sensors

MDPI

Article

A Lightweight Anonymous Client–Server Authentication Scheme for the Internet of Things Scenario: LAuth

Yuwen Chen *, José-Fernán Martínez, Pedro Castillejo and Lourdes López

Departamento de Ingeniería Telemática y Electrónica (DTE), Escuela Técnica Superior de Ingeniería y, Sistemas de Telecomunicación (ETSIST), Universidad Politécnica de Madrid (UPM), C/Nikola Tesla, s/n, 28031 Madrid, Spain; jf.martinez@upm.es (J.-F.M.); pedro.castillejo@upm.es (P.C.); lourdes.lopez@upm.es (L.L.)

* Correspondence: yuwen.chen@upm.es; Tel.: +34-913-365-526

Received: 2 October 2018; Accepted: 26 October 2018; Published: 30 October 2018

Abstract: The Internet of Things (IoT) connects different kinds of devices into a network, and enables two-way communication between devices. A large amount of data are collected by these devices and transmitted in this network, it is necessary to ensure secure communications between these devices, to make it impossible for an adversary to undermine this communication. To ensure secure communication, many authentication protocols have been proposed, in this study, a fully anonymous authentication scheme for the Internet of things scenario has been proposed, it enables the remote client to anonymously connect to the server and being serviced by the server. The proposed scheme has been verified by AVISPA and BAN Logic, and the result shows that it is safe. Besides, the simulation shows that the proposed scheme is more efficient in computation cost and communication cost.

Keywords: mutual authentication; lightweight authentication; internet of things; elliptic curve cryptography; user anonymity; IoT security and privacy

1. Introduction

The Internet of Things is a network that connects all kinds of sensors, actuators, and other embedded devices. These devices can exchange data remotely via the network. A significant amount of data are collected by these devices and transmitted in this network. Among these data, there are many personal data, for example, blood pressure, pulse, and electrocardiogram, as well as home environment data, home humidity, and home temperature, etc. People are reluctant to let any party use the data without authorization. There is a need for an authentication scheme to make sure that the data is only accessible to authorized members. Authentication schemes have been studied in the past to solve this problem.

However, in some cases, mutual authentication is not sufficient for protecting the privacy of the clients. In the healthcare environment, an adversary can eavesdrop the information flow and find out which patient's data is being transmitted. The client's medical condition is revealed in this way. In this study, a light weighted authentication and key establishment scheme was proposed, which enables the remote client to be authenticated anonymously by the server. In the proposed scheme, we only used some light weighted security operations: XOR operations, hash functions and a minimal amount of asymmetric encryptions to fulfill perfect forward secrecy, as discussed in the previous work, these operations are relatively light weighted ones, we will continue to discuss this problem in Section 7.1. As energy consumption is of paramount importance in the context where energy are provided by small batteries, there is a high demand for a lightweight authentication scheme [1,2]. For these two reasons, we come up with this authentication scheme. Our contributions are mainly three-fold:

1. We propose a lightweight anonymous authentication for the Internet of things scenario; the scheme achieves various security features: perfect forward privacy, user anonymity, resistance to an offline dictionary attack, etc. In addition, to verify the security features of the proposed scheme, the proposed scheme is also verified by AVISPA and the BAN Logic.
2. We specially design the password changing phase, making it more efficient compared to that in the related works.
3. We simulate the proposed scheme and other related schemes using C++. The results show the communication cost and the computation cost are reduced compared with related proposals.

In Section 2, we discussed the related works, in Section 3, we introduced the proposed scheme, Sections 4 and 5 are security analyses using AVISPA and BAN logic, Section 6 is the formal security analysis section. In Section 7, we compared the proposed scheme with related works. In Section 8, we analyzed the security features. Section 9 is the conclusion part.

2. Related Work

Tu et al. proposed an authentication protocol based on a smart card; the protocol is a two-factor authentication scheme based on an elliptic curve [3]. However, this scheme is found to be vulverable to impersonation attacks; an attacker can impersonate as a legal server according to Farash [4]. Ibrahim et al. proposed secure anonymous mutual authentication for star two-tier wireless body area networks [5]. Chaudhry et al. proposed a remote user authentication scheme using elliptic curve cryptography that can withstand various attacks in the internet of things scenario, for example, smart card lost attack, replay attack [6]. Kumari analyzed the scheme of Farash [7], and they found that Farash's scheme is vulnerable to various attacks, for example, impersonation attack, password guessing attack and temporary session specific information reveal attack, etc.

Jing et al. proposed an authentication between user and server, which could protect well the identity privacy of the user [8], however, their scheme requires extra storage capacity at the server side. In the scheme of Xiong [9], only registered users can authenticate each other and build a shared key, besides, this shared key is only known by the two registered users and the network manager could not know this shared key. According to the public information transmitted between the two users, an adversary is unable to learn this shared key. The scheme of Jing et al. is a scheme equipped with elliptic curve cryptographic primitives. Their scheme achieves anonymity regardless of network infrastructure. Their scheme enables the server to provide various services for a client more than once with a negligible computational cost [10]. Idrissi proposed a security scheme for mobile agent based on two techniques: anonymous authentication and intrusion detection [11]. In the work of Xiong et al. [12], the anonymity is enabled, however the gateway has to store a lot of the identity and key pairs.

In some schemes, the gateway assigns a random number, and a unique key based on this number to the clients. This number is used as an indicator of the key, the user encrypts his identity with this key. Many other schemes use this way to protect the identity of the users, for example, the scheme in the works of [13–18]. Biometrics are used in the scheme of Wu et al. [19], Odelu et al. [20], Wang et al. [21] and Islam et al. [22]. Human beings' biometrics are extracted as random strings by using the fuzzy extractor.

The partial public key method is a popular method that has been used. He et al. proposed an efficient identity-based privacy-preserving authentication scheme for vehicular ad hoc networks [23], batch verification is used in this study. The concept of partial public key is also used in the scheme of Islam et al. [24]. In their scheme, a user register at the server several times, in order to get more than one authentication keys, then the user can use different keys for authentication to achieve anonymity. The scheme of Porambage et al. [25] also used the partial public key concept. Tsai et al. proposed a scheme for distributed mobile cloud computing services [26], the security strength of their scheme is based on bilinear pairing and dynamic nonce generation. There are other schemes that based on the elliptic curve security [27–29].

3. The Proposed Scheme

3.1. Structure of the Scheme

There are two types of entities in the scheme: remote clients and the server, which is shown in Figure 1.

1. A client is the one who wants to access the services provided by the server. A client first registers at the server, after the registration, he can conduct a mutual authentication with the server, after authentication, the two can build a shared key, the client can access to the server's service using this key.
2. A server is the one that provides different kinds of services to the client. A server is also responsible for the registration and password modification for the client. Before the server provides a service to a client, the server has to make sure if the client is a registered one or not.

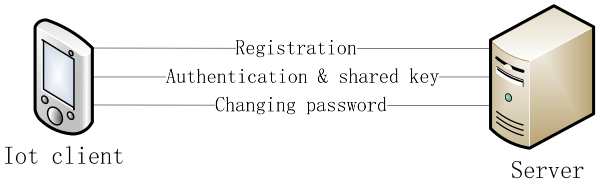

Figure 1. The structure of the proposed scheme.

The proposed scheme is a mutual authentication scheme between the client and the server. The scheme consists of three phases: registration phase of the client, mutual authentication and key establishment phase and the phase for a client to change his password.

3.2. System Initialization

In the beginning, the server S generates and publicizes the parameters of an elliptic curve, which is $\{p, a, b, P, n, h\}$. After that, S generates its private key X_{GWN}, and keeps it as a secret. The symbols that will be used in this study are summarized in Table 1.

Table 1. Symbols used in this study.

Symbols	Meaning
S	The server
C_i	The i_{th} client
ID_i	The i_{th} client's identity
$\vert\vert$	String connector, connecting two strings
$\oplus$	XOR operation
P	The generator of ECC
T_1	Timestamp
h	The SHA-256 hash function
h_1	A hash a string to a random number function

3.3. Registration Phase

All the clients have to register at the server, a client C_i with identity ID_i generates a registration request message, and sends this request to the server S.

1. Client C_i chooses a random number r_i.
2. Client C_i calculates a hash message $MP_i = h(r_i||ID_i||PW_i)$.
3. Client C_i sends $\{ID_i, MP_i\}$ to the server.

When the server S receives the message, server S generates the keys for client C_i, after that, the server S sends these keys to the client C_i. Table 2 is a description of the process.

1. Server S calculates a hash message $d_i = h(ID_i||X_{GWN})$.
2. Server S calculates $f_i = d_i \oplus MP_i$.
3. Server S chooses a random number k_i.
4. Server S calculates a hash message $e_i = h(k_i||X_{GWN})$.
5. Server S calculates $h_i = e_i \oplus MP_i$.
6. Server S sends $\{f_i, h_i, k_i\}$ and other system parameters to the client C_i.

Table 2. Registration phase.

Client	Server
ID_i, PW_i	master key X_{GWN}
random number r_i	
$MP_i = h(r_i\|\|ID_i\|\|PW_i)$	
$\{ID_i, MP_i\}$ $\longrightarrow$	$d_i = h(ID_i\|\|X_{GWN})$ $f_i = d_i \oplus MP_i$ random number k_i
	$e_i = h(k_i\|\|X_{GWN})$
	$h_i = e_i \oplus MP_i$
Stores $\{f_i, h_i, k_i\}$	$\{f_i, h_i, k_i\}$ $\longleftarrow$

3.4. Authentication Phase

If a client C_i with identity ID_i wants to ask a service from the server S, first, the two have to authenticate each other and build a shared key. The client C_i inserts the smart card into a card reader, inputs his identity ID_i' and password PW_i'. The smart card (SC) prepares the following message and sends it to the server S.

1. The client C_i inserts its smart card into a card reader, inputs his identity ID_i' and password PW_i'.
2. SC computes: $MP_i' = h(r_i||ID_i'||PW_i')$.
3. SC uses MP_i' to get $d_i = f_i \oplus MP_i'$ and $e_i = h_i \oplus MP_i'$.
4. SC gets the current timestamp T_1 and the random number k_i.
5. SC gets a random number $k_1 \in [1, n-1]$, and calculates $A_1 = k_1 \cdot P$.
6. SC gets the hash $M_1 = h(A_1||ID_i'||k_i||d_i||T_1)$.
7. SC computes $M_2 = (ID_i'||M_1) \oplus e_i$.
8. Finally, SC sends $\{k_i, A_1, M_2, T_1\}$ to the server S.

When the server S receives the incoming message, it first checks the correctness of the message, after the verification, the server will generate the shared key between himself and the client. Then the server prepares the message for sending back to the client.

1. Server S checks the freshness of the T_1, if T_1 is not fresh, server S abandons the incoming message, the scheme ends here.
2. Server S calculates the key $h(k_i||X_{GWN})$ based on k_i.
3. Server S uses the key $h(k_i||X_{GWN})$ to decrypt M_2 to get $ID_i'||M_1', ID_i'||M_1' = h(k_i||X_{GWN}) \oplus M_2$.
4. Server S calculates $d_i' = h(ID_i'||\ X_{GWN})$ based on the identity ID_i'.
5. Server S checks if $M_1' = h(A_1||ID_i'||k_i||d_i'||T_1)$, if they are equal, the server accepts the incoming message, otherwise, the scheme terminates here.
6. Server S gets a random number $k_2 \in [1, n-1]$, and calculates $B_2 = k_2 \cdot P$.
7. Server S calculates the shared key $SK = h(k_2 \cdot A_1||T_1)$.
8. Server S calculates a new random number $k_{inew} = h_1(SK||T_1)$.
9. Server S calculates a hash message $e_{inew} = h(k_{inew}||X_{GWN})$.
10. Server S calculates $M_3 = h(B_2||e_{inew}||k_{inew}||d_i'||SK)$.
11. Server S computes $M_4 = (e_{inew}||M_3) \oplus h(d_i'||T_1)$.
12. Server S sends $\{B_2, M_4\}$ to the client C_i.

When client C_i gets the message$\{B_2, M_4\}$, C_i will do the following steps to authenticate the incoming message, if the client verifies the message, he will build a shared key with the server.

1. Client C_i computes the shared key as $SK' = h(k_1 \cdot B_2||T_1)$.
2. Client C_i decrypts M_4 to get $e_{inew}'||M_3' = M_4 \oplus h(d_i||T_1)$.
3. Client C_i computes the random number $k_{inew}' = h_1(SK'||T_1)$.
4. Client C_i checks if $M_3' = h(B_2|| e_{inew}'||k_{inew}'||d_i||SK')$, if they are equal, C_i accepts the shared key SK', and now client C_i and the server S can communicate using the shared key $SK = SK'$, otherwise the scheme terminates here.
5. Client C_i updates $h_i = e_{inew}' \oplus MP_i'$ and $k_i = k_{inew}'$.

Now the client C_i and the server S have authenticated each other and built a shared key. The Table 3 below depicts the whole process.

Table 3. Authentication phase.

Client	**Server**
$\boldsymbol{ID_i, PW_i}$	**Master Key** $\boldsymbol{X_{GWN}}$
User: inserts SC into the terminal	
User: input ID_i' and PW_i'	
SC: $MP_i' = h(r_i||ID_i'||PW_i')$	
SC: $d_i = f_i \oplus MP_i'$	
SC: $e_i = h_i \oplus MP_i'$	
SC: gets timestamp T_1, k_i	
Random number k_1, $A_1 = k_1 \cdot P$	
SC: gets $M_1 = h(A_1||ID_i'||k_i||d_i||T_1)$	
SC: $M_2 = (ID_i'||M_1) \oplus e_i$	
$\{k_i, A_1, M_2, T_1\}$ $\longrightarrow$	Checks the freshness of T_1 $ID_i'||M_1' = h(k_i||X_{GWN}) \oplus M_2$
	$d_i' = h(ID_i'|| X_{GWN})$
	Check if $M_1' = h(A_1||ID_i'||k_i||d_i'||T_1)$
	Random number k_2, $B_2 = k_2 \cdot P$
	$SK = h(k_2 \cdot A_1||T_1)$
	$k_{inew} = h_1(SK||T_1)$
	$e_{inew} = h(k_{inew}||X_{GWN})$
	$M_3 = h(B_2||e_{inew}||k_{inew}||d_i'||SK)$
	$M_4 = (e_{inew}||M_3) \oplus h(d_i'||T_1)$
$SK' = h(k_1 \cdot B_2||T_1)$ $e_{inew}'||M_3' = M_4 \oplus h(d_i||T_1)$	$\{ B_2, M_4\}$ $\longleftarrow$
$k_{inew}' = h_1(SK'||T_1)$	
Check if $M_3' = h(B_2||e_{inew}'||k_{inew}'||d_i||SK')$	
$h_i = e_{inew}' \oplus MP_i'$, $k_i = k_{inew}'$	
Agree on the key $SK = SK'$	

3.5. Password Change Phase

When a client C_i wants to change his password, he can send a request to the server S, this request is sent in public channel. Table 4 is a description of this process.

1. The client C_i inserts his smart card into a card reader, inputs his identity and password ID_i' and PW_i'.
2. SC computes: $MP_i' = h(r_i||ID_i'||PW_i')$.
3. SC uses MP_i' to get $d_i = f_i \oplus MP_i'$ and $e_i = h_i \oplus MP_i'$.
4. SC gets the current timestamp T_1 and the random number k_i.
5. SC gets the hash $M_1 = h(ID_i'||k_i||d_i||T_1)$.
6. SC computes $M_2 = (ID_i'||M_1) \oplus e_i$.
7. Finally, SC sends $\{k_i, M_2, T_1\}$ to the server S.

When the server S receives the message, server S will verify if the message is from a legitimate client, after that, the server S sends a replay to the client C_i.

1. Server S checks the freshness of the T_1, if T_1 is not fresh, server S abandons the incoming message.
2. Server S calculates the key $h(k_i||X_{GWN})$ based on k_i.
3. Server S uses the key $h(k_i||X_{GWN})$ to decrypt M_2 to get $ID_i'||M_1'$, $ID_i'||M_1' = h(k_i||X_{GWN})\oplus M_2$.
4. Server S calculates $d_i' = h(ID_i'|| X_{GWN})$ based on the identity ID_i'.
5. Server S checks if $M_1' = h(ID_i'||k_i||d_i'||T_1)$, if they are equal, the server verifies the incoming message, otherwise, the scheme terminates here.
6. Server S calculates $M_3 = h(ID_i'||d_i'||k_i||T_1)$.
7. Server S sends $\{M_3\}$ to the client C_i.

When a client C_i receives the replay message from the server S, the smart card checks the correctness of this message, if it is from the server S, then the smart card will allow the client C_i to input his new password.

1. SC checks if $M_3 = h(ID_i'||d_i||k_i||T_1)$, if they are equal, then the client is allowed to change his password.
2. SC computes $d_i = f_i \oplus MP_i'$ using the stored f_i and the old MP_i'.
3. SC computes $e_i = h_i \oplus MP_i'$ using the stored h_i and the old MP_i'
4. Client C_i inputs the new password PW_i^*.
5. SC updates MP_i' to be $MP_i^* = h(r_i||ID_i||PW_i^*)$.
6. SC uses this new MP_i^* to update the stored version of f_i and h_i to get $f_i' = d_i \oplus MP_i^*$, $h_i' = e_i \oplus MP_i^*$.

Table 4. Password change phase.

Client	Server
$\boldsymbol{ID_i, PW_i}$	**Master Key $\boldsymbol{X_{GWN}}$**
User: inserts SC into the terminal	
User: input ID_i' and PW_i'	
SC: $MP_i' = h(r_i||ID_i'||PW_i')$	
SC: $d_i = f_i \oplus MP_i'$	
SC: $e_i = h_i \oplus MP_i'$	
SC: gets timestamp T_1, k_i	
SC: gets $M_1 = h(ID_i'||k_i||d_i||T_1)$	
SC: $M_2 = (ID_i'||M_1)\oplus e_i$	
$\{k_i, M_2, T_1\}$ $\longrightarrow$	Check the freshness of T_1 $ID_i'||M_1' = h(k_i||X_{GWN})\oplus M_2$
	$d_i' = h(ID_i'|| X_{GWN})$
	Check if $M_1' = h(ID_i'||k_i||d_i'||T_1)$
	$M_3 = h(ID_i'||d_i'||k_i||T_1)$.
Check if $M_3 = h(ID_i'||d_i||k_i||T_1)$ $d_i = f_i \oplus MP_i'$	$\{ M_3\}$ $\longleftarrow$
$e_i = h_i \oplus MP_i'$	
$MP_i^* = h(r_i||ID_i|| PW_i^*)$	
$f_i' = d_i \oplus MP_i^*$	
$h_i' = e_i \oplus MP_i^*$	

4. Security Analysis by AVISPA

Automated Validation of Internet Security Protocols and Applications (AVISPA) is "a push-button tool for the automated validation of Internet security-sensitive protocols and applications" [30]. To test security features of the scheme in this study, we write the scheme in a role-based language called High-Level Protocols Specification Language (HLPSL), which is used for describing protocols and specifying their intended security features. The HLPSL code is listed in Appendix A.

We run the security check by using the CL-based Model-Checker [31], and the checker of On-the-Fly Model-Checker (OFMC) [32,33]. The simulation result shown in Table 5 demonstrates that the proposed scheme is safe.

Table 5. Simulation results of AVISPA.

CL-AtSe back-end	OFMC
SUMMARY	% OFMC
SAFE	% Version of 2006/02/13
DETAILS	SUMMARY
BOUNDED_NUMBER_OF_SESSIONS	SAFE
TYPED_MODEL	DETAILS
PROTOCOL	BOUNDED_NUMBER_OF_SESSIONS
/home/iotdev/avispa/avispa-1.1/testsuite/results/light.if	PROTOCOL
	/home/iotdev/avispa/avispa-1.1/testsuite/results/light.if
GOAL	GOAL
As Specified	as_specified
	BACKEND
BACKEND	OFMC
CL-AtSe	COMMENTS
	STATISTICS
STATISTICS	parseTime: 0.00s
	searchTime: 0.01s
Analysed: 1 states	visitedNodes: 4 nodes
Reachable: 0 states	depth: 2 plies
Translation: 0.00 s	
Computation: 0.00 s	

5. Security Analysis Using BAN Logic

We conducted a security analysis of the proposed scheme using Burrows-Abadi-Needham Logic (BAN logic) [34]. By using BAN logic, we can determine whether the exchanged information is trustworthy, secure against eavesdropping. For more information on the symbols and primary postulates of BAN logic, please refer to our previous work [35].

5.1. The Premise and Proof Goals

Suppose there are two entities in the system: client C_i and the server S. Before we start the proof, we first translate the messages into an idealized form of BAN logic, the results are shown in Table 6.

Table 6. The idealized form of the messages.

Message	Flow	Idealized Form
1	$C_i \to S$	$\{k_i, A_1, \{A_1, ID_i, k_i, T_1\}_{d_i}, T_1\}$
2	$S \to C_i$	$\{B_2, \{e_{inew}, B_2, k_{inew}, d_i, SK\}_{h(d_i \vert\vert T_1)}\}$

The goals in BAN Logic are described below. These goals can ensure C_i and S to agree on a shared key SK.

$$1.\ C_i \mid\equiv\ C_i \overset{SK}{\leftrightarrow} S\ 2.\ S \mid\equiv\ S \overset{SK}{\leftrightarrow} C_i$$

5.2. Assumptions

We make some assumptions to help us to prove the protocol; assumptions are listed in Table 7. First, we show the proof of assumption A1 and A3.

1. According to the "#()-introduction" rule, client C_i creates T_1

$$C_i \mid \equiv \#(T_1) \tag{1}$$

2. According to (1) and the "promotion #" rule:

$$C_i \mid \equiv \#(M_4) \tag{2}$$

3. According to (2) and the "promotion #" rule:

$$C_i \mid \equiv \#(B_2, M_4) \tag{3}$$

4. According to (3) and the "elimination of multipart messages" rule:

$$C_i \mid \equiv \#(B_2) \tag{4}$$

In this part, we show the proof of assumption A2 and A4. By checking the timestamp T_1, the server S can judge if T_1 is fresh or not, if T_1 is not fresh, the server S will abandon the message and the scheme ends here. Thus, we only consider the situation that server S believes timestamp T_1 is fresh, which is $S \mid \equiv \#(T_1)$.

5. According to the "promotion #" rule:

$$S \mid \equiv \#(k_i, A_1, M_2, T_1). \tag{5}$$

6. According to (5) and the "elimination of multipart messages" rule:

$$S \mid \equiv \#(A_1) \tag{6}$$

After registration, both server S and the client C_i believe that they have a shared key d_i. Translating into BAN Logic, we get assumptions A6: $S \mid \equiv C_i \overset{d_i}{\leftrightarrow} S$ and $C_i \mid \equiv S \overset{d_i}{\leftrightarrow} C_i$. We can get assumptions A5: $C_i \mid \equiv S \xleftrightarrow{h(d_i||T_1)} C_i$ based on $C_i \mid \equiv S \overset{d_i}{\leftrightarrow} C_i$. Assumption A7 says that client C_i believes server S has complete control over the data B_2, assumption A8 says that server S believes client C_i has complete control over the data A_1.

Table 7. Assumptions.

Number	Assumptions	Number	Assumptions
A1	$C_i \mid \equiv \#(T_1)$	A2	$S \mid \equiv \#(T_1)$
A3	$C_i \mid \equiv \#(B_2)$	A4	$S \mid \equiv \#(A_1)$
A5	$C_i \equiv S \xleftrightarrow{h(d_i\|\|T_1)} C_i$	A6	$S \equiv C_i \xleftrightarrow{d_i} S$
A7	$C_i \mid \equiv S \Longmapsto B_2$	A8	$S \mid \equiv C_i \Longmapsto A_1$

5.3. The Proof of the Proposed Scheme

In this section, we start the proof. According to the message $\{k_i, A_1, \{A_1, ID_i, k_i, T_1\}_{d_i}, T_1\}$, which the client C_i sends to server S, we can get the followings:

7. According to the message $\{k_i, A_1, \{A_1, ID_i, k_i, T_1\}_{d_i}, T_1\}$:

$$S \triangleleft \{k_i, A_1, \{A_1, ID_i, k_i, T_1\}_{d_i}, T_1\} \tag{7}$$

8. According to (7) and " ','-elimination" rule:

$$S \triangleleft \{A_1, ID_i, k_i, T_1\}_{d_i} \tag{8}$$

9. According to (8), A6 and "$\mid\sim$ introduction" rule:

$$S \mid\equiv C_i \mid\sim \{A_1, ID_i, k_i, T_1\} \tag{9}$$

10. According to (9) and " ','-elimination" rule:

$$S \mid\equiv C_i \mid\sim A_1 \tag{10}$$

11. According to A4, (10), and "$\mid\sim$elimination" rule:

$$S \mid\equiv C_i \mid \equiv A_1 \tag{11}$$

12. According to A8, (11), and "jurisdiction or control" rule:

$$S \mid \equiv A_1 \tag{12}$$

13. As k_2 is randomly created by S, according to "#()- introduction" rule:

$$S \mid \equiv \#(k_2) \tag{13}$$

14. According to (13), A2, A4, and "#()- promotion" rule:

$$S \mid \equiv \#(SK) \tag{14}$$

15. According to (11), (14), and "$\overset{k}{\leftrightarrow}$ introduction" rule:

$$S \mid \equiv S \overset{SK}{\longleftrightarrow} C_i \tag{15}$$

Now we have proved the second goal, we will begin to prove the first goal by analyzing the message server S sends to client C_i: $\{B_2, \{e_{inew}, B_2, k_{inew}, d_i, SK\}_{h(d_i||T_1)}\}$.

16. According to the message $\{B_2, \{e_{inew}, B_2, k_{inew}, d_i, SK\}_{h(d_i||T_1)}\}$:

$$C_i \triangleleft \{B_2, \{e_{inew}, B_2, k_{inew}, d_i, SK\}_{h(d_i||T_1)}\} \tag{16}$$

17. According to (16) and " ','-elimination" rule:

$$C_i \triangleleft \{e_{inew}, B_2, k_{inew}, d_i, SK\}_{h(d_i||T_1)} \tag{17}$$

18. According to (17), A5 and "$\mid\sim$ introduction" rule:

$$C_i \mid\equiv S \mid\sim \{e_{inew}, B_2, k_{inew}, d_i, SK\} \tag{18}$$

19. According to (18) and " ','-elimination" rule:

$$C_i \mid\equiv S \mid\sim B_2 \tag{19}$$

20. According to A3, (19), and "$\mid\sim$elimination" rule:

$$C_i \mid\equiv S \mid \equiv B_2 \tag{20}$$

21. According to A7, (20), and "jurisdiction or control" rule:

$$C_i \mid \equiv B_2 \tag{21}$$

22. As k_1 is randomly created by C_i, according to "#()- introduction" rule:

$$C_i \mid \equiv \#(k_1) \tag{22}$$

23. According to (22), A1, A3, and "#()- promotion" rule:

$$C_i \mid \equiv \#(SK) \tag{23}$$

24. According to (20), (23), and "$\overset{k}{\leftrightarrow}$ introduction" rule:

$$C_i \mid \equiv C_i \overset{SK}{\longleftrightarrow} S \tag{24}$$

Now, we have proved the two goals of the scheme. We can say that the proposed scheme is secure under BAN logic.

6. Formal Security Analysis

Suppose G_1 is a cyclic additive group of prime order q, P is the generator of G_1, the Elliptic Curve Computational Diffie–Hellman (*ECCDH*) problem is thought to be a computational hardness. The security of the shared key of the proposed scheme is based on the computational hardness of the *ECCDH* problem.

Definition 1. *ECCDH problem. For any* $a, b, c \in Z_q^*$, *given an instance* $< aP, bP >$, *it is computationally intractable to compute* $cP = abP$.

Theorem 1. *The proposed scheme achieves shared key security if and only if the ECCDH problem is unable to be solved in polynomial time.*

We define the shared key security as that an adversary is unable to get the shared key between the client C_i and server S based on the messages transferred publicly between them.

Proof.

(⇒) Suppose there is an efficient algorithm $\mathcal{O}_I$ which could break the *ECCDH* problem in probabilistic polynomial time. The adversary is able to get the messages publicly sent between the client C_i and the server S: $\{k_i, A_1, M_2, T_1\}$, and $\{B_2, M_4\}$. Suppose $a \cdot P = A_1 = k_1 \cdot P$ and $\cdot P = B_2 = k_2 \cdot P$, adversary $\mathcal{A}_{\mathcal{J}}$ is able to get the $cP = k_1 \cdot k_2 \cdot P$ by using efficient algorithm $\mathcal{O}_I$, the adversary is able to break the security of the shared key and get the shared key $h(k_1 \cdot k_2 \cdot P \,||T_1)$.

(⇐) Suppose there is an efficient algorithm $\mathcal{O}_{II}$ which could get the shared key between client C_i and server S, as the hash operation is secure, the adversary has to get the shared key by calculating $k_1 \cdot k_2 \cdot P$. This means given $A_1 = k_1 \cdot P$ and $B_2 = k_2 \cdot P$, an adversary $\mathcal{A}_{II}$ is able to get $k_1 \cdot k_2 \cdot P$. For the *ECCDH* problem, suppose $a \cdot P = A_1 = k_1 \cdot P$ and $b \cdot P = B_2 = k_2 \cdot P$, the adversary is able to get $c \cdot P = a \cdot b \cdot P = k_1 \cdot k_2 \cdot P$. This apparently contradicts the hardness of the *ECCDH* problem. □

Theorem 2. *The proposed scheme achieves perfect forward privacy if and only if the ECCDH problem is unable to solve in polynomial time.*

Proof.

The proof of perfect forward privacy is similar to Theorem 1. Even if the private key of the client is leaked to the adversary. What the adversary get is the same public information $\{k_i, A_1, M_2, T_1\}$ and $\{B_2, M_4\}$. Thus it is unable to get the past session key, neither. □

7. Comparison

In this section, we compared our scheme with related works in computation cost, computation at the registration phase and the authentication phase. The schemes are implemented in C++, the running codes have been upload to a public repository in the github.com [36]. The MIRACL C/C++ Library is used in this study [37], the library can be accessed at github.com [38]. The experiment is conducted in Visual Studio C++ 2017 on a 64-bits Windows 7 operating system, 3.5 GHz processor, 8 GB memory. The hash function is SHA-256, the symmetric encryption/decryption function is AES in MR_PCFB1 form, the 256-bit long key for symmetric encryption/decryption function is generated by SHA-256 hash operation. The Koblitz curve secp256k1 which is recommended by NIST is used in this study [39]. The parameters of this curve are listed in Appendix B. The code is compiled in x86 form, this simulation does not take into account the transmission of the data.

7.1. Computational Performance Analysis

First, we compared the computation costs of these schemes in the form of operation per phase, T_{H}, T_{MUL}, T_{ADD}, $T_{E/D}$ are used for the computation cost for SHA-256 operation, element multiplication operation of G_1, element addition operation of G_1, and AES symmetric encryption/decryption operation. The results are listed at Table 8. As shown in the table, we can find that in all conditions, the computation cost of the proposed scheme is the minimal, as T_{MUL} > T_{H} and $T_{E/D}$ > T_{H}. Thus, the proposed scheme has an advantage in the computation cost and energy consumption compared to related works. To test the analysis of the computation cost, we also simulated the schemes in the aforementioned environment respectively.

Table 8. Computation costs in the form of operation per phase.

Reference	Registration Phase	Authentication Phase	Password Change Phase
Tu et al. [**Error! Bookmark not defined.**]	$2T_H + 1T_{MUL}$	$10T_H + 7T_{MUL} + 1T_{ADD}$	$6T_H + 1T_{MUL} + 4T_{E/D}$
Chaudhry et al. [**Error! Bookmark not defined.**]	$5T_H + 1T_{MUL}$	$14T_H + 6T_{MUL} + 1T_{ADD}$	---
Wu et al. [**Error! Bookmark not defined.**]	$4T_H$	$12T_H + 4T_{MUL} + 4T_{E/D}$	$9T_H + 1T_{MUL} + 2T_{E/D}$
Our scheme	$3T_H$	$14T_H + 4T_{MUL}$	$9T_H$

First, we run the registration phase of different schemes 5, 10, 15, 20 and 25 times separately. The computation times are shown in Figure 2. The horizontal axis represents the number of runs of the experiment, the vertical axis represents the time required for the experiment to run, and the unit is milliseconds. The computation cost of Wu et al. [**Error! Bookmark not defined.**] and that of the proposed scheme are relatively smaller, while the scheme of Chaudhry et al. [**Error! Bookmark not defined.**], and that of Tu et al. [**Error! Bookmark not defined.**] cost more computation time. This is mainly because the proposed scheme and the scheme of Wu et al. [**Error! Bookmark not defined.**] only need lightweight operations, SHA-256 hash operations and XOR operation, while for the scheme of Chaudhry et al. [**Error! Bookmark not defined.**], and that of Tu et al. [**Error! Bookmark not defined.**], symmetric encryption/decryption operations are required, these operations cost more computation time.

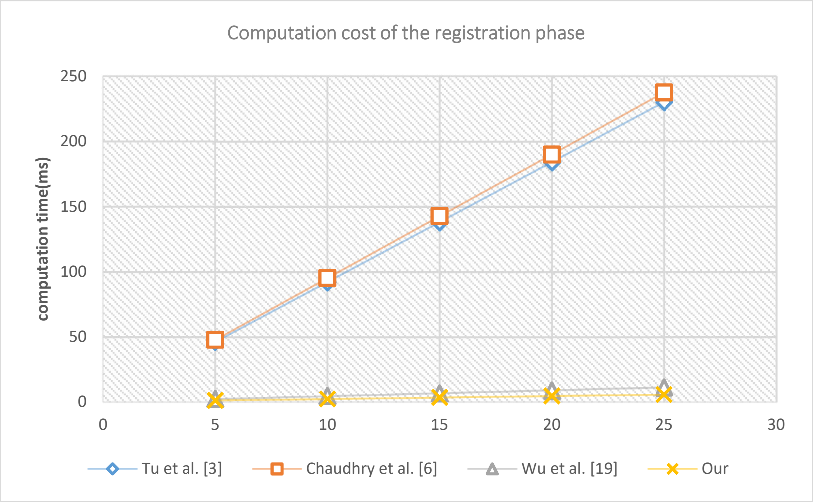

Figure 2. The computation cost of registration phase.

Second, we run the authentication and key establishment phase of different schemes 5, 10, 15, 20 and 25 times separately. The computation costs are shown in Figure 3. The horizontal axis represents the number of running the experiment, the vertical axis stands for the number of milliseconds to accomplish the experiment. The computation cost of Wu et al. [**Error! Bookmark not defined.**] and that of the proposed scheme are relatively smaller, while the scheme of Chaudhry et al. [**Error! Bookmark not defined.**], and the scheme of Tu et al. [**Error! Bookmark not defined.**] cost more computation time. The computation cost of the proposed scheme is the minimal.

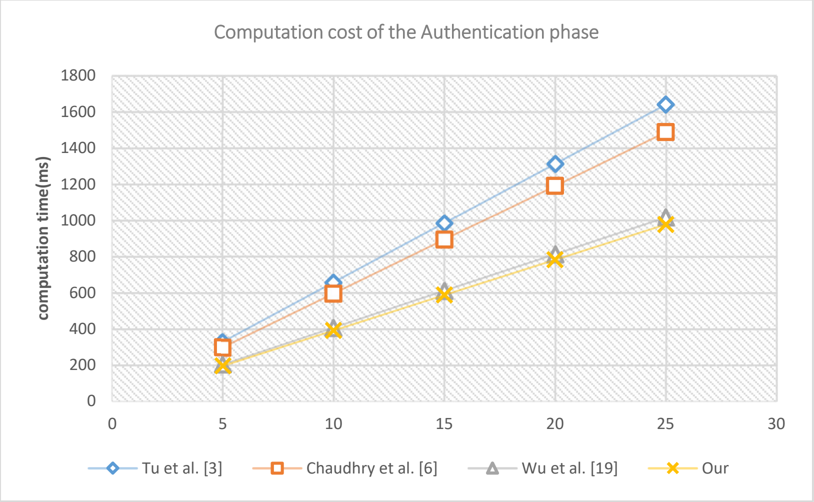

Figure 3. The computation cost of authentication phase.

Third, we run the password change phase 5, 10, 15, 20 and 25 times separately. The computation costs are shown in Figure 4. In this figure, the horizontal axis indicates the number of times the experiment was run; the vertical axis indicates the number of milliseconds to accomplish the experiment. The computation cost of the proposed is the minimal, the computation cost of Wu et al. [**Error! Bookmark not defined.**], and that of Tu et al. [**Error! Bookmark not defined.**] are much higher, this is because in the proposed scheme only SHA-256 hash operations and XOR operation are needed, while in the scheme of Wu et al. [**Error! Bookmark not defined.**], and in the scheme of Tu et al. [**Error! Bookmark not defined.**], symmetric encryption/decryption, and elliptic curve operation are needed, these operations cost more computation time.

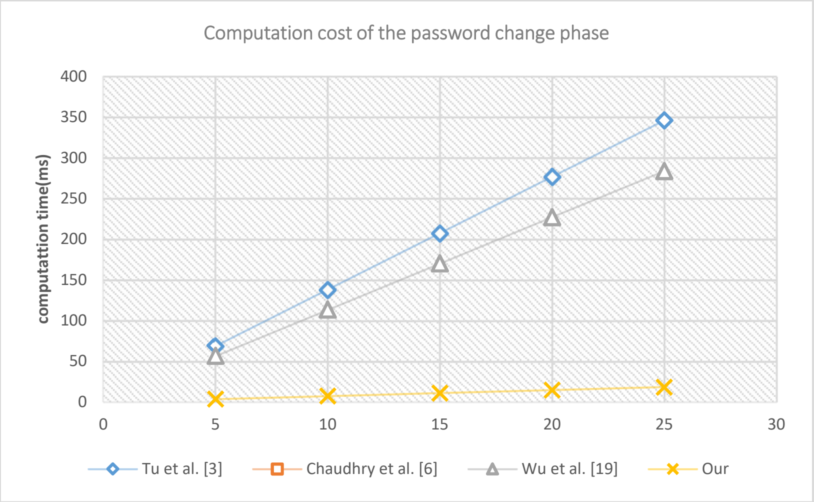

Figure 4. The computation cost of password change phase.

7.2. Communication Performance Analysis

In this part, we compared all the schemes in communication cost. We use the same criteria as that in the study of Jing et al. [**Error! Bookmark not defined.**], the identity costs 2 bytes. The general hash operation in this study is SHA-256, the result of a hash operation is set to be 32 bytes. In this study, the random number is set to be 4 bytes, the timestamp is set to be 4 bytes. The element of the G_1 of the Koblitz curve secp256k1 is 64 bytes. The order $|q|$ of G_1 is 32 bytes long.

At the registration phase, the client sends $\{ID_i, MP_i\}$ to the server, MP_i is a result of hash, it is 32 bytes long. The length of this message is 2 + 32 = 34 byte. The server sends $\{f_i, h_i, k_i\}$, f_i is 32 byte long, h_i is also 32 byte long. k_i is 4 bytes a random number. The length of this message is 32 + 32 + 4 = 68 byte long. In the registration phase, the communication cost is 34 + 68 = 102 byte.

At the authentication phase, the client has to send $\{k_i, A_1, M_2, T_1\}$ to the server, k_i is a random number of be 4 bytes, A_1 is an element of G_1, it is 64 bytes long, $M_2 = (ID_i' || M_1) \oplus e_i$, Id_i' is an identity, it is 2 bytes long, M_1 is the result of an hash operation, it is 32 bytes long, the length of M_2 is 32 + 2 = 34 byte. T_1 is a 4 bytes long timestamp. The length of this message is 4 + 64 + 34 + 4 = 106. The server has to send $\{B_2, M_4\}$ back to the client, B_2 is an element of G_1, it is 64 bytes long. $M_4 = (e_{inew} || M_3) \oplus h(d_i' || T_1)$, e_{inew} and M_3 are the results of hash, they are both 32 bytes long, the length of M_4 is 32 + 32 = 64 byte. The length of this message is 64 + 64 = 128 byte long. The communication cost of is 106 + 128 = 234 byte.

At the password change phase, the client has to send $\{k_i, M_2, T_1\}$ to the server, k_i is a random number of be 4 bytes, $M_2 = (ID_i' || M_1) \oplus e_i$, Id_i' is an identity, it is 2 bytes long, M_1 is the result of an hash operation, it is 32 bytes long, the length of M_2 is 32 + 2 = 34 byte. T_1 is a 4 bytes long timestamp. The length of this message is 4 + 34 + 4 = 42. The server has to send $\{M_3\}$ back to the client, M_3 is the result of hash, it is 32 bytes long, the length of this message is 32 byte long. The communication cost of this phase is 42 + 32 = 74 byte.

The communication costs of other schemes are computed in the same way, note that, in the scheme of Tu et al. [**Error! Bookmark not defined.**], to change a client's password, the client and the server has to build a shared key in advance, thus, the communication cost of the password change phase is calculated as the communication cost of the authentication phase and the messages sent during the password change process. The scheme of Chaudhry et al. [**Error! Bookmark not defined.**] does not have a password change phase; we did not calculate their scheme's communication cost. The result is shown in Table 9.

Table 9. Communication costs of different schemes.

Reference	Registration Phase	Authentication Phase	Password Change Phase
Tu et al. [**Error! Bookmark not defined.**]	98 byte	230 byte	456 byte
Chaudhry et al. [**Error! Bookmark not defined.**]	130 byte	226 byte	---
Wu et al. [**Error! Bookmark not defined.**]	102 byte	238 byte	138 byte
Our scheme	102 byte	234 byte	74 byte

8. Security Feature Analyses

In this section, we analyzed the security features of different schemes. At the end of this section, we concluded the results into a table.

8.1. Client Anonymity

Regarding client anonymity, in the proposed scheme, the identity of the user is encrypted by a shared key between the client and the server, the adversary is unable to find out the real identity of the client. In the scheme of Tu et al. [**Error! Bookmark not defined.**], the identity of the user is transmitted transparently; the adversaries can get the identity easily. In the scheme of Chaudhry et al. [**Error! Bookmark not defined.**] and Wu et al. [**Error! Bookmark not defined.**], the identity is encrypted, too.

8.2. Perfect Forward Privacy

Perfect forward privacy means that even when an adversary gets the private key of the client or the server, it is unable to recover the past session key based on this private key and the publicly transmitted messages. As we have proved in Section 5, the proposed scheme gains perfect forward privacy.

Meanwhile, the scheme of Chaudhry et al. [**Error! Bookmark not defined.**] cannot ensure perfect forward privacy, if the adversary gets the private key msk and the session related messages $DID_{ua}, EID_{ua}, Q_{ua}$ and T_{sb}, H_{sb} . The adversary is able to compute the past session key in the following manner:

$$M'_{ua} = msk \cdot Q_{ua}$$

$$EID_{ua} = M'_{ua} \oplus DID_{ua}$$

$$TID'_{ua} = H_1(msk \oplus ID_{ua}) \cdot P$$

$$Q'_{sb} = T_{sb} \oplus M'_{ua}$$

$$SK = H_5(Q_{ua} \oplus TID'_{ua} \oplus M'_{ua} \oplus TID'_{ua})$$

8.3. Reply Attack

In the proposed scheme, there is a timestamp T_1 in the message $\{k_i, A_1, M_2, T_1\}$, and the timestamp T_1 is also concealed in the hash message $M_1 = h(A_1||ID'_i||k_i||d_i||T_1)$. If an adversary sends a former message to the server, the server will abandon this message after checking the timestamp. However, if the adversary replaces the timestamp T_1 with a new one, the server can still find it out by checking the hash message $M_1 = h(A_1||ID'_i||k_i||d_i||T_1)$. Thus, an adversary is unable to launch a replay attack. For the scheme of Chaudhry et al. [**Error! Bookmark not defined.**], if an adversary sends a former message to the server, the server is unable to judge if the message is a previous one or not, therefore, their scheme is subjected to replay attack.

8.4. Offline Dictionary Attack

In the proposed scheme, if the adversary gets the message in the smartcard $\{f_i, h_i, k_i, r_i\}$. The adversary could conduct an offline dictionary attack in the following steps:

1. The adversary insert the smart card into a card reader, inputs a random identity and password pair ID'_i and PW'_i.
2. SC computes: $MP'_i = h(r_i||ID'_i||PW'_i)$.
3. SC uses MP'_i to get $d_i = f_i \oplus MP'_i$ and $e_i = h_i \oplus MP'_i$.
4. SC gets the current timestamp T_1, and gets k_i.
5. SC gets a random number $k_1 \in [1, n-1]$, and calculates $A_1 = k_1 \cdot P$.
6. SC gets the hash $M_1 = h(A_1||ID'_i||k_i||d_i||T_1)$.
7. SC computes $M_2 = (ID'_i||M_1) \oplus e_i$.
8. Finally, SC sends $\{k_i, A_1, M_2, T_1\}$ to the server S.
9. If the server sends back a replay message, the identity and password pair is correct, otherwise, go to step 1.

Now, q_{send} is used as the number of times an adversary can send a message to the server S in a time period, the server will set a limit on q_{send}, if the q_{send} exceeds this preset limit, The server will no longer process the incoming messages from this adversary, the adversary cannot continuing the dictionary attack in this time period. The $|D_{id}|$,$|D_{pass}|$ are used as the dictionary size of the identity and the password. Thus the probability p_{adv} that adversary correctly guesses the identity and password pair correctly is:

$$p_{adv} = \frac{q_{send}}{|D_{id}| * |D_{pass}|}$$

Set $|D_{id}|$,$|D_{pass}|$ to be large enough, the p_{adv} will be a small value, the aforementioned analysis is based on the authentication phase, the attack on the password changing phase is the same.

Meanwhile, in the scheme of Chaudhry et al. [**Error! Bookmark not defined.**], the adversary could conduct an offline dictionary attack in the following steps:

1. The adversary inserts the smart card into a card reader, inputs a random identity and password pair ID_i' and PW_i'.
2. The adversary waits for the computation of the smart card.
3. If the smart card sends out a message, the identity and password pair is correct, otherwise, goes to step 1.

As there is not a limit, the adversary can try as many times as he wants, thus the adversary will finally get the correct identity and password pair. This also means our scheme can withstand the smart card lost attack, when the smart card is lost, the adversary cannot launch an offline dictionary attack to get the private key of the client.

8.5. Impersonation Attack

In the scheme of Tu et al. [**Error! Bookmark not defined.**], an adversary can impersonate the server. Given the message a user sends to the server, $\{username, V, W\}$, an adversary can forge the following message, the user is unable to find out if this message is coming from an adversary or the server:

$$\text{Generate random numnber } c, r \in Z_n$$
$$C = c \cdot P, K = c \cdot V$$
$$SK = h_1(K||r||username)$$
$$Auth_s = h_2(K||W||r||SK)$$

However, in the proposed scheme, if an adversary wants to impersonate the server, it has to get $d_i' = h(ID_i'|| X_{GWN})$, the probablity that an adversary correctly guesses d_i' is$p_{d_i} = 1/(|D_{id}| * |D_{X_{GWN}}|)$, where $|D_{X_{GWN}}|$ means the dictionary size of the server's private key.

8.6. Secret Information Leakage Problem

In the scheme of Tu et al. [**Error! Bookmark not defined.**], if an adversary accidentally get the session ephemeral information b. The adversary is able to get the secret information $h(username||s) \cdot P$ in the following manner:

$$h(username||s) \cdot P = b^{-1} \cdot V'$$

With this secret information, the adversary can impersonate a legitimate client. However, in the proposed scheme, even the session ephemeral information is leaked, the adversary is unable to get the client's secret information.

Finally, we get Table 10, we find that the proposed scheme has more security features than the schemes in the related works.

Table 10. Security features comparison.

Security Feature	**Tu et al. [**Error! Bookmark not defined.**]**	**Chaudhry et al. [**Error! Bookmark not defined.**]**	**Wu et al. [**Error! Bookmark not defined.**]**	**Our Scheme**
Client anonymity	×	√	√	√
Client being tracked	×	√	√	√
Reply attack	×	×	×	√
Impersonation attack	×	√	√	√
Offline dictionary attack	√	×	√	√
Smart card lost attack	√	×	√	√
Changing password	√	×	√	√

Secret information leakage problem	×	√	√	√
Perfect forward privacy	√	×	√	√

9. Conclusions

In this study, an authentication and key establishment scheme between remote clients and a server is proposed. The proposed scheme has been verified by AVISPA and BAN Logic, the verification results show that the proposed scheme can withstand various attacks. The proposed scheme has been simulated in C++, by comparison, it shows clearly that the proposed scheme is more efficient compared to the related works regarding the computation cost and the communication cost. Besides, the proposed has more security features compared to the related works. Our work is part of the LifeWear project, in which we focus on the safety of data transmission and identity privacy problem.

Author Contributions: Conceptualization, J.-F.M.; Methodology, Y.C; Validation, Y.C.; Formal Analysis, Y.C.; Investigation, Y.C., P.C. and L.L.; Resources, J.-F.M.; Data Curation, Y.C.; Writing—Original Draft Preparation, Y.C.; Writing—Review & Editing P.C. and L.L.; Visualization, Y.C.; Supervision, J.-F.M. and L.L.; Project Administration, J.-F.M.; Funding Acquisition, J.-F.M.

Funding: This research was funded by Spanish Ministry of Industry, Energy and Tourism, grant number TSI-010400-2010-100 and the Chinese Scholarship Council (CSC) grant number 201507040027.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

The role of the client.

```
role sender(Ui,Sj: agent,
                      Di,Ei: symmetric_key,
                      H     : hash_func,
                      P     : text,
                      SND_US,RCV_US : channel (dy))
                      SND_US,RCV_US : channel (dy))

played_by Ui

def=

  local State: nat, K1,T1,A1,IDi,Ki,M1,M2,SK,B2 ,Einew: text
  const user_server_sk,user_id:protocol_id

  init State := 0
  transition

   1. State = 0      /\ RCV_US(start)=|>
       State':= 2      /\ Ki1' := new()
                             /\ T1':= new()
                             /\ A1':= exp(P,K1')
                             /\ M1':= xor(Ei,(A1'.IDi))
                             /\ M2':= H(A1',IDi,Ki,Di,T1)
                             /\ SND_US(Ki.M1'.M2'.T1)

   2. State = 2      /\ RCV_US( B2'.
                                                  xor(
                                                   (Einew'.
                                                    H(H(exp(B2',K1).T1).T1).
                                                    H(B2'.
                                                            Einew'.
                                                            H(H(exp(B2',K1).T1).T1).
                                                            Di.
                                                            H(exp(B2',K1).T1))),
                                                    H(Di,T1)
                                                    )
                                                )=|>

  State':= 4      /\ SK':= H(exp(B2',K1).T1)
                             /\ Ei':= Einew'
                             /\ Ki':= H(H(exp(B2',K1).T1).T1)

                             /\ secret(IDi,user_id,{Sj,Ui})
                             /\ witness(Ui,Sj,user_server_sk,SK')
                             /\ request(Ui,Sj,user_server_sk,SK')

end role
```

The role of the server.

```
role server(   Ui,Sj: agent,
                    Di,Ei :symmetric_key,
                    Xgwn   :symmetric_key,
                    H     : hash_func,
                    P   : text,
                    SND_US,RCV_US: channel(dy))
played_by Sj

  def=

  local State:   nat,A1,T1,Ki,IDi,SK,K2,B2,Kinew,Einew,M3,M4: text
  const user_server_sk,user_id:protocol_id
  init   State := 1
  transition

   1. State     = 1     RCV_US( Ki'.
                                             xor(H(Ki'.Xgwn),(A1'.IDi')).
                                             H(A1',IDi',Ki',Di',T1').
                                             T1'
                                             ) =|>

     State' := 3     /\ K2' := new()
                          /\ B2' := exp(P,K2')
                          /\ SK' := exp(A1',K2')
                          /\ Kinew' := H(SK',T1')
                          /\ Einew':= H(Kinew',Xgwn)
                          /\ M3' := H(B2',Einew',Kinew',Di',SK')
                          /\ M4' := xor((Einew'.Kinew'.M3'),H(Di',T1'))
                          /\ SND_US( B2,M4')

                          /\ secret(IDi,user_id,{Sj,Ui})
                          /\ witness(Sj,Ui, user_server_sk,SK')
                          /\ request(Sj,Ui, user_server_sk,SK')
end role
```

The role of the session.

```
role session(Ui, Sj : agent,
                      Di,Ei, Xgwn : symmetric_key,
                              H : hash_func,
                              P : text)

def=

       local         SU,RU,SS,RS:channel(dy)

   composition
       user     (Ui,Sj,    Di,Ei,            H,P, SU,RU)
   /\ server (Ui,Sj,    Di,Ei,Xgwn,    H,P, SS,RS)
end role
```

The role of the environment.

```
role environment()

def=
   const ui,sj : agent,
   di,xgwn,dii,ei: symmetric_key,
   user_server_sk,user_id:protocol_id,
   h : hash_func,
   p    : text

   intruder_knowledge={ui, sj,    dii,eii,xgwni,    h,p}

   composition
                     session(ui,sj,     di,ei,xgwn, h,p)
                /\ session( i,sj,    dii,eii,xgwn, h,p)
                /\ session(ui, i,    di,ei,xgwni, h,p)
end role
```

The role of the goals.

```
goal
   % Confidentiality (G12)
   secrecy_of user_server_sk,user_id

   % Message authentication (G2)
   authentication_on user_server_sk
end goal
```

Appendix B

The parameters of the Koblitz curve secp256k1 by NIST are listed in this part. The curve is defined as $E: y^2 = x^3 + ax + b$ over F_p. The bit length of p is 256 bit.

p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F
a = 0
b = 7
G_x = 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B
G_y = 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F
n = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141
h = 01

Reference

1. Almenares, F.; Arias, P.; Marin, A.; Diaz-Sanchez, D.; Sanchez, R. Overhead of using secure wireless communications in mobile computing. *IEEE Trans. Consum. Electron.* **2013**, *59*, 335–342.
2. Potlapally, N.R.; Ravi, S.; Raghunathan, A.; Jha, N.K. A study of the energy consumption characteristics of cryptographic algorithms and security protocols. *IEEE Trans. Mob. Comput.* **2006**, *5*, 128–143.
3. Tu, H.; Kumar, N.; Chilamkurti, N.; Rho, S. An improved authentication protocol for session initiation protocol using smart card. *Peer-to-Peer Netw. Appl.* **2015**, *8*, 903–910.
4. Farash, M.S. Security analysis and enhancements of an improved authentication for session initiation protocol with provable security. *Peer-to-Peer Netw. Appl.* **2016**, *9*, 82–91.
5. Ibrahim, M.H.; Kumari, S.; Das, A.K.; Wazid, M.; Odelu, V. Secure anonymous mutual authentication for star two-tier wireless body area networks. *Comput. Methods Programs Biomed.* **2016**, *135*, 37–50.
6. Chaudhry, S.A.; Naqvi, H.; Mahmood, K.; Ahmad, H.F.; Khan, M.K. An Improved Remote User Authentication Scheme Using Elliptic Curve Cryptography. *Wirel. Pers. Commun.* **2017**, *96*, 5355–5373.
7. Kumari, S.; Chaudhry, S.A.; Wu, F.; Li, X.; Farash, M.S.; Khan, M.K. An improved smart card based authentication scheme for session initiation protocol. *Peer-to-Peer Netw. Appl.* **2017**, *10*, 92–105.
8. Liu, J.; Zhang, L.; Sun, R. 1-RAAP: An Efficient 1-Round Anonymous Authentication Protocol for Wireless Body Area Networks. *Sensors* **2016**, *16*, 728.
9. Xiong, H. Cost-Effective Scalable and Anonymous Certificateless Remote Authentication Protocol. *IEEE Trans. Inf. Forensics Secur.* **2014**, *9*, 2327–2339.
10. Liu, J.; Zhang, Z.; Chen, X.; Kwak, K.S. Certificateless Remote Anonymous Authentication Schemes for WirelessBody Area Networks. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *25*, 332–342.
11. Idrissi, H. Anonymous ECC-Authentication and Intrusion Detection Based on Execution Tracing for Mobile Agent Security. *Wirel. Pers. Commun*. **2017**, *94*, 1799–1824.
12. Xiong, L.; Peng, D.; Peng, T.; Liang, H.; Liu, Z. A Lightweight Anonymous Authentication Protocol with Perfect Forward Secrecy for Wireless Sensor Networks. *Sensors* **2017**, *17*, 2681.
13. Li, X.; Ibrahim, M.H.; Kumari, S.; Sangaiah, A.K.; Gupta, V.; Choo, K.-K.R. Anonymous mutual authentication and key agreement scheme for wearable sensors in wireless body area networks. *Comput. Netw*. **2017**, *129*, 429–443.
14. Kumari, S.; Khan, M.K. Cryptanalysis and improvement of "a robust smart-card-based remote user password authentication scheme". *Int. J. Commun. Syst.* **2014**, *27*, 3939–3955.
15. Jiang, Q.; Ma, J.; Li, G.; Yang, L. An Efficient Ticket Based Authentication Protocol with Unlinkability for Wireless Access Networks. *Wirel. Pers. Commun*. **2014**, *77*, 1489–1506.
16. Li, X.; Niu, J.; Kumari, S.; Liao, J.; Liang, W.; Khan, M.K. A new authentication protocol for healthcare applications using wireless medical sensor networks with user anonymity. *Secur. Commun. Netw.* **2015**, *15*, 2643–2655.
17. Wu, F.; Xu, L.; Kumari, S.; Li, X. A new and secure authentication scheme for wireless sensor networks with formal proof. *Peer-to-Peer Netw. Appl.* **2017**, *10*, 16–30.
18. Das, A.K. A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks. *Peer-to-Peer Netw. Appl.* **2016**, *9*, 223–244.
19. Wu, F.; Xu, L.; Kumari, S.; Li, X. A novel and provably secure biometrics-based three-factor remote authentication scheme for mobile client–server networks. *Comput. Electr. Eng.* **2015**, *45*, 274–285.
20. Odelu, V.; Das, A.K.; Goswami, A. An efficient biometric-based privacy-preserving three-party authentication with key agreement protocol using smart cards. *Secur. Commun. Netw.* **2015**, *8*, 4136–4156.
21. Wang, C.; Xu, G.; Sun, J. An Enhanced Three-Factor User Authentication Scheme Using Elliptic Curve Cryptosystem for Wireless Sensor Networks. *Sensors* **2017**, *17*, 2946.
22. Islam, S.H. Provably secure dynamic identity-based three-factor password authentication scheme using extended chaotic maps. *Nonlinear Dyn.* **2014**, *78*, 2261–2276.
23. He, D.; Zeadally, S.; Xu, B.; Huang, X. An Efficient Identity-Based Conditional Privacy-Preserving Authentication Scheme for Vehicular Ad Hoc Networks. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 2681–2691.
24. Islam, S.H.; Khan, M.K. Provably secure and pairing-free identity-based handover authentication protocol for wireless mobile networks. *Int. J. Commun. Syst.* **2016**, *29*, 2442–2456.
25. Porambage, P.; Schmitt, C.; Kumar, P.; Gurtov, A.; Ylianttila, M. PAuthKey: A Pervasive Authentication Protocol and Key Establishment Scheme for Wireless Sensor Networks in Distributed IoT Applications. *Int. J. Distrib. Sens. Netw.* **2014**, *2014*, e357430.

26. Tsai, J.L.; Lo, N.W. A Privacy-Aware Authentication Scheme for Distributed Mobile Cloud Computing Services. *IEEE Syst. J.* **2015**, *9*, 805–815.
27. Mishra, D.; Das, A.K.; Mukhopadhyay, S. A secure and efficient ECC-based user anonymity-preserving session initiation authentication protocol using smart card. *Peer-to-Peer Netw. Appl.* **2016**, *9*, 171–192.
28. Li, X.; Peng, J.; Kumari, S.; Wu, F.; Karuppiah, M.; Choo, K.K. An enhanced 1-round authentication protocol for wireless body area networks with user anonymity. *Comput. Electr. Eng.* **2017**, doi:10.1016/j.compeleceng.2017.02.011.
29. Nam, J.; Choo, K.K.; Han, S.; Kim, M.; Paik, J.; Won, D. Efficient and anonymous two-factor user authentication in wireless sensor networks: Achieving user anonymity with lightweight sensor computation. *PLoS ONE* **2015**, *10*, e0116709.
30. Armando, A.; Basin, D.; Boichut, Y.; Chevalier, Y.; Compagna, L.; Cuéllar, J.; Drielsma, P.H.; Héam, P.C.; Kouchnarenko, O.; Mantovani, J.; et al. The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications. In Proceedings of the International Conference on Computer Aided Verification, Edinburgh, UK, 6–10 July 2005; Springer: Berlin/Heidelberg, Germany, 2005; pp. 281–285. Available online: http://link.springer.com/chapter/10.1007/11513988_27 (accessed on 29 October 2018).
31. Turuani, M. The CL-Atse Protocol Analyser. In *Lecture Notes in Computer Science, Proceedings of the 17th International Conference on Rewriting Techniques and Applications, RTA, Seattle, WA, USA, 12–14 August 2006*; Pfenning, F., Ed.; Springer: Berlin/Heidelberg, Germany, 2006.
32. Basin, D.; Mödersheim, S.; Vigano, L. Constraint Differentiation: A New Reduction Technique for Constraint-Based Analysis of Security Protocols. In Proceedings of the CCS’03, Washington, DC, USA, 27–30 October 2003; Atluri, V., Liu, P., Eds.; ACM Press: New York, NY, USA, 2003; pp. 335–344. Available online: http://www.avispa-project.org (accessed on 29 October 2018).
33. Basin, D.; Mödersheim, S.; Vigano, L. OFMC: A Symbolic Model-Checker for Security Protocols. *Int. J. Inf. Secur.* **2005**, *4*, 181–208.
34. Burrows, M.; Abadi, M.; Needham, R.M. A Logic of Authentication. *Proc. R. Soc. Lond. A Math. Phys. Eng. Sci.* **1989**, *426*, 233–271, doi:10.1098/rspa.1989.0125.
35. Chen, Y.; Martínez, J.F.; Castillejo, P.; López, L. A Privacy Protection User Authentication and Key Agreement Scheme Tailored for the Internet of Things Environment: PriAuth. *Wirel. Commun. Mob. Comput.* **2017**, *2017*, 5290579, doi:10.1155/2017/5290579.
36. Available online: https://github.com/SevenBruce/lAuth (accessed on 28 September 2018).
37. Available online: https://libraries.docs.miracl.com/miracl-user-manual/about (accessed on 1 March 2018).
38. Available online: https://github.com/miracl/MIRACL (accessed on 28 September 2018).
39. Available online: https://csrc.nist.gov/csrc/media/publications/fips/186/3/archive/2009-06-25/documents/fips_186-3.pdf (accessed on 3 April 2018).