider attacks.

The proposed protocol resists a man-in-the-middle attack

Proof

The man-in-the-middle attack is that an attacker intercepts the communication between the legal user and other entity (e.g. sensor node, gateway) and successfully masquerades as the user or other entity by some methods. In our proposed protocol, *U* is authenticated by *Sn* in the login phase, *Sn* and *U* are authenticated by *GW* in the authentication

Table 2 Security comparison

	Benenson et al. [7]	Das [10]	Chen and Shih [16]	Yuan et al. [19]	Yeh et al. [17]	Yoon and Yoo [20]	Ohood et al. [21]	Wenbo and Peng [22]	Ours
Data Confidentiality	NP	NP	NP	NP	NP	NP	P	NP	P
Data Integrity	NP	P	P	NP	NP	P	P	P	P
Password Update	NR	NP	R	NP	P	NR	NR	P	P
Key Agreement	NP	NP	NP	NP	NP	NP	P	P	P
Mutual Authentication	NP	NP	P	NP	NP	P	P	NP	P
Denial Attack	No	No	No	Yes	No	Yes	Yes	No	Yes
DoS Attack	No	No	No	No	No	No	Yes	No	Yes
Compromise Attack	Yes	No	No	No	No	No	Yes	Yes	Yes
Replay Attack	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes
Impersonation Attack	No	Yes	Yes	No	No	Yes	Yes	No	Yes
Insider Attack	Yes	No	No	No	No	Yes	Yes	Yes	Yes
Forgery Attack	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes
Stolen-Verifier Attack	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
Guessing Attack	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Man-in-the-Middle Attack	No	No	Yes	No	No	Yes	Yes	No	Yes

Yes: Resist Attack, No: Not Resist Attack, P: Provided, NP: Not Provided, R: Required, NR: Not Required.

Table 3 Computation performance comparison

	Benenson et al. [7]	Das [10]	Chen and Shih [16]	Yeh et al. [17]	Yoon and Yoo [20]	Ohood et al. [21]	Yuan et al.[19]	Wenbo et al. [22]	Ours
Registration Phase	$1Texp$	$1Th$	$1Th$	$4Th + 2Tmp$	$3Th$	$2Th$	$4Th$	$3Th + 1Tpm$	$4Th + 4Tpm + 3Taes$
Login and Authentication Phase	$2nTh + 3nTexp$	$5Th$	$7Th$	$11Th + 4Tpa + 8Tpm + 2Te$	$10Th$	$4Trc + 8Th$	$9Th$	$15Th + 6Tpm$	$14Th + 6Tpair + 8Taes$
Total	$2nTh + 3nTexp + 1Texp$	$6Th$	$8Th$	$15Th + 4Tpa + 10Tpm + 2Te$	$13Th$	$4Trc + 10Th$	$13Th$	$18Th + 7Tpm$	$18Th + 4Tpm + 11Taes + 6Tpair$

request phase, and Sn also authenticates GW in the authentication response phase, U validates Sn and GW in the authentication response phase. That is to say, our proposed protocol can provide complete mutual authenticate among entities and resists the man-in-the-middle attack.

Table 2 shows the security functionality comparisons between our proposed protocol and the related protocols. According to the Table 2, although the Ohood et al.'s protocol presents the same security as ours, the Ohood et al.'s protocol needs some complicated biometric equipments. Compared against each other, our protocol provides is more security services than the other protocols.

Performance analysis The section summarizes the performance results of the proposed protocol. We define the notation Th as the hash function computation cost, $Texp$ as the modular exponential computation cost, Tpm as the elliptic curve point multiply cost, Tpa as the elliptic curve point addition cost, $Tpair$ as pairing computation cost, Trc as $RC5$ computation cost, $Taes$ as AES computation cost, Te as the elliptic curve polynomial computation cost. The comparison of related protocols is illustrated in the Table 3.

According to Table 3, Chen et al.'s protocol needs eight hash function computations, Yoon el at.'s needs thirteen hash function computations, Yuan et al.'s also need thirteen hash function computations, Das's protocol needs six hash function computations. And Benenson et al.'s protocol needs 2n hash function computations and $3n + 1$ modular exponential computations [22]. Ohood et al.'s biometric authentication protocol needs four RC5 computations and ten hash function computations. Yeh et al.'s protocol needs fifteen hash function computations, four elliptic curve point addition computations , ten elliptic curve point multiply computations and two elliptic curve polynomial computations. Wenbo et al.'s protocol needs eighteen hash function computations and seven elliptic curve point multiply computations. Our proposed protocol needs eighteen hash function computations, four elliptic curve point multiply computations, eleven AES computations and six pairing computations. Although our protocol needs more computations than their protocols, their protocols suffer from security issues or need complicated biometric equipments. Our protocol addressed these issues and provides better security and more security services than the other related protocols.

Conclusion

In the paper, we discussed an approach of data capturing for big data that is data collecting via sensor networks and its user authentication protocol. We have analyzed Wenbo et al.'s user authentication protocol for WSNs. The Wenbo's protocol, which does not provide mutual authentication between user and sensor node and confidentiality service, is susceptible to insider, replay, denial, compromise, forgery, man-in-the-middle and DoS attacks. We have also reviewed the protocols of Yeh et al., which does not provide mutual authentication and protect against insider, denial, compromise, man-in-the-middle and DoS attacks, of Das, which is vulnerable to forgery, denial, compromise, DoS, man-in-the-middle attacks, of Benenson et al., which susceptible to denial, compromise, DoS, man-in-the-middle attacks, of Chen et al. which is vulnerable to denial, insider, compromise and DoS attacks, of other biometric authentication protocols. Since WSNs need more secure mutual authentication method in an insecure network environment, we use the IBE mechanism to design a news user authentication protocol. Our protocol can prevent all the problems of the former schemes. Furthermore, it enhances the WSNs authentication with higher security than the other protocol. Therefore, the protocol is more suited to open and higher security WSNs environment in despite of more computation cost.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

ZQ designed the user authentication protocol and analyzed performance. TC implemented the security analysis for protocol. ZX analyzed the security of Wenbo's protocol. RC coordinated the whole study. All authors read and approved the final manuscript.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant No. 11271003, the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No.20134410110003, High Level Talents Project of Guangdong, Guangdong Provincial Natural Science Foundation under Grant No.S2012010009950 and No. S2012040007370, the Project of Department of Education of Guangdong Province under Grant No 2013KJCX0146, and the Natural Science Foundation of Bureau of Education of Guangzhou under Grant No. 2012A004. We sincerely thank all the researchers in our references section for the inspiration they provide.

Author details

[1]Key Laboratory of Mathematics and Interdisciplinary Sciences of Guangdong Higher Education Institutes, Guangzhou University, Guangzhou, China. [2]School of Mathematics and Information Science, Guangzhou University, Guangzhou, China. [3]College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China. [4]Faculty of Science and Technology, University of Stavanger, Stavanger, Norway.

Received: 15 September 2014 Accepted: 16 February 2015

References

1. Kenneth NC, Viktor MS (2013) The rise of big data: how It's changing the way we think about the world. Fortuna's Corner in cloud computing, Cybersecurity, Dow, Intelligence Community, Internet, Markets, national security, S & P, Uncategorized, US Military, April 24, 2013
2. Sastry N, Wagner D (2004) Security considerations for IEEE 802.15.4 networks. In Proceedings of the ACM Workshop on Wireless Security (WiSe'04). 32–42
3. WG802.15 (2003) 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. IEEE, Washington, DC, USA
4. Das ML, Saxena A, Gulati VP (2004) A dynamic ID-based remote user authentication scheme. IEEE Trans Consum Electron 50(2):629–31
5. Leung KC, Cheng LM, Fong AS, Chan CK (2003) Cryptanalysis of a modified remote user authentication scheme using smart cards. IEEE Trans Consum Electron 49(4):1243–5
6. Watro R, Kong D, Cuti S F, Gardiner C, Lynn C, Kruus P (2004) TinyPK: securing sensor networks with public key technology. In Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'04), 59–64
7. Benenson Z, Gedicke N, Raivio O (2005) Realizing robust user authentication in sensor networks. In Real-World Wireless Sensor Networks (REALWSN), 14
8. Wong KHM, Yuan Z, Jiannong C, Shengwei W (2006) A dynamic user authentication scheme for wireless sensor networks. In Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, 244-251
9. Tseng HR, Jan RH, Yang W (2007) An improved dynamic user authentication scheme for wireless sensor networks. In Proceedings of the 50th Annual IEEE Global Telecommunications Conference (GLOBECOM'07), 986–990
10. Das ML (2009) Two-factor user authentication in wireless sensor networks. IEEE Trans Wirel Commun 8(3):1086–90
11. Ko KC (2008) A novel dynamic user authentication scheme for wireless sensor networks. In Proceedings of the IEEE International Symposium on Wireless Communication Systems, ISWCS'08, 608–612
12. Nyang DH, Lee MK (2009) Improvement of Das's two-factor authentication protocol in wireless sensor networks. Available via DIALOG. http://eprint.iacr.org/2009/631.pdf. Accessed 15 Jan 2014.
13. Vaidya B, Rodrigues JJ, Park JH (2010) User authentication schemes with pseudonymity for ubiquitous sensor network in NGN. Int J Communication Syst 23(9–10):1201–22
14. Khan MK, Alghathbar K (2010) Cryptanalysis and security improvements of 'two-factor user authentication in wireless sensor networks'. Sensors 10(3):2450–9
15. Khan MK, Alghathbar K (2010), Security Analysis of Two-Factor Authentication In Wireless Sensor Networks. In Proceedings of Advances in Computer Science and Information Technology: AST/UCMA/ISA/ACN 2010 Conferences, 55–60
16. Chen TH, Shih WK (2010) A robust mutual authentication protocol for wireless sensor networks. ETRI J 32(5):704–12
17. Yeh HL, Chen TH, Liu PC, Kim TH, Wei HW (2011) A secured authentication protocol for wireless sensor networks using elliptic curves cryptography. Sensors 11(5):4767–79
18. Han W (2013), Weakness of a secured authentication protocol for wireless sensor networks using elliptic curves cryptography. Available via DIALOG. http://eprint.iacr.org/2011/293. Accessed 15 May 2014.
19. Yuan J, Jiang C, Jiang Z (2010) A biometric-based user authentication for wireless sensor networks. Wuhan Univ J Nat Sci 15(3):272–6
20. Yoon EJ, Yoo K Y(2011) A new biometric-based user authentication scheme without using password for wireless sensor networks. In Proceedings of the 20th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 279–284
21. Ohood A, Mznah AR, Abdullah AD (2013) An efficient biometric authentication protocol for wireless sensor networks. Int J Distributed Sensor Networks 4:1–13
22. Wenbo S, Peng G (2013) A new user authentication protocol for wireless sensor networks using elliptic curves cryptography. Int J Distrib Sens Netw 3:1–7
23. Quan Z (2011) Trusted transmission model of wireless sensor networks, Ph. d. Theis. South China Agricultural University, China
24. Quan Z, Gui F, Deqin X, Jiuhao L (2010) trusted transport model based cluster-merkle-tree for WSNs. in Processing of 2010 IEEE International Conference on Computer Application and System Modeling V1, 564 -568
25. Quan Z, GUI F, Deqin X, Yi T (2012) Trusted architecture for farmland WSNs. in Processing of 2012 Forth IEEE International Conference on Cloud Computing Technology and Science, 782–787
26. Boneh D, Franklin M (2001) Identity based encryption from the Weil pairing. in processing of Advances in Cryptology. Lect Notes Computer Sci 2139:213–29