IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

SPECIAL SECTION ON MOBILE SERVICE COMPUTING WITH INTERNET OF THINGS

Received March 28, 2019, accepted April 14, 2019, date of publication April 23, 2019, date of current version May 3, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2912870*

# A Lightweight Mutual Authentication and Key Agreement Scheme for Medical Internet of Things

**ZISANG XU[1], CHENG XU[1], WEI LIANG[2], JIANBO XU[3], AND HAIXIAN CHEN[1]**

[1]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
[2]School of Opto-Electronic and Communication Engineering, Xiamen University of Technology, Xiamen 361024, China
[3]School of Computer science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China

Corresponding author: Cheng Xu (chengxu@hnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772185, Grant 61872138, and Grant 61572188, and in part by the Natural Science Fund Project in Hunan Province under Grant 2016JJ2058.

**ABSTRACT** Wireless body area networks play an indispensable role in the medical Internet of Things. It is a network of several wearables or implantable devices that use wireless technologies to communicate. These devices usually collect the wearer's physiological data and send it to the server. Some health care providers can access the server over the network and provide medical care to the wearer. Due to the openness and mobility of the wireless network, the adversary can easily steal and forge information, which exchanged in the communication channel that leaks wearer's privacy. Therefore, a secure and reliable authentication scheme is essential. Most of the existing authentication schemes are based on asymmetric encryption. However, since the sensor devices in wireless body area networks are typically resource-constrained devices, their computing resources cannot afford to use asymmetric encryption. In addition, most of the existing lightweight authentication schemes have various security vulnerabilities, especially the lack of forwarding secrecy. Therefore, we propose a secure lightweight authentication scheme for the wireless body area networks. With this scheme, forward secrecy can be guaranteed without using asymmetric encryption. We use the automatic security verification tool ProVerif to verify the security of our scheme and analyze informal security. The experimental results and the theoretical analysis indicate that our scheme significantly reduces the computational cost compared with the schemes using asymmetric encryption and that it has a lower security risk compared with the lightweight schemes.

**INDEX TERMS** Authentication, IoT, security, wireless body area network.

## I. INTRODUCTION

The Internet of Things (IoT) will consist of 50 billion connected devices by 2020. There is no doubt that we are in an IoT era [1]–[3]. The medical Internet of Things, which is formed by the IoT and medical applications, has developed rapidly, and the wireless body area network (WBAN) is an indispensable part of the medical Internet of Things. A WBAN is a network of several wearable or implantable devices that communicate using wireless technology [4], [5]. Figure 1 shows a typical WBAN architecture. Some wearable or implantable devices attached to the human body monitor the patient's physiological data and send the data to a server. However, since these wearable or implantable devices are resource-constrained devices, they usually need to communicate with the server through intermediate nodes, which are usually mobile phones, routers, etc. A doctor or other medical service provider can access the server via the Internet and will remotely diagnose or provide other medical services for the patient. This will increase the efficiency of medical services, even promoting healthy living styles.

The patient's physiological data is extremely sensitive and private, however, due to the openness and mobility of the wireless network, it is easily stolen or forged by an adversary, which will lead to extremely serious consequences and may even endanger the lives of patients. Therefore, the WBAN needs a secure and reliable authentication and key agreement scheme to ensure that only legitimate devices can access the server and the confidentiality of the transmitted data. Most of existing authentication schemes are based

The associate editor coordinating the review of this manuscript and approving it for publication was Shuiguang Deng.

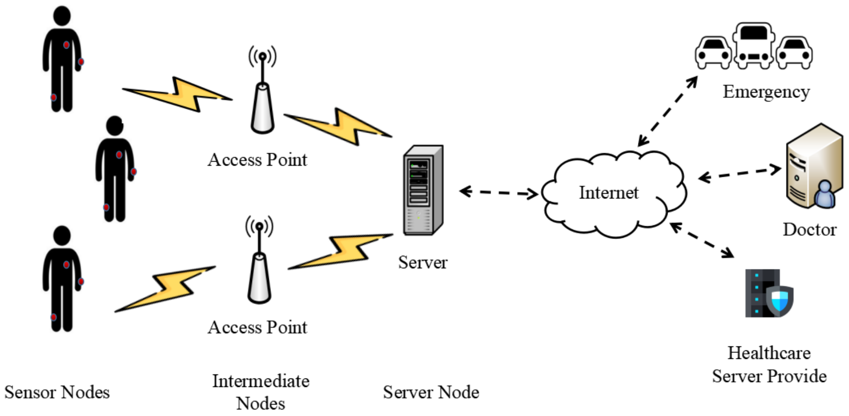


**FIGURE 1.** The architecture of WBAN.

on asymmetric encryption, such as elliptic curve cryptography (ECC). However, the wearable or implantable devices in WBANs are resource-constrained devices, and their computing resources cannot afford to use asymmetric encryption, so only lightweight authentication schemes can be used. Moreover, the sensor nodes in the WBAN are easy to be captured by the adversary and obtain all the data stored in its memory [6]. Therefore, once the sensor node is captured by the adversary, it is difficult for the lightweight authentication scheme to guarantee the forward secrecy of the sensor node [7].

Our scheme focuses on ensuring forward secrecy and lightweight. Since the sensor nodes in the WBAN are easily captured by the adversary, we add a parameter that will never be transmitted in the channel on both sides of the authentication and ensure that the parameter is different in each round of authentication. This parameter will be used as part of the session key to ensure forward secrecy.

The contributions of this paper are given as follows.

- We propose a lightweight and anonymous mutual authentication and key agreement scheme for WBAN. It only needs to perform hash function operations and XOR operations.
- With this scheme, forward secrecy can be guaranteed without using asymmetric encryption.
- We use the automatic security verification tool ProVerif to verify the security of our scheme and analyze informal security.
- Compared to lightweight authentication schemes proposed for WBAN, our scheme has lower security risks, especially guaranteeing forward secrecy. Compared to authentication schemes based on asymmetric encryption, our scheme significantly reduces the computational cost while ensuring security.

The remaining parts of this paper are arranged as follows. The related work is discussed in Section 2. In Section 3, we introduce the network model and the threat model. The proposed scheme is demonstrated completely in Section 4. In Section 5, we use the ProVerif tool to perform security analysis on the proposed scheme and discussed informal security analysis. Section 6 shows the performance analysis and the comparison with related works. Finally, we conclude the paper in Section 7.

## II. RELATED WORK

Recently, some physiological signal-based schemes have been proposed, which usually design an authentication scheme using an electrocardiogram (ECG) as a biometric key. In 2018, Koya and Deepthi [8] proposed an anonymous hybrid mutual authentication and key agreement scheme based on ECG. However, their scheme is vulnerable to sensor node capture attack and lacks forward secrecy. In addition, physiological signal-based authentication schemes typically require sensors to monitor unique physiological signals, such as ECG, which makes such schemes lack universality.

The channel-based schemes [9], [10] usually assume that the channel between a pair of communication parts is reciprocal, and both communication parts will capture the characteristics of the channel as authentication parameters. Since the channel is reciprocal, the authentication parameters captured by both communication parts during authentication phase should be the same. However, such assumptions are not always correct. The characteristics and features of the channel will change from time to time. Therefore, such schemes are difficult to apply in reality.

Most of existing authentication schemes are based on asymmetric encryption. In 2014, Turkanovic *et al.* [11] proposed a authentication and key agreement scheme for wireless sensor networks (WSNs) based on elliptic curve cryptography (ECC). However, Chang and Le [12] pointed out that the scheme of Turkanovic *et al.* [11] could not resistant to stolen smart card and smart card breach attack and failed to achieve forward and backward secrecy. In addition, when the sensor node is captured, the scheme is also vulnerable to impersonation attack and sensor node spoofing attack. Chang and Le fixed these problems and proposed a new authentication scheme based on ECC. In 2018, Li *et al.* [13] pointed out that Chang and Le's scheme [12] is short of proper mutual authentication, not applicable to practical applications, and vulnerable to stolen smart card attack and tracking attack. Li *et al.* [13] proposed a robust and energy efficient authentication protocol and fixed the above problem. In 2016, Gope and Hwang [14] proposed an efficient mutual authentication and key agreement scheme for global mobility networks. However, in 2018, Li *et al.* [15] pointed out that Gope and Hwang's scheme [14] lacks session key update and wrong password detection mechanisms, lacks forward secrecy, and is vulnerable to denial-of-service attack. They fixed these problems and proposed an improvement scheme for global mobility network. But their scheme could not resist replay attacks. Schemes [11]–[15] are all based on asymmetric encryption schemes, which usually have a lower security risk. However, these schemes usually require a lot of computing resources which the sensor nodes in WBAN may not be affordable.

Janbabaei *et al.* [16] proposed an anonymous mutual authentication scheme for IoT infrastructure in 2016. However, this scheme is only for authentication between different sensor nodes in the same server. In 2018, Hwang *et al.* [17] proposed an authentication scheme based on the multi-server model. There are two servers in their scheme, where the TLS server is responsible for the TLS handshake protocol and the application server is responsible for all other tasks. However, the network models of the Janbabaei *et al.* [16] and Hwang *et al.* [17] schemes are different from our scheme.

Both schemes [18] and [19] are lightweight authentication schemes for WBAN, but they all have multiple security vulnerabilities. In 2016, Ibrahim *et al.* [18] proposed an anonymous mutual authentication and key agreement scheme for WBAN. However, during the authentication phase of this scheme, the parameters for authentication need to be updated sequentially between the sensor node and the server. It makes this scheme vulnerable to jamming/desynchronization attack. In 2017, Li *et al.* [19] proposed a scheme that can fix the vulnerability of jamming/desynchronization attack. But their scheme has the risk of impersonation attack after capturing a sensor node, and lacks forward secrecy.

## III. SYSTEM MODELS

### A. NETWORK MODEL

Our network model adopts a two-hop centralized architecture. Figure 2 shows the network model of our scheme. There are three different types of nodes in our network model, the sensor node (SN), the access point (AP), and the server. The SN needs to be verified by the server and it is a resource-constrained device. The AP acts as an intermediate node between the SN and the server. It has more resources than the SN and is usually a mobile phone or router. The server is rich in resources and is usually a server. We assume that the AP can always communicate directly with the server. However, due to the limited communication power, the SN is not always within the communication range of the server. In order to ensure that the SN can always communicate with the server, we assume that there is always at least one AP communicating directly with the SN.

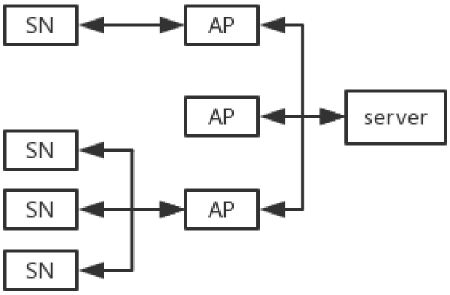


FIGURE 2. The network model of our scheme.

Mutual authentication is required before data is exchanged between the SN and the server. The authentication process is as follows. First, the SN sends the authentication information to the AP within the communication range. Second, the AP forwards the received information to the server. Third, after the server authenticates the SN, it generates a session key and sends authentication information to the AP. Fourth, the AP forwards the received information to the SN. Fifth, after the SN authenticates the server, it generates a session key. Now the session key can be used to encrypt the information that needs to be exchanged between the SN and the server.

### B. THREAT MODEL

The threat model we use is as follows:

- The server is assumed to be trusted node. However, an adversary can infiltrate the server's database and obtain all the data in it, except the master key $K_{ser}$.
- The adversary is able to obtain all the data exchanged in the channel, and he can inject new data and replace or replay the previously sent data.
- We assume that the SN is not physically protected because of cost constraints. Therefore, once the adversary compromises an SN, he can extract the secret information stored in its memory. We also assume that the adversary is able to capture as many SNs as possible.
- We use the well-known Dolev-Yao threat model [20], which assumes that the two parties communicate in an insecure channel.

## IV. PROPOSED SCHEME

Table 1 shows the important symbols used in our scheme. In our scheme, $P_{Ks}$ is an important parameter used to ensure forward secrecy. First, $P_{Ks}$ has never been transmitted over the channel. Second, $P_{Ks}$ is different for each round of certification. Third, $P_{Ks}$ will be used as a parameter to generate session key $Ks$. This makes it impossible for an adversary to obtain any historical $P_{Ks}$, which prevents him from obtaining any session key generated by the authentication.

Our scheme has three phases: the initialization phase, the registration phase, and the authentication phase. The initialization phase and the registration phase are executed by

**TABLE 1. Symbols used in our protocol.**

| Symbol | Description |
|---|---|
| SA | System administrator |
| SN | Sensor node requesting authentication |
| AP | Access point (Intermediate node) |
| Server | Server node |
| $K_{ser}$ | Master secret key of the server |
| $ID_{SN}$ | Permanent identity of the SN |
| $ID_{AP}$ | Permanent identity of the AP |
| $A_{SN}, B_{SN}, X$ | Authentication parameters |
| $P_{Ks}$ | The hash result of the previous session key |
| $r, n1, n2$ | Temporary secret parameters picked by the server |
| $t1, t2$ | Timestamp generated by the SN or server |
| $Ks$ | Session key to be agreed on |
| $(a, b)$ | Concatenation of data $a$ and data $b$ |
| $\oplus$ | Bitwise XOR operation |
| $X \rightarrow Y : Z$ | Entity $X$ sends the message $Z$ to entity $Y$ via a public channel |

the system administrator (SA) in a secure environment. In the authentication phase, the SN exchanges information with the server on insecure channel and performs authentication and key agreement scheme. The three phases of our proposed scheme are described in detail as follows:

### A. INITIALIZATION PHASE

The process of SA initializing the server is as follows.

**Step I1** The SA generates a master key $K_{ser}$ for the server.

**Step I2** The SA stores master key $K_{ser}$ in the server's memory.

### B. REGISTRATION PHASE

The process of SA registering SNs and APs is as follows.

**Step R1** For each SN, the server generates a unique identity $ID_{SN}$, a unique r and a random $P_{Ks}$. For each AP, the server generates a unique identity $ID_{AP}$.

**Step R2** The SA computes $A_{SN} = r \oplus K_{ser}$, $B_{SN} = h(r, K_{ser})$, $X = ID_{SN} \oplus h(r, K_{ser})$.

**Step R3** The SA stores tuple $(ID_{SN}, A_{SN}, B_{SN}, P_{Ks})$ in the SN's memory, and stores $A_{SN}, X$ and $P_{Ks}$ as a tuple $<A_{SN}, X, P_{Ks}>$ in the server's memory. The server may store multiple such tuples.

**Step R4** The SA stores the AP's identity $ID_{AP}$ in the server's memory.

### C. AUTHENTICATION PHASE

Table 2 shows the authentication phase. The authentication phase between the SN and the server is as follows.

**Step A1** SN→AP: $(A_{SN}, S1, S2, t1)$, the SN performs as follows.

- Generates $n1$ and timestamp $t1$.
- Computes $S1 = B_{SN} \oplus n1$, $S2 = h(ID_{SN}, A_{SN}, S1, t1, n1)$.
- Sends message $(A_{SN}, S1, S2, t1)$ to the AP.

**Step A2** AP→server: $(A_{SN}, S1, S2, t1, ID_{AP})$.

- The AP simply forwards the information received by the SN to the server and only places its identity $ID_{AP}$ to the message.

**TABLE 2. The premises in our code.**

| The premises of channels, constants, equations, reductions, and functions |
|---|
| (*——events———*) |
| free c:channel. |
| (*——constants——*) |
| free IDSN:bitstring [private]. |
| free Kser:bitstring [private]. |
| free r:bitstring [private]. |
| (*——functions,reductions and equations———*) |
| fun xor(bitstring,bitstring):bitstring.(*XOR operation*) |
| equation forall m:bitstring,n:bitstring;xor(xor(m,n),n)=m. |
| fun con(bitstring,bitstring):bitstring.(*string concatenation*) |
| fun h(bitstring):bitstring.(*hash function*) |

**Step A3** server→AP: $(S3, S4, S5, S6, t2, ID_{AP})$, the server performs as follows.

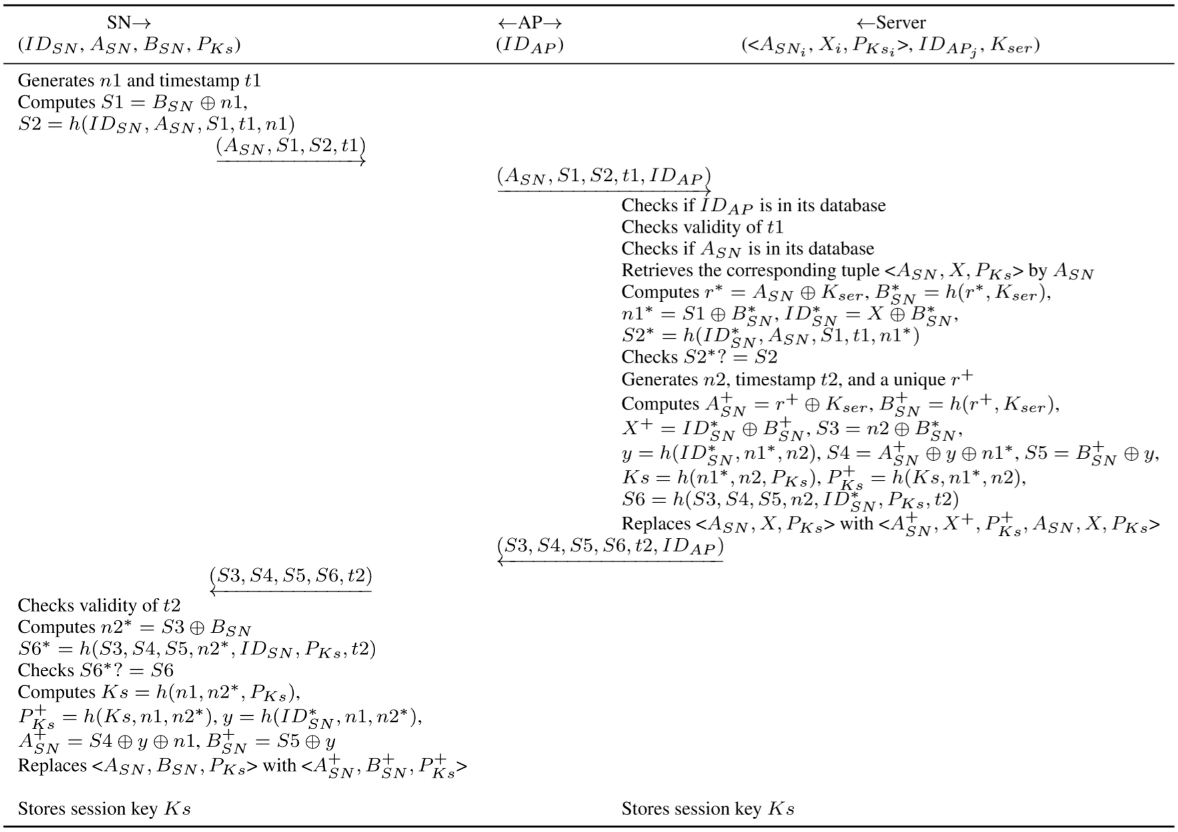
- Checks if $ID_{AP}$ is in its database. Terminates the session if the check fails.
- Checks the condition $t_{new} - t1 < \Delta t$ holds or not, where $t_{new}$ is the time of the message was received and $\Delta t$ is the maximum communication delay. Terminates the session if the condition does not hold.
- Checks if $A_{SN}$ is in its database. Terminates the session if the check fails.
- Retrieves the corresponding tuple $<A_{SN}, X, P_{Ks}>$ by $A_{SN}$.
- Computes $r^* = A_{SN} \oplus K_{ser}$, $B^*_{SN} = h(r^*, K_{ser})$, $n1^* = S1 \oplus B^*_{SN}$, $ID^*_{SN} = X \oplus B^*_{SN}$, $S2^* = h(ID^*_{SN}, A_{SN}, S1, t1, n1^*)$.
- Checks $S2^*? = S2$. Terminates the session if the check fails.
- Generates $n2$, timestamp $t2$, and a unique $r^+$.
- Computes $A^+_{SN} = r^+ \oplus K_{ser}$, $B^+_{SN} = h(r^+, K_{ser})$, $X^+ = ID^*_{SN} \oplus B^+_{SN}$, $S3 = n2 \oplus B^*_{SN}$, $y = h(ID^*_{SN}, n1^*, n2)$, $S4 = A^+_{SN} \oplus y \oplus n1^*$, $S5 = B^+_{SN} \oplus y$, $Ks = h(n1^*, n2, P_{Ks})$, $P^+_{Ks} = h(Ks, n1^*, n2)$, $S6 = h(S3, S4, S5, n2, ID^*_{SN}, P_{Ks}, t2)$.
- Replaces the tuple $<A_{SN}, X, P_{Ks}>$ with the tuple $<A^+_{SN}, X^+, P^+_{Ks}, A_{SN}, X, P_{Ks}>$ in its memory.
- Sends message $(S3, S4, S5, S6, t2, ID_{AP})$ to the AP.

**Step A4** AP→SN: $(S3, S4, S5, S6, t2)$.

- The AP simply forwards the information received by the server to the SN, and drops its identity $ID_{AP}$.

**Step A5** SN: the SN performs as follows.

- Checks the condition $t_{new} - t2 < \Delta t$ holds or not, where $t_{new}$ is the time of the message was received and $\Delta t$ is the maximum communication delay. Terminates the session if the condition does not hold.
- Computes $n2^* = S3 \oplus B_{SN}$, $S6^* = h(S3, S4, S5, n2^*, ID_{SN}, P_{Ks}, t2)$.
- Checks $S6^*? = S6$. Terminates the session if the check fails.
- Computes $Ks = h(n1, n2^*, P_{Ks})$, $P^+_{Ks} = h(Ks, n1, n2^*)$, $y = h(ID^*_{SN}, n1, n2^*)$, $A^+_{SN} = S4 \oplus y \oplus n1$, $B^+_{SN} = S5 \oplus y$.
- Replaces the parameters $A_{SN}, B_{SN}, P_{Ks}$ with the parameters $A^+_{SN}, B^+_{SN}, P^+_{Ks}$ respectively in its memory.



SN→ $(ID_{SN}, A_{SN}, B_{SN}, P_{Ks})$

←AP→ $(ID_{AP})$

←Server $(<A_{SN_i}, X_i, P_{Ks_i}>, ID_{AP_j}, K_{ser})$

Generates $n1$ and timestamp $t1$
Computes $S1 = B_{SN} \oplus n1$,
$S2 = h(ID_{SN}, A_{SN}, S1, t1, n1)$

$\xrightarrow{(A_{SN}, S1, S2, t1)}$

$\xrightarrow{(A_{SN}, S1, S2, t1, ID_{AP})}$

Checks if $ID_{AP}$ is in its database
Checks validity of $t1$
Checks if $A_{SN}$ is in its database
Retrieves the corresponding tuple $<A_{SN}, X, P_{Ks}>$ by $A_{SN}$
Computes $r^* = A_{SN} \oplus K_{ser}$, $B^*_{SN} = h(r^*, K_{ser})$,
$n1^* = S1 \oplus B^*_{SN}$, $ID^*_{SN} = X \oplus B^*_{SN}$,
$S2^* = h(ID^*_{SN}, A_{SN}, S1, t1, n1^*)$
Checks $S2^* ? = S2$
Generates $n2$, timestamp $t2$, and a unique $r^+$
Computes $A^+_{SN} = r^+ \oplus K_{ser}$, $B^+_{SN} = h(r^+, K_{ser})$,
$X^+ = ID^*_{SN} \oplus B^+_{SN}$, $S3 = n2 \oplus B^*_{SN}$,
$y = h(ID^*_{SN}, n1^*, n2)$, $S4 = A^+_{SN} \oplus y \oplus n1^*$, $S5 = B^+_{SN} \oplus y$,
$Ks = h(n1^*, n2, P_{Ks})$, $P^+_{Ks} = h(Ks, n1^*, n2)$,
$S6 = h(S3, S4, S5, n2, ID^*_{SN}, P_{Ks}, t2)$
Replaces $<A_{SN}, X, P_{Ks}>$ with $<A^+_{SN}, X^+, P^+_{Ks}, A_{SN}, X, P_{Ks}>$

$\xleftarrow{(S3, S4, S5, S6, t2, ID_{AP})}$

$\xleftarrow{(S3, S4, S5, S6, t2)}$

Checks validity of $t2$
Computes $n2^* = S3 \oplus B_{SN}$
$S6^* = h(S3, S4, S5, n2^*, ID_{SN}, P_{Ks}, t2)$
Checks $S6^* ? = S6$
Computes $Ks = h(n1, n2^*, P_{Ks})$,
$P^+_{Ks} = h(Ks, n1, n2^*)$, $y = h(ID^*_{SN}, n1, n2^*)$,
$A^+_{SN} = S4 \oplus y \oplus n1$, $B^+_{SN} = S5 \oplus y$
Replaces $<A_{SN}, B_{SN}, P_{Ks}>$ with $<A^+_{SN}, B^+_{SN}, P^+_{Ks}>$

Stores session key $Ks$

Stores session key $Ks$

**FIGURE 3.** The authentication phase of our scheme.

## V. SECURITY ANALYSIS

This section is divided into two parts. In the first part, we use the automatic security verification tool ProVerif [21] to formally verify the security of our scheme. In the second part, we discuss informal security.

### A. SIMULATION BASED ON PROVERIF TOOL

ProVerif is a widely used automatic cryptographic protocol verifier. It can handle many symmetric and public-key cryptographic primitives, including encryption and signatures, hash functions, and Diffie-Hellman key agreements, which are specified as rewritten rules or as equations [22]. Table 2 shows the premises containing channels, constants, equations, reductions, and functions in our code. There is a correspondence relation among three events: *SNAcServer, ServerAcSN*, and *End*. The *event SNAcServer* means the SN successfully authenticated the server. The *event ServerAcSN* means the server successfully authenticated the SN. The *event End* means the end of the authentication process. We also need to use ProVerif to verify whether the parameters $K_{ser}$ and $ID_{SN}$ can be obtained by the adversary. Table 3 shows the events and the queries in our code. In the simulation, we removed the AP from the network model. Because the AP is an intermediate node, it only forwards the received information without doing anything, and the identity of the AP $ID_{AP}$ is not a secret parameter. Table 4 shows the process of the SN and the server in our code.

**TABLE 3.** The events and the queries in our code.

| (*——events——-*) | (*——queries——-*) |
|---|---|
| event SNAcServer().<br>event ServerAcSN().<br>event End(). | query attacker (Kser).<br>query attacker (IDSN).<br>query inj-event(SNAcServer())==><br>inj-event(ServerAcSN()).<br>query inj-event(End())==><br>inj-event(ServerAcSN()). |

Finally, We use the following code to start the verification.

*process*
*new PKs:bitstring;*
*let ASN = xor(r, Kser) in*
*let BSN = h(con(r, Kser)) in*
*let X = xor(IDSN, h(con(r, Kser))) in*
*((!SN(IDSN, ASN, BSN, PKs))|(!server(ASN, X, PKs, Kser)))*

The results are as follow.

*RESULT not attacker(Kser[]) is true.*

*RESULT not attacker(IDSN[]) is true.*

*RESULT inj-event(SNAcServer) ==> inj-event (ServerAcSN) is true.*

*RESULT inj-event(End) ==> inj-event(ServerAcSN) is true.*

**TABLE 4. The process of the server and the SN.**

```
The process of the SN and the server
(*—-process of the SN—-*)
let SN (IDSN:bitstring, ASN:bitstring, BSN:bitstring, PKs:bitstring) =
new t1:bitstring;
new n1:bitstring;
let S1 = xor(BSN, n1) in
let S2 = h(con(IDSN, con(ASN, con(S1, con(t1, n1))))) in
out (c, (ASN, S1, S2, t1));
in(c, (S3:bitstring, S4:bitstring, S5:bitstring, S6:bitstring, t2:bitstring));
let tn2 = xor(S3, BSN) in
let tS6 = h(con(S3,con(S4,con(S5,con(tn2,con(IDSN,con(PKs,t2))))))) in
if (tS6 = S6) then event SNAcServer();
let Ks = h(con(n1, con(tn2, PKs))) in
let nPKs = h(con(Ks, con(n1, tn2))) in
let y = h(con(IDSN, con(n1, tn2))) in
let nASN = xor(S4, xor(y, n1)) in
let nBSN = xor(S5, y) in
event End().

(*—-process of the server—-*)
let server(ASN:bitstring, X:bitstring, PKs:bitstring, Kser:bitstring)=
in (c,(ASN:bitstring, S1:bitstring, S2:bitstring, t1:bitstring));
let tr = xor(ASN,Kser) in
let tBSN = h(con(tr,Kser)) in
let tn1 = xor(S1, tBSN) in
let tIDSN = xor(X, h(con(tr, Kser))) in
let tS2 = h(con(tIDSN, con(ASN, con(S1, con(t1, tn1))))) in
if (tS2 = S2) then event ServerAcSN();
new n2:bitstring;
new nr:bitstring;
new t2:bitstring;
let nASN = xor(nr, Kser) in
let nBSN = h(con(nr, Kser)) in
let nX = xor(tIDSN, nBSN) in
let S3 = xor(n2, tBSN) in
let y = h(con(IDSN, con(tn1, n2))) in
let S4 = xor(nASN, xor(y, tn1)) in
let S5 = xor(nBSN,y) in
let Ks = h(con(tn1, con(n2, PKs))) in
let nPks = h(con(Ks, con(tn1, n2))) in
let S6 = h(con(S3,con(S4,con(S5,con(n2,con(tIDSN, con(PKs,t2))))))) in
out(c, (S3, S4, S5, S6, t2)).
```

They mean that our scheme is secure after analysis by ProVerif. The adversary cannot get the parameters $K_{ser}$ and $ID_{SN}$. All related events are executed normally.

## *B. DISCUSSIONS OF OTHER ATTACKS*

### 1) EAVESDROPPING ATTACK

The adversary can obtain all the transmitted information from the common channel. This means that the adversary can obtain the parameters $A_{SN}, S1, S2, t1, S3, S4, S5, S6, t2, ID_{AP}$ transmitted during each round of authentication. We need to protect the parameters $ID_{SN}$, $K_{ser}$ and the session key $Ks$. First of all, the adversary cannot obtain the master key $K_{ser}$ from $A_{SN}$ because he does not know $r$, and $r$ is random and fresh. Secondly, the adversary cannot obtain the parameters $n1$ and $n2$ from $S1$ or $S3$. Because the adversary does not know $B_{SN}$. In addition, $n1$ and $n2$ are random and fresh, which leads to the adversary cannot obtain any useful information from the parameters $S4$ and $S5$, because $S4 = A_{SN}^{+} \oplus h(ID_{SN}, n1, n2) \oplus n1$ and $S5 = B_{SN}^{+} \oplus h(ID_{SN}, n1, n2)$, the adversary does not know $n1$, $n2$, and $ID_{SN}$. Finally, the adversary cannot obtain any one parameter from $S2$ and $S6$, because all parameters in $S2$ and $S6$ are protected by the one-way function $h(.)$. Therefore, our scheme prevents eavesdropping attack.

### 2) SENSOR NODE ANONYMITY AND UNTRACEABILITY

The identity of the SN $ID_{SN}$ is never transmitted directly in the channel. From the previous section, we also showed that the adversary could not obtain $ID_{SN}$ through the eavesdropping attack. Moreover, the parameters $r$, $n1$, $n2$, $t1$, and $t2$ for each round of authentication phase are random and fresh. It makes the parameters $A_{SN}$, $S1$, $S2$, $S3$, $S4$, $S5$, and $S6$ are also different in each round of authentication phase. As for $ID_{AP}$, an AP is usually connected to multiple SNs. It is impossible to track a SN by $ID_{AP}$ alone. Therefore, our scheme guarantees the SN anonymity and untraceability.

### 3) SENSOR NODE IMPERSONATION ATTACK

We assume that the adversary compromised a SN and got the tuple $(ID_{SN}, A_{SN}, B_{SN}, P_{Ks})$ stored in its memory. At this point, the adversary cannot obtain any useful parameters from $B_{SN}$ and $P_{Ks}$ because all parameters are protected by the one-way function $h(.)$. The adversary also cannot obtain $r$ or $K_{ser}$ from $A_{SN} = r \oplus K_{ser}$ because $r$ is random and fresh. Moreover, all $ID_{SN}$ and $P_{Ks}$ are different in different SNs, and $ID_{SN}$ require authentication by the server. Therefore, the adversary cannot create a new valid tuple $(ID_{SN}, A_{SN}, B_{SN}, P_{Ks})$ or $(A_{SN}, S1, S2, t1)$.

### 4) REPLAY ATTACK

When the SN sends information to the server, it will generate a timestamp $t1$ and compute $S2 = h(ID_{SN}, A_{SN}, S1, t1, n1)$. The adversary cannot obtain $ID_{SN}$ and $n1$, so it is impossible to create a new valid $S2$, which ensures that the timestamp $t1$ cannot be modified. After the server receives $t1$, it will check the validity of $t1$. Similarly, when the server sends information to the SN, it will generate a timestamp $t2$ and compute $S6 = h(S3, S4, S5, n2, ID_{SN}, P_{Ks}, t2)$. The adversary cannot obtain $ID_{SN}$, $n1$, and $P_{Ks}$, so it is impossible to create a new valid $S2$, which ensures that the timestamp $t2$ cannot be modified.

### 5) SENSOR NODE CAPTURE ATTACK

We assume that the adversary can capture as many SNs as possible, which means that he can obtain several tuples $(ID_{SN}, A_{SN}, B_{SN}, P_{Ks})$. At this point, we need to guarantee that the captured tuples do not make the adversary obtain the master key $K_{ser}$. Only $A_{SN}$ and $B_{SN}$ contain the information of the master key $K_{ser}$. However, $A_{SN} = r \oplus K_{ser}$ and $B_{SN} = h(r, K_{ser})$, $r$ is random and fresh, the adversary cannot obtain any $r$, so he cannot obtain $K_{ser}$ from any $A_{SN}$. The adversary also cannot obtain any parameters from $B_{SN}$, because all parameters in $B_{SN}$ are protected by the one-way function $h(.)$.

### 6) FORWARD/BACKWARD SECRECY

In our scheme, the session key $Ks = h(n1, n2, P_{Ks})$, where $n1$ and $n2$ are random numbers generated during this round of authentication, and $P_{Ks} = h(Ks', n1', n2')$, where $Ks'$, $n1'$, and $n2'$ are generated during the previous round of authentication. If the adversary compromises a session key $Ks$, he still cannot obtain the previous or future session keys. Because $n1$ and $n2$ are random and fresh in each round of authentication phase, which means that $Ks$ in each round of authentication phase is different. Even if the adversary obtains all the historical communication data in the channel and compromises the master key $K_{ser}$, which means he can obtain all $n1$ and $n2$ in each round of authentication, but he cannot obtain any previous session keys. Because $P_{Ks}$ is never transmitted in the channel. The only way to obtain the $P_{Ks}$ is to capture the sensor node. However, after capturing the sensor node, the adversary can only get the latest $P_{Ks}$, and all historical $P_{Ks}$s cannot be obtained because the previous round parameter $P_{Ks}$ is protected by a one-way function $h(.)$. Therefore, our scheme has perfect forward/backward secrecy.

### 7) SERVER SPOOFING ATTACK

The adversary needs to create a valid tuple ($S3$, $S4$, $S5$, $S6$, $t2$) before he can perform a server spoofing attack. First of all, the adversary cannot obtain $K_{ser}$, so he cannot create a valid $S3$, since $S3 = n2 \oplus h(r, K_{ser})$. Secondly, the adversary cannot obtain $ID_{SN}$, $P_{Ks}$, $n1$ and $n2$, so he cannot create a valid $S4$, $S5$ or $S6$, since $S4 = A^{+}_{SN} \oplus y \oplus n1^{*}$, $S5 = B^{+}_{SN} \oplus y$, and $S6 = h(S3, S4, S5, n2, ID_{SN}, P_{Ks}, t2)$. Therefore, the adversary cannot create a valid tuple ($S3$, $S4$, $S5$, $S6$, $t2$).

### 8) JAMMING/DESYNCHRONIZATION ATTACK

The adversary can block communication at any stage during the authentication phase, which can result in the communication parties not being able to synchronize updates. If the adversary blocks the communication during step A1 and step A2, the SN only needs to restart a new round of authentication. If the adversary blocks the communication during step A3 and step A4, there are two tuples in the server's memory, which are ($A^{+}_{SN}, X^{+}, P^{+}_{Ks}$) and ($A_{SN}, X, P_{Ks}$). The tuple ($A^{+}_{SN}, X^{+}, P^{+}_{Ks}$) is the updated authentication parameter, and the tuple ($A_{SN}, X, P_{Ks}$) is the unupdated authentication parameter. When the SN restarts a new round of authentication, the server still can use the unupdated tuple ($A_{SN}, X, P_{Ks}$) authenticated successfully. Therefore, our scheme does not have the risk of jamming/desynchronization attack.

## VI. COMPLEXITY EVALUATION AND PERFORMANCE COMPARISON

### A. STORAGE COST

In our scheme, each SN needs to store a tuple ($ID_{SN}$, $A_{SN}$, $B_{SN}$, $P_{Ks}$) and a session key $Ks$. Each server needs to store a tuple (<$A_{SN_i}$, $X_i$, $P_{Ks_i}$>, $ID_{AP_j}$, $K_{ser}$) and a session key $Ks$. We suppose that $h(.)$ uses the SHA-256 hash algorithm and the output of SHA-256 is 256 bits. Therefore, we can get $|ID_{SN}| = |A_{SN}| = |B_{SN}| = |P_{Ks}| = |X| = |K_{ser}| = |Ks| =$ 256 bits. We also suppose that $|ID_{AP}| = 32$ bits. So each SN needs to store 1280 bits, and each server needs to store (768*i*+32*j*+512) bits, where *i* is the number of registered SN and *j* is the number of registered AP. Table 5 shows the storage cost of our scheme.

**TABLE 5.** Storage cost of our scheme.

| Node | Storage cost (in bits) |
|---|---|
| SN | 1280 |
| AP | 32 |
| server | 768*i*+32*j*+512 |

**TABLE 6.** Computational cost of our scheme.

| Node | Computational cost |
|---|---|
| SN | $5t_h + 5t_{xor} \approx 5t_h$ |
| server | $7t_h + 9t_{xor} \approx 7t_h$ |

**TABLE 7.** Computation time required for different operations.

| Symbol | Description | Cost(ms) |
|---|---|---|
| $T_h$ | Time cost of one hash | 0.0052 |
| $T_{ecsm}$ | Time cost of one scalar multiplication on ECC | 0.4276 |
| $T_{bp}$ | Time cost of one bilinear pairing | 1.2828 |

**TABLE 8.** Communication cost of our scheme.

| Communication between nodes | Computational cost |
|---|---|
| SN→AP | 832 |
| AP→server | 864 |
| server→AP | 1120 |
| AP→SN | 1088 |
| total | 3904 |

### B. COMPUTATIONAL COST AND COMPUTATION TIME

We use the symbols $T_h$ and $T_{xor}$ to represent the computing time required for one hash function operation and one XOR operation. In the authentication phase, the SN performs 5 hash function operations and 5 XOR operations, and the server performs 7 hash function operations and 9 XOR operations. Compared with the hash function operation, the XOR operation requires a very short time and can be ignored. Therefore, the computation cost of the SN becomes $5t_h + 5t_{xor} \approx 5t_h$, and the computation cost of the HN becomes $7t_h + 9t_{xor} \approx 7t_h$. Table 6 shows the computation cost of our scheme.

As for the computation time, the test platform and environment are as follow: CPU: Intel(R) Core TM i7-4710HQ 2.50GHz, Memory:8G, OS:Win8 64-bit, Software: Visual C++ 2010, MIRACL C/C++ Library. Under this condition, an SHA-256 hash function operation takes approximately 0.0052ms, and one scalar multiplication on ECC with 160-bit point takes approximately 0.4276ms [23]. Since the time of a bilinear pairing is usually greater than three times the time of the scalar multiplication on ECC [24], we assume that the time of a bilinear pairing is as three times as that of the scalar multiplication on ECC, which is approximately 1.2828ms.

**TABLE 9.** Summarizes the computation time and communication cost in our scheme and other related works.

| Scheme | User/Sensor node computation cost | AP node/FA computation cost | Hub node/server/HA computation cost | Total cost | Communication cost |
|---|---|---|---|---|---|
| Ibrahim et al. [18] | $5t_h = 0.026\text{ms}$ | 0 | $8t_h = 0.0416\text{ms}$ | $13t_h = 0.0676\text{ms}$ | 3584 bits |
| Li et al. [19] | $3t_h = 0.0156\text{ms}$ | 0 | $5t_h = 0.026\text{ms}$ | $8t_h = 0.0416\text{ms}$ | 4288 bits |
| Li et al. [15] | $10t_h + 4t_{ecsm} + t_{bp} = 3.0452\text{ms}$ | $5t_h + 3t_{ecsm} + t_{bp} = 2.5916\text{ms}$ | $6t_h + 2t_{ecsm} = 0.8864\text{ms}$ | $21t_h + 9t_{ecsm} + 2t_{bp} = 6.5232\text{ms}$ | Not mentioned |
| Ours | $5t_h = 0.026\text{ms}$ | 0 | $7t_h = 0.0364\text{ms}$ | $12t_h = 0.0624\text{ms}$ | 3904 bits |

We use the symbols $T_{ecsm}$ and $T_{bp}$ to represent the computing time required for one Scalar multiplication on ECC and one Bilinear pairing. Table 7 summarizes the computation time required for different operations.

### C. COMMUNICATION COST

We assume that the timestamp is 64 bits. In the step A1, the SN sends tuple $(A_{SN}, S1, S2, t1)$ to the AP, so the tuple has $256 + 256 + 256 + 64 = 832$ bits. In the step A2, the AP sends tuple $(A_{SN}, S1, S2, t1, ID_{AP})$ to the server, so the tuple has 864 bits. In the step A3, the server sends tuple $(S3, S4, S5, S6, t2, ID_{AP})$ to the AP, so the tuple has $256 + 256 + 256 + 256 + 64 + 32 = 1120$ bits. In the step A4, the AP sends tuple $(S3, S4, S5, S6, t2)$ to the SN, so the tuple has 1088 bits. Table 8 shows the communication cost of our scheme.

### D. COMPARISONS WITH RECENT SCHEMES

The scheme of Ibrahim *et al.* [18] proposed an lightweight anonymous mutual authentication and key agreement scheme for WBAN. Their scheme has a low communication cost, but it cannot resist jamming/desynchronization attack. The scheme of Li *et al.* [19] further reduces the computational cost compared to the scheme of Ibrahim *et al.* [18], and fixes the vulnerability of jamming/desynchronization attack. But their scheme has higher communication costs, has the risk of impersonation attack after capturing a sensor node, and lacks forward secrecy. The scheme of Li *et al.* [15] uses asymmetric encryption, such as bilinear pairing, which significantly increases the computational cost while reducing security risks.

In the scheme of Ibrahim *et al.* [18], the execution time of the sensor node is $5t_h + 2t_{xor} \approx 5t_h$, and the execution time of the hub node is $8t_h + 4t_{xor} \approx 8t_h$. The total communication cost of their scheme in authentication phase is 3584 bits. In the scheme of Li *et al.* [19], the execution time of the sensor node is $3t_h + 6t_{xor} \approx 3t_h$, and the execution time of the hub node is $5t_h + 11t_{xor} \approx 5t_h$. The total communication cost of their scheme in authentication phase is 4288 bits. In the scheme of Li *et al.* [15], the execution time of the user is $10t_h + 4t_{ecsm} + t_{bp} + 2t_{xor} \approx 10t_h + 4t_{ecsm} + t_{bp} = 319t_h$, the foreign agent (FA) $5t_h + 3t_{ecsm} + t_{bp} = 241.5t_h$, and the home agent (HA) $6t_h + 2t_{ecsm} + t_{xor} \approx 6t_h + 2t_{ecsm} = 151t_h$. Table 9 summarizes the computation time and communication cost in our scheme and other related works.

As for security, the scheme of Ibrahim *et al.* [18] cannot resist jamming/desynchronization attack. Because during the authentication phase of their scheme, the parameters for authentication need to be updated sequentially between the sensor node and the server. Once the adversary blocks the network at a certain stage, the authentication parameters in the hub node may be successfully updated, and the authentication parameters in the sensor node are not, which will result in the sensor node unable to successfully complete the next round of authentication unless it is re-registered. Their scheme also lacks forward secrecy. In the scheme of Li *et al.* [19], as long as the adversary captures a sensor node and obtains all the data in its memory, he can perform impersonation attack by forging the identity of the sensor node. Because the data related to the identity of the sensor node is not stored in the memory of the hub node, this makes it impossible for the hub node to verify it. In addition, if the adversary also obtained all historical communication data transmitted on the channel, he can obtain all the previous session keys between the sensor node and the hub node. Therefore, the scheme of Li *et al.* [19] lacks forward secrecy, and has the risk of sensor node impersonation attack after any sensor node has been captured. The scheme of Li *et al.* [15] would be vulnerable to replay attack. Because their scheme only uses random numbers to defend against replay attacks, and does not protect the random numbers. Table 10 shows the security properties comparison. In conclusion, although our scheme is higher in computational cost than the scheme of Li *et al.* [19], our scheme has lower security risk. Compared with the scheme of Ibrahim *et al.* [18] and the ECC-based scheme of Li *et al.* [15], our scheme guarantees the security while reducing the computational cost.

**TABLE 10.** Security properties comparison.

| Functionality | $A1$ | $A2$ | $A3$ | $A4$ | $A5$ |
|---|---|---|---|---|---|
| Ibrahim et al. [18] | × | × | √ | √ | √ |
| Li et al. [19] | × | √ | × | √ | √ |
| Li et al. [15] | √ | √ | √ | × | √ |
| Ours | √ | √ | √ | √ | √ |

$A1$: Forward/Backward security, $A2$: Jamming/desynchronization attack, $A3$: Sensor node impersonation attack, $A4$: Replay attack, $A5$: Hub/server node impersonation attack

## VII. CONCLUSION

In this paper, we propose a lightweight and anonymous mutual authentication and key agreement scheme for WBAN, it only needs to perform hash function operations and

XOR operations. With this scheme, forward secrecy can be guaranteed without using asymmetric encryption. Even after the adversary captures the sensor node, compromises the master key $K_{ser}$, and obtains all historical communication data in the channel, he still cannot obtain any previous session keys. We also used the automatic security verification tool ProVerif to verify the security of our scheme and discussed other common attacks. The experimental results and theoretical analysis indicate that our scheme significantly reduces the computational cost compared with the schemes using asymmetric encryption and that it has a lower security risk compared with the lightweight schemes. In addition, our scheme is not limited to WBAN, we will consider applying the proposed method of guaranteeing forward secrecy in this paper to more scenarios that require lightweight authentication schemes, such as Internet of Vehicles and mobile service computing [25]–[27].

## REFERENCES

[1] S. Deng, Z. Xiang, J. Yin, J. Taheri, and A. Y. Zomaya, "Composition-driven IoT service provisioning in distributed edges," *IEEE Access*, vol. 6, pp. 54258–54269, 2018.

[2] S. Deng *et al.*, "Toward risk reduction for mobile service composition," *IEEE Trans. Cybern.*, vol. 46, no. 8, pp. 1807–1816, Aug. 2016.

[3] S. Deng, L. Huang, D. Hu, J. L. Zhao, and Z. Wu, "Mobility-enabled service selection for composite services," *IEEE Trans. Services Comput.*, vol. 9, no. 3, pp. 394–407, May/Jun. 2016.

[4] H. Cao, V. Leung, C. Chow, and H. Chan, "Enabling technologies for wireless body area networks: A survey and outlook," *IEEE Commun. Mag.*, vol. 47, no. 12, pp. 84–93, Dec. 2009.

[5] K. Akkaya, M. Younis, and M. Youssef, "Efficient aggregation of delay-constrained data in wireless sensor networks," in *Proc. Int. Conf. Comput. Syst. Appl. (AICCSA)*, Jan. 2005, pp. 904–909.

[6] U. Varshney, "Pervasive healthcare: Applications, challenges and wireless solutions," *Commun. Assoc. Inf. Syst.*, vol. 16, no. 1, pp. 57–72, 2005.

[7] R. Canetti, S. Halevi, and J. Katz, "A forward-secure public-key encryption scheme," in *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), vol. 2656. Berlin, Germany: Springer, 2003.

[8] A. M. Koya and P. P. Deepthi, "Anonymous hybrid mutual authentication and key agreement scheme for wireless body area network," *Comput. Netw.*, vol. 140, pp. 138–151, Jul. 2018.

[9] L. Shi, J. Yuan, S. Yu, and M. Li, "ASK-BAN: Authenticated secret key extraction utilizing channel characteristics for body area networks," in *Proc. ACM Conf. Secur. Privacy Wireless Mobile Netw. (WiSec)*, 2013, pp. 155–166.

[10] J. M. Hamamreh and H. Arslan, "Secure orthogonal transform division multiplexing (OTDM) waveform for 5G and beyond," *IEEE Commun. Lett.*, vol. 21, no. 5, pp. 1191–1194, May 2017.

[11] M. Turkanović, B. Brumen, and M. Hölbl, "A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the Internet of Things notion," *Ad Hoc Netw.*, vol. 20, pp. 96–112, Sep. 2014.

[12] C.-C. Chang and H.-D. Le, "A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 357–366, Jan. 2016.

[13] X. Li, J. Peng, J. Niu, F. Wu, J. Liao, and K. K. R. Choo, "A robust and energy efficient authentication protocol for industrial Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1606–1615, Jun. 2018.

[14] P. Gope and T. Hwang, "An efficient mutual authentication and key agreement scheme preserving strong anonymity of the mobile user in global mobility networks," *J. Netw. Comput. Appl.*, vol. 62, pp. 1–8, Feb. 2016.

[15] X. Li, J. Niu, S. Kumari, F. Wu, and K.-K. R. Choo, "A robust biometrics based three-factor authentication scheme for global mobility networks in smart city," *Future Gener. Comput. Syst.*, vol. 83, pp. 607–618, Jun. 2018.

[16] S. Janbabaei, H. Gharaee, and N. Mohammadzadeh, "Lightweight, anonymous and mutual authentication in IoT infrastructure," in *Proc. 8th Int. Symp. Telecommun. (IST)*, Sep. 2016, pp. 162–166.

[17] D.-H. Hwang, J.-M. Shin, and Y.-H. Choi, "Authentication protocol for wearable devices using mobile authentication proxy," in *Proc. 10th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2018, pp. 700–702.

[18] M. H. Ibrahim, S. Kumari, A. K. Das, M. Wazid, and V. Odelu, "Secure anonymous mutual authentication for star two-tier wireless body area networks," *Comput. Methods Programs Biomed.*, vol. 135, pp. 37–50, Oct. 2016.

[19] X. Li, M. H. Ibrahim, S. Kumari, A. K. Sangaiah, V. Gupta, and K.-K. R. Choo, "Anonymous mutual authentication and key agreement scheme for wearable sensors in wireless body area networks," *Comput. Netw.*, vol. 129, pp. 429–443, Dec. 2017.

[20] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. 29, no. 2, pp. 198–208, Mar. 1983.

[21] B. Blanchet, B. Smyth, and V. Cheval. (2015). *ProVerif 1.92: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial*. [Online]. Available: https://bensmyth.com/files/ProVerif-manual-version-1.92.pdf

[22] Q. Jiang, Z. Chen, B. Li, J. Shen, L. Yang, and J. Ma, "Security analysis and improvement of bio-hashing based three-factor authentication scheme for telecare medical information systems," *J. Ambient Intell. Humanized Comput.*, vol. 9, no. 4, pp. 1061–1073, 2018.

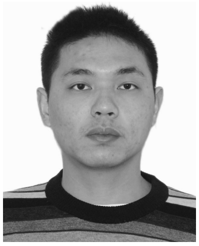
[23] F. Wu *et al.*, "A novel and provably secure authentication and key agreement scheme with user anonymity for global mobility networks," *Secur. Commun. Netw.*, vol. 9, no. 16, pp. 3527–3542, 2016.

[24] D. He, C. Chen, S. Chan, and J. Bu, "Secure and efficient handover authentication based on bilinear pairing functions," *IEEE Trans. Wireless Commun.*, vol. 11, no. 1, pp. 48–53, Jan. 2012.

[25] S. Deng, H. Wu, W. Tan, Z. Xiang, and Z. Wu, "Mobile service selection for composition: An energy consumption perspective," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 3, pp. 1478–1490, Jul. 2017.

[26] S. Deng, L. Huang, J. Taheri, J. Yin, M. Zhou, and A. Y. Zomaya, "Mobility-aware service composition in mobile communities," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 3, pp. 555–568, Mar. 2017.

[27] W. Liang, J. Long, T.-H. Weng, X. Chen, K.-C. Li, and A. Y. Zomaya, "TBRS: A trust based recommendation scheme for vehicular CPS network," *Future Gener. Comput. Syst.*, vol. 92, pp. 383–398, Mar. 2019.



**ZISANG XU** was born in 1991. He received the bachelor's degree in electronic information engineering from the Wuhan University of Engineering, in 2011, and the master's degree in computer science and technology from the Hunan University of Science and Technology, in 2014. He is currently pursuing the Ph.D. degree in computer science and technology with Hunan University. He participated in the National Natural Science Foundation of China, CPS Instantiation—Research on the Smart Inspection Robot of Catenary. His research interests include embedded systems, information security, and cryptography.

**CHENG XU** was born in 1962. He received the Ph.D. degree in computer science and engineering from the Wuhan University of Technology, in 2006. He is currently a Professor and a Ph.D. Supervisor with the College of Computer Science and Electronic Engineering, Hunan University. He has published 28 papers and hosted several national and provincial nature fund projects. His main research interests include embedded systems, digital video processing, and automated test and control. He is a member of the China Computer Federation.

**WEI LIANG** received the Ph.D. degree from Hunan University, in 2013. He was a Postdoctoral Scholar with the Department of Computer Science and Engineering, Lehigh University, USA, from 2014 to 2016. He has been a Guest Researcher with the State Key Laboratory of Information Security, Sun Yat-sen University. He is currently a Professor with the School of Software, Xiamen University of Technology. He has published more than 110 journal/conference papers in journals, such as the IEEE Transactions on Computational Biology and Bioinformatics, *Future Generation Computer Systems*, *Wireless Personal Communications*, *Microprocessors and Microsystems*, the *International Journal of Communication Systems*, and the *Journal of Sensors*, *Security and Communication Networks*, and *Nonlinear Dynamics*. His research interests include networks security protection, embedded systems and hardware/IP protection, fog computing, and security management in WSN.

**JIANBO XU** received the M.S. degree from the Department of Computer Science and Technology, National University of Defense Technology, China, in 1994, and the Ph.D. degree from the College of Computer Science and Electronic Engineering, Hunan University, China, in 2003. Since 2003, he has been a Professor with the School of Computer science and Engineering, Hunan University of Science and Technology. His research interests include network security and distributed computing.

**HAIXIAN CHEN** received the bachelor's degree in Internet of Things engineering from Hunan University, in 2016, where she is currently pursuing the master's degree in computer science and technology. Her main research directions are embedded systems, information security, and cryptography. She has received several national and provincial awards.