An Anonymous Authentication Protocol Based on Cloud for Telemedical Systems

Wenmin Li,1 Shuo Zhang1, Qi Su,2 Qiaoyan Wen,1 and Yang Chen1

1State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
2State Key Laboratory of Cryptology, Beijing 100878, China

Correspondence should be addressed to Shuo Zhang; zsl990@vip.qq.com

Received 10 April 2018; Accepted 2 August 2018; Published 2 September 2018

Academic Editor: Joseph Liu

Copyright © 2018 Wenmin Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Telecare medical information systems (TMIS) enable patients to access healthcare delivery services conveniently. With the explosive development occurring in cloud computing and services, storage of personal medical and health information outsourcing to cloud infrastructure has been a potential alternative. However, this has entailed many considerable security and privacy issues. In order to address the security loopholes, we propose a promising solution satisfying the requirements of cloud computing scenarios for telemedical systems. The proposed scheme could provide both data confidentiality and message authenticity while preserving anonymity. Furthermore, the formal security proof demonstrates that the proposed scheme is resistant to various attacks. The performance comparisons show the proposal's workability and it is well suited to adoption in telemedical services.

1. Introduction

With the explosion of cloud computing and services, there has been a growing trend to use the cloud for large-scale data storage and management. It is a new style of computing that offers dynamically scalable network services to external customers delivered over the Internet. Cloud computing provides a powerful underlying architecture for telemedicine, which is an emerging treatment mode for delivering appropriate healthcare services remotely. It facilitates medical practitioners and patients to establish communication over public networks and patients can acquire the medical services via electronic networks conveniently. This will significantly lower the social and economic expenses, while enhancing the medical quality and efficiency.

Cloud computing introduces a new way for medical systems to store and manage medical data, which is complex task. As wearable devices are becoming more and more powerful, patients can obtain their health information timely. They also could upload and access their medical records to the cloud through mobile devices. This can help medical institutions to quickly obtain patients' physical condition in urgent cases for proper medical diagnosis and treatment process. Any delay in the access to medical record at the time of emergency would cause severe errors, which profoundly affects patient's therapeutic process. In the cloud based telecare medical information systems, the cloud database is responsible for storing patients' critical medical data and updates it as the medical treatment availed by the patient. However, the storage of patients' electronic medical records such as personal information, medical records, and physiological parameters in the medical server may result in the exposure of patients' privacy. Cloud computing offers expansively developing prospects of new and better models of healthcare; it also raises some security issues due to new potential ways for data theft. And hence, safeguarding security and patients’ privacy in cloud based telecare medical information systems are very significant. Authentication mechanism is a prerequisite to verify the legality of all participants and tackle the illegal access in distributed systems, such as wireless interface systems [1, 2], multiple server architecture based systems [3], smart card based system [4], and mobile radio systems [5, 6]. Furthermore, the anonymous authentication could protect users anonymity and prevent the disclosure of private information [7, 8]. Therefore, a secure authentication protocol is a proper
solution to provide security and privacy for TMIS [9–11].
Hitherto, authentication protocol for integration telemedical
systems in cloud computing environment recently has drawn
significant attention from academia [12–20].
In 2012, Padhy et al. [12] introduced a cloud based model
for rural healthcare systems. In 2013, Banerjee et al. [13]
presented a new architecture for cloud based healthcare
application to serve patients in emergency. Nevertheless, their
scheme is unable to offer confidentiality of transmitted data.
One year later, Chen et al. [14] proposed a medical data
exchange protocol in cloud computing environment. In their
scheme, patients and doctors could be convenient to access
medical resources outsourced in the cloud. Unfortunately,
their scheme could not resist impersonation attacks or pro-
vide patient anonymity. To fix the defects, a modification
was developed in the same year [15]. In 2016, Chiou et al.
[16] showed that their scheme still lacks privacy protection
and message authentication. Then, the authors proposed a
new privacy authentication scheme based on cloud for TMIS
which provided a “real” and complete telemedicine system.
However, in 2007, Mohit et al. [17], Cheng et al. [20], and Li et
al. [18] identified Chiou et al’s protocol that failed to preserve
patients’ privacy and forward security and suffers from
mobile device stolen attack, respectively. Meanwhile, Mohit
et al. [17] and Cheng et al. [20] both presented an improved
mechanism for cloud-assisted medical care systems. Recently,
Li et al. [19] pointed out that Mohit et al’s proposal also
was susceptible to health report revelation and inspection
report forgery attacks. In Cheng et al’s scheme [20], the inputs
of bilinear maps are generators in the corresponding cyclic
groups, rather than random numbers of integer field \( \mathbb{Z}_p \).
This will bring about errors in the authentication process.

In this paper, we design a telemedical information model
based on cloud authentication which allows patients to
remotely access medical services with privacy. Further, we
discuss its security and prove that it can withstand various
attacks. Compared with the state of the art, our scheme
provides formal security proofs and achieves better efficiency
in terms of computation cost. Performance and functionality
analysis shows that it is more secure and practical for cloud
based telemedicine system.

The remaining of this paper is organized as follows.
Section 2 describes our robust cloud based authentication
scheme for TMIS, together with formally proving its security
in Section 3. Subsequently, we compare the performance
with the previous schemes in Section 4. Finally, we draw the
conclusions in Section 5.

2. The Proposed Scheme

In this section, we present an anonymous authentication
scheme on the basis of cloud for medical environment. There
are five participants in our scheme: including patients \( P \),
healthcare center \( H \), doctors \( D \), cloud \( C \), and sensors \( S \).
Healthcare center is trusted medical center. The cloud servers
possess the jurisdiction to store patients’ medical data which
can be accessed by patients and doctors remotely. Sensors
can collect and measure the patient’s health information
timely. In Figure 1, we depict the structure of the cloud based
authentication system for TMIS simply.

Our scheme consists of four phases which are described as
follows. In order to initialize this protocol, the key generation
center (KGC) chooses a multiplication cyclic group \( G \) and
a generator \( g \in G \) with order \( p \), where \( p \) is a large prime
number. Then KGC selects random numbers \( k_x \in \mathbb{Z}_p \) \( (x =\{H, P, D, C\}) \) and computes \( PK_x = g^{k_x} \mod p \). Finally, KGC
issues the public key and secret key pairs \((PK_x, k_x)\) to the
participants.

We list the used notations of the proposal as follows.
<table>
<thead>
<tr>
<th>$H$</th>
<th>$C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generates the inspection $m_H = (ID_P, Data_H, T_H)$,</td>
<td>Generates random number $t_H$ and timestamp $T_H^1$,</td>
</tr>
<tr>
<td>Generates random number $t_H$, and computes $T_H^1$,</td>
<td>Computes $K_1 = g^{t_H}$, $s_1 = PK_H C = g^{s_{1}k_{H}}$,</td>
</tr>
<tr>
<td>Computes $K_1 = g^{t_H}$, $s_1 = PK_H C = g^{s_{1}k_{H}}$,</td>
<td>$s_1 = h \left( PK_H^C</td>
</tr>
<tr>
<td>$C_i = E_{s_{1}}(s_1)$,</td>
<td>$C_i = E_{s_{1}}(s_1)$,</td>
</tr>
<tr>
<td>Checks the validity of $T_H^1$.</td>
<td>Checks the validity of $T_H^1$.</td>
</tr>
<tr>
<td>Computes $s_1' = K_1^{s_1} = g^{s_{1}k_{H}}$,</td>
<td>Computes $s_1' = K_1^{s_1} = g^{s_{1}k_{H}}$,</td>
</tr>
<tr>
<td>Decrypts $C_i$ with $s_1'$ to obtain $s_1$.</td>
<td>Decrypts $C_i$ with $s_1'$ to obtain $s_1$.</td>
</tr>
<tr>
<td>Checks $s_1' = h \left( PK_H^C</td>
<td></td>
</tr>
<tr>
<td>Signs the medical report $Sig_H = S_{pk_H}(m_H)$,</td>
<td>Signs the medical report $Sig_H = S_{pk_H}(m_H)$,</td>
</tr>
<tr>
<td>Computes $C_2 = E_{pk_H^C}(m_H, Sig_H)$,</td>
<td>Computes $C_2 = E_{pk_H^C}(m_H, Sig_H)$,</td>
</tr>
<tr>
<td>$s_4 = E_{sk_H}(ID_P, C_1, s_1)$,</td>
<td>$s_4 = E_{sk_H}(ID_P, C_1, s_1)$,</td>
</tr>
<tr>
<td>Checks whether $ID_P$ is a new identity,</td>
<td>Checks whether $ID_P$ is a new identity,</td>
</tr>
<tr>
<td>Stores $ID_P, C_2$ in $P$'s storage space.</td>
<td>Stores $ID_P, C_2$ in $P$'s storage space.</td>
</tr>
</tbody>
</table>

**Figure 2: Healthcare center uploading phase.**

(i) $ID_X$: the identity of $x$
(ii) $m_X$: the health report of $x$
(iii) $Data_X$: the health data of $x$
(iv) $sk_{xy}$: the session key shared between $x$ and $y$
(v) $PK_x$: the public key of $x$
(vi) $k_x$: the secret key of $x$
(vii) $Sig_x$: signature signed by $x$
(viii) $T^i_x$: the $i$th timestamp generated by $x$
(ix) $G$: a multiplicative cyclic group of prime order $q$
(x) $g$: the generator of $G$
(xi) $h(\cdot)$: a one-way hash function
(xii) $\oplus$: exclusive-OR operation
(xiii) $\|$: string concatenation operation
(xiv) $E_{key}(M)$: encrypting $M$ using symmetric encryption with key $key$
(xv) $D_{key}(C)$: decrypting the ciphertext $C$ with key $key$
(xvi) $S_{key}(M)$: signing $M$ with secret key $key$
(xvii) $V_{key}(Sig_X)$: verifying the signature $Sig_X$

2.1. Healthcare Center Uploading Phase. As shown in steps 1.1 and 1.2 in Figure 1, patient $P$ makes a health inspection in the healthcare center $H$ and $H$ uploads the generated inspection record to the cloud server $C$. In Figure 2, we will further describe the authentication process of the phase.

**Step 1.** After generating the inspection report, $H$ selects a random number $t_H$ and computes $K_1 = g^{t_H}$, $s_1 = PK_H C = g^{s_{1}k_{H}}$, $s_2 = h \left( PK_H^C || T_H^1 \right) = h \left( g^{s_{1}k_{H}} || T_H^1 \right)$, $C_i =$
where $T^1_H$ is the current timestamp. After that, it sends
\{ID_{H}, C_1, T^1_H, K_1\} to the cloud $C$.

Step 2. On receiving \{ID_{H}, C_1, T^1_H, K_1\} from $H$, $C$ checks the
validity of $T^1_H$. If it is invalid, $C$ aborts the received messages;
otherwise, it computes $s'_1 = K^1_{KC} = g^{s_1K_C}$ and decrypts $C_1$ to recover the values $s_2$. Subsequently, $C$ calculates $s'_2 = h(K^2_{KC} || T^2_H)$ and compares it with the decrypted $s_2$. If the equivalence holds, the legitimacy of $H$ is assured. Then $C$ generates a random number $t_{C}$ and acquires the
timestamp $T^1_C$ to compute $K_2 = g^{t_{C}}$, $sk_{IKC} = K^1_{t_{C}} = g^{t_{C}K_C}$, and $s_3 = h(PK^2_{HK} || sk_{IKC} || T^1_C) = h(g^{s_1K_C} || sk_{IKC} || T^1_C)$. Finally, $C$
transmits the authentication message \{$K_2, s_3, T^1_C$\} to $H$.

Step 3. Upon receiving the reply message, $H$ checks the
validity of $T^1_C$. If $T^1_C \neq T^1_H \geq \Delta T$, $H$ terminates this phase;
otherwise, it computes $sk_{IKC} = K^2_{t_{C}} = g^{t_{C}K_C}$, and $s'_3 = h(PK^2_{HK} || sk_{IKC} || T^1_C)$ and compares $s'_3$
with the received $s_3$. If they are not equal, the uploading phase
is given up by $H$; else, $H$ signs $P$’s medical report
$m_{H} = (ID_{p}, Date_{H}, T_{H})$ with its secret key: $Sig_{H} = S_{sk_{H}}(m_{H})$.
Note that $T_{H}$ is the current timestamp when $P$ makes health
inspection. After that, $H$ calculates $C_3 = E_{PK^2_{HK}}(m_{H}, Sig_{H}) = E_{g^{s_1K_C}}(m_{H}, Sig_{H})$, $s_4 = E_{sk_{HC}}(ID_{P}, C_2, s_3)$ and uploads $s_4$ to the cloud.

Step 4. On receiving $s_4$, $C$ decrypts it with $sk_{HC}$ to recover
$ID_{P}, C_2, s_3$ and verifies whether $s_4$ is equal to $s_3$ or not. If it is
true, the healthcare center is authentic. After that, it verifies
whether $P$ is a new user or not. If $P$ is a new user, $C$ stores
\{ID_{P}, C_2\} in a new storage space; else, it stores $C_2$ in $P$’s
database.

2.2. Patient Uploading Phase. As shown in steps 2.1 and
2.2 in Figure 1, patient $P$ collects health information
$m_{K}$ measured by body sensors $S$ and he could upload the health
data to the cloud. In Figure 3, we will depict the detailed
process.

Step 1. When $P$’s mobile phone collects the measured
information, then it generates the timestamp $T^1_P$ and a random
number $a_{P}$ to compute $K_3 = g^{a_{P}}$, $s_5 = PK^3_{CP} = g^{a_{P}K_C}$, $s_6 = h(PK^2_{HK} || T^1_P) = h(g^{s_1K_C} || T^1_P)$, $C_3 = E_{s_4}(ID_{P}, s_5)$. Subsequently, $P$ transmits
$\{C_3, T^1_P, K_3\}$ to $C$.

Step 2. After receiving the messages, $C$ verifies the freshness of
$T^1_P$ by checking whether $T^1_P - T^1_H \leq \Delta T$ or not. If it
is valid, $C$ decrypts $C_3$ to obtain the values $ID_{P}$ and $s_{4}$ with
the computed $s'_5 = K^3_{KC} = g^{s_4K_C}$. After that, $C$ calculates
$s'_6 = h(PK^3_{CP} || T^1_P)$ and verifies whether the equation $s'_6 = s_6$
holds. If it does, $P$ is legitimate user. Then $C$ selects random
numbers $a_{C}$ and computes $K_4 = g^{a_{C}}$, $sk_{PC} = K^3_{a_{C}} = g^{a_{C}K_C}$, and $s_7 = h(PK^3_{PC} || sk_{PC} || T^1_C) = h(g^{s_4K_C} || sk_{PC} || T^1_C)$, where $T^1_C$ is the acquired timestamp. Finally, $C$ transmits the
\{$K_4, s_7, C_3, T^1_C$\} to patient $P$.

Step 3. On receiving response, $P$ checks the validity of $T^1_C$.
If $T^1_C$ is invalid, $P$ terminates the procedure. If $T^1_C$ is fresh, $P$
computes $sk_{PC} = K^3_{a_{C}} = g^{a_{C}K_C}$, $s'_7 = h(PK^3_{PC} || sk_{PC} || T^1_C) = h(g^{s_4K_C} || sk_{PC} || T^1_C)$ and verifies $s_7$ is valid by checking whether $s'_7 = s_7$ holds. If so, $P$ decrypts $C_2$ with the computed
$PK^3_{HK}$ to recover $m_{H} = (ID_{p}, Date_{H}, T_{H})$ andSig_{H}. Subsequently,
he/she verifies the validity of $H$’s signature $Sig_{H}$. If $Sig_{H}$ is valid, $P$ chooses a random number $sn$ and computes
$C_4 = E_{PK^3_{HK}}(m_{H}, sn, Sig_{H}) = E_{g^{a_{C}K_C}}(m_{H}, sn, Sig_{H})$, where
$m_{H} = (ID_{p}, Date_{H}, T_{H})$ is the collected measured data. Note
that $T_{H}$ is the current timestamp when the body sensors
monitor $P$’s physical condition. Then $P$ calculates $s_8 =
E_{sk_{HC}}(ID_{P}, C_4, s_7)$ and uploads $s_8$ to the $C$.

Step 4. On receiving the reply message, $C$ decrypts $s_8$ with
$sk_{HC}$ and obtains $ID_{P}, C_4, s_7$. After that, the cloud server
verifies $P$’s validity by checking whether $s'_7$ equals to $s_7$ or not.
If so, $C$ stores $C_4$ in $P$’s storage space to replace $C_2$; otherwise, it
resumes the procedure.

2.3. Treatment Phase. As shown in steps 3.1 and 3.2 in
Figure 1, $D$ is appointed by $P$ and obtains $P$’s identity $ID_{P}$
and appointment sequence value $sn$. Subsequently, $D$ can down-
load $P$’s inspection report and measured health information
from $C$, and he/she also can upload the diagnosing records
with his/her signature to $C$. The details of the execution steps
are further illustrated in Figure 4.

Step 1. $D$ selects a random number $b_{D}$ and computes $K_5 =
g^{b_{D}}, s_9 = PK^{b_{D}}_{CD} = g^{b_{D}K_C}, s_{10} = h(PK^{b_{D}}_{CD} || T^1_D) =
h(g^{b_{D}K_C} || T^1_D), C_5 = E_{s_{10}}(s_{10})$, where $T^1_D$ is the acquired
current timestamp. Then $D$ transmits \{ID_{D}, C_5, T^1_D, K_5\} to the cloud $C$.

Step 2. After receiving messages from $D$, $C$ checks $T^1_D - T^1_C \leq \Delta T$.
If it is invalid, $C$ terminates the phase; otherwise, it computes
$s'_9 = K^5_{KC} = g^{s_9K_C}$ and decrypts $C_5$ to obtain the values
$s_{10}$. Later, $C$ verifies $s'_9 = h(PK^{b_{D}}_{CD} || T^1_D)$ with the
decrypted $s_{10}$ to confirm the legitimacy of $D$. If they are equal, $C$ generates random numbers $b_{C}$ and the timestamp
$T^1_C$ and computes $K_6 = g^{b_{C}}, sk_{CD} = K^5_{b_{C}} = g^{b_{C}K_C}, s_{11} = h(PK^{b_{C}}_{CD} || sk_{CD} || T^1_C) = h(g^{b_{D}K_C} || sk_{CD} || T^1_C)$. Finally, $C$
sends $\{K_{6}, s_{11}, C_5, T^1_C\}$ to $D$.

Step 3. On receiving \{$K_{6}, s_{11}, C_5, T^1_C$\}, $D$ checks the freshness of
$T^1_C$. If so, $D$ computes $sk_{CD} = K^5_{b_{C}} = g^{b_{C}K_C}, s'_{11} = h(PK^{b_{C}}_{CD} || sk_{CD} || T^1_C)$ and compares $s'_{11}$ with the
received $s_{11}$ to assure $C$’s authenticity. If they are not equal,
the phase is terminated by himself/herself; otherwise, $D$ uses
the appointment sequence number $sn$ to compute $PK^{s_{11}K_C}$ and
decrypts $C_4$ with it to recover $m_{H}, m_{S}, Sig_{H}$. Subsequently, $D$
verifies whether the signature $Sig_{H}$ is valid or not by checking
$m_{H} = V_{PK^{s_{11}K_C}}(Sig_{H})$. If it is valid, $D$ diagnoses $P$’s symptom
on the basis of \{$m_{H}, m_{S}$\} and generates the diagnostic records
$m_{D} = (ID_{p}, Date_{D}, T_{D})$, where $T_{D}$ is the timestamp when
the doctor generates $m_{D}$. After that, $D$ uses his/her private
key $k_D$ to sign $m_D; \text{Sig}_D = S_k_D(m_D)$. Then, $D$ calculates $C_6 = E_{PK^C_p}(m_1, m_3, m_5, \text{Sig}_D)$, $s_{12} = E_{sk_CD}(ID_D, C_6, s_{11})$ and sends $s_{12}$ to the cloud.

Step 4. Upon receiving $s_{12}$, $C$ decrypts it with $sk_{CD}$ and obtains $ID_D, C_6, s_{11}$. Later, it checks $s_{11} = s_{11}$. If the equation holds, the validity of $D$ is confirmed; otherwise, this phase fails. After that, $C$ replaces $C_4$ with $C_6$ and stores it in $P$’s storage space.

2.4. Checking Report Phase. As shown in step 4.1 in Figure 1, patient $P$ can access the cloud to obtain the medical record via the mobile phone. In Figure 5, we depict the detailed process of the phase.

Step 1. $P$ generates the timestamp $T_p^2$ and a random number $v_p$ to compute $K_7 = g^{v_p}$, $s_{13} = PK_C^p = g^{sk_{CP} v_p}$, $s_{14} = h(PK_C^p || T_p^2)$, $s_{15} = h(g^{sk_{CP} v_p} || T_p^2)$. Then, $P$ transmits the request $\{C_7, T_p^2, K_7\}$ to the cloud $C$.

Step 2. $C$ verifies the freshness of $T_p^2$ after receiving the request from $P$. If so, $C$ decrypts $C_7$ with computed $s'_{13} = K_7^{v_p}$ and obtains the $ID_D, s'_{11}$. Subsequently, $C$ calculates $s'_{14} = h(PK_K^C || T_p^2)$ and verifies $s'_{14} = s_{14}$. If they are equal, $C$ computes $K_8 = g^{sk_{CP} v_p}$ and $s_{15} = h(PK_K^C || sk_{PC'} || T_C^2) = h(g^{sk_{CP'} || sk_{PC'}} || T_C^2)$, where $v_p$ and $T_C^2$ are generated random value and the acquired timestamp, respectively. After that, $C$ sends the $\{K_8, s_{15}, C_6, T_C^2\}$ to $P$.

Step 3. On receiving response $\{K_8, s_{15}, C_6, T_C^2\}$, $P$ checks the validity of $T_C^2$. If $T_C^3 - T_C^2 \geq \Delta T$, $P$ aborts the session; otherwise, he/she calculates $sk_{PC'} = K_8^{v_p}$, $s'_{15} = h(PK_K^C || T_C^2)$, $C_7 = E_{sk_{CP}}(ID_D, s_{14})$. Then, $P$ transmits the request $\{C_7, T_p^2, K_7\}$ to the cloud $C$.

---

<table>
<thead>
<tr>
<th>$P$</th>
<th>$C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obtains the measured information $m_p = {ID_p, Data_p, T_p}$, generates random number $\alpha_p$ and timestamp $T_p^1$, computes $K_3 = g^{\alpha_p}$, $s_1 = PK_C^p = g^{sk_{CP} \alpha_p}$, $s_6 = h(PK_C^p</td>
<td></td>
</tr>
<tr>
<td>${C_1, T_p^1, K_3}$</td>
<td></td>
</tr>
<tr>
<td>Checks the validity of $T_p^1$, computes $s_{10} = K_3^{\alpha_p}$, decrypts $C_{12}$ with computed $s'<em>{10} = K_3^{v_p}$, computes $s'</em>{12} = E_{sk_{CD}}(C_1)$. Verifies $m_1 = V_{sk_{CD}}(\text{Sig}<em>D)$. Generates random numbers $s</em>{11}$, computes $C_4 = E_{PK^C_p}(m_1, m_3, m_5, \text{Sig}<em>D)$, $s_8 = E</em>{sk_CD}(ID_D, C_4, s_{11})$.</td>
<td></td>
</tr>
<tr>
<td>${K_4, s_7, C_2, T_C^1}$</td>
<td></td>
</tr>
<tr>
<td>Decrypts $s_8$ with $sk_{PC'}$ to obtain $s'_8$, checks $s'_8 = s_8$, stores $C_4$ in $P$’s storage space to replace $C_2$.</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 3:** Patient uploading phase.
**Figure 4: Treatment phase.**

\( s_{10} = h(K^b_P || T^D_D) = h(g^{\alpha_b} || T^D_D) \)

\( C_i = E_k(s_{10}) \)

\[ \{ID, C_i, T^D_D, K_s \} \]

Checks the validity of \( T^D_D \),

Computes \( s_i' = K^b_P = g^{\alpha_b} \),

Decrypts \( C_i \) with \( s_i' \) to obtain \( s_{10} \),

Checks \( s_{10} \)? = \( h(K^b_P || T^D_D) \).

Generates random number \( b_P \) and timestamp \( T^D_D \),

Computes \( K_e = g^{\alpha_e}, sk_{CD} = K^b_P = g^{\alpha_b}, \)

\( s_{11} = h(K^b_P || sk_{CD} || T^D_D) \).

\[ \{K_e, s_{11}, C_e, T^D_D \} \]

Checks the validity of \( T^D_D \),

Computes \( s_{CD} = K^b_P = g^{\alpha_b} \),

Checks \( s_i ? = h(K^b_P || sk_{CD} || T^D_D) \).

Computes \( PK^P_{PC} \) and decrypts \( C_e \) to recover \( m_H, m_S, Sig_H \).

Verifies \( m_j ? = V_{PK^P_{PC}}(Sig_H) \).

Generates diagnosis record \( m_D \),

Signs the diagnosis record \( Sig_D = S_{sk}(m_D) \).

Computes \( C_e = E_{sk_{PC}}(ID_D, C_e, s_{11}) \),

\( s_{12} = E_{sk_{PC}}(ID_D, C_e, s_{11}) \).

\[ \{s_{12} \} \]

Decrypts \( s_{12} \) with \( sk_{PC} \) to obtain \( s'_{12} \),

Checks \( s'_{11} ? = s_{11} \),

Stores \( C_e \) in \( P \)’s storage space to replace \( C_6 \).

**3. Security Proof**

In this section, we will prove our scheme to be secure in standard model. We reduce the security of our authentication scheme to cryptography basic elements [21, 22]. At first, in order to achieve this goal, we will introduce the definitions of security, a structured security model, and the basic assumptions. Then we use all of them to prove the result.

**Definition 1 (semantic security).** For arbitrary security parameter \( N \), if and only if any polynomial time adversary has a negligible advantage against the scheme, we say the scheme has semantic security.

The definitions are inherited and modified from the methodology of Bellare, Pointcheval, and Rogaway [23] and the game-based structure [24] is used to prove this scheme achieving semantic security.
Security Model. In the security model, the adversary plays a game with an oracle. The oracle runs the real protocol and answers the queries of adversary to simulate the real interaction of participants. After a range of queries, the adversary gets different capabilities. When the adversary finishes the training and obtains enough messages, oracle should answer the test query once. Finally, we judge if the adversary wins or loses by what the adversary gets. The adversary and the oracle are denoted by $A$ and $B$, respectively.

Init: before replying to queries of $A$, $B$ generates the system parameters including security parameter $N$, a multiplication cyclic group $G$, and a generator $g \in G$ with order $p$, where $p$ is a large prime number related to $N$. Then $B$ selects random numbers $k_x \in Z_p$ and computes $PK_x = g^{k_x} \mod p$ for $x \in \{H, P, D, C\}$. We notice that in a complete system $P$ and $D$ are not unique. Then $B$ prepares public key and secret key pairs denoted by $(PK_P, k_P)$ and $(PK_D, k_D)$ for $P_i$ and $D_j$, where $i \in [1, n]$ and $j \in [1, m]$. $B$ marks up all $P$ and $D$ with void state. Then it maintains a list of $SID$ recording simulated conversations. $SID_{P,D}^e$ represents the $e$th conversation involved $P_i$ and $D_j$. Noticeably, any $SID$ has a void state before being invoked.

After the init phase, $A$ is allowed to make queries for simulating the real protocol.

Corrupt($P_i$): $B$ gives $k_P$ back to $A$ and marks up the state of $P_i$ with corrupted.

Corrupt($D_j$): $B$ gives $k_D$ back to $A$ and marks up the state of $D_j$ with corrupted.

Reveal($SID_{P,D}^e$): this query simulates abuse of session keys $sk_e$. 

---

**Figure 5: Checking report phase.**

<table>
<thead>
<tr>
<th>$P$</th>
<th>$C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generates random number $v_p$ and timestamp $T_p^2$, Computes $K_1 = g^{v_p}$, $s_{1k} = PK_1^{v_p} = g^{s_{1k}}$, $s_{1k} = h(PK_1^{v_p} | T_p^2) = h(g^{s_{1k}} | T_p^2)$, $C_1 = E_{v_p}(ID_P, s_{1k})$.</td>
<td>${C_1, T_2^2, K_1}$ Checks the validity of $T_2^2$, Computes $s_{1k} = K_1^{v_p} = g^{s_{1k}}$, Decrypts $C_1$ with $s_{1k}$ to obtain $s_{1k}$, Checks $s_{1k} = h(PK_1^{v_p} | T_p^2)$, Generates random numbers $v_c$ and timestamp $T_c^e$, Computes $K_e = g^{v_c}$, $sk_{PC} = K_e^{v_c} = g^{sk_{PC}}$, $s_{1k} = h(PK_e^{v_c} | sk_{PC} | T_c^e)$, ${K_e, s_{1k}, C_e, T_c^e}$ Checks the validity of $T_c^e$, Computes $sk_{PC} = K_e^{v_c} = g^{sk_{PC}}$, Checks $s_{1k} = h(PK_e^{v_c} | sk_{PC} | T_c^e)$, Computes $(m_{1x}, m_{2x}, m_{3x}, Sig_{ID}) = D_{E_{sk_{PC}}}(C_e)$, Verifies $m_{1x} = V_{PK_e}(Sig_{ID})$, Computes $C_x = E_{sk_{PC}}(m_{1x}, m_{2x}, m_{3x},)$, $s_{1k} = E_{sk_{PC}}(ID_P, C_e, s_{1k})$.</td>
</tr>
</tbody>
</table>
We denote upper bound of adversaries' advantage against DDH as \( \epsilon_{\text{DDH}} \). So \( \epsilon_{\text{DDH}} \) should be negligible if the assumption is right.

**Assumption 3** (hash). There exists a secure irreversible hash function which achieves strong collision resistance.

We denote the advantage of adversaries against the hash function as \( \epsilon_{\text{hash}} \). So \( \epsilon_{\text{hash}} \) should be negligible if the assumption is right.

**Assumption 4** (signature). There exists a secure digital signature scheme.

We denote advantage of adversaries against this signature scheme as \( \epsilon_{\text{sig}} \). So \( \epsilon_{\text{sig}} \) should be negligible if the assumption is right.

**Proof.** A PPT adversary \( \mathcal{A} \) is attacking the protocol. We use a series of games to bound the advantage of \( \mathcal{A} \). The advantage of \( \mathcal{A} \) in Game 1 is defined as

\[
\text{Adv}_{1} \triangleq 2 \cdot \Pr [\mathcal{A} \text{ succeeds in Game } i] - 1. \tag{1}
\]

The games used to bound the advantages of \( \mathcal{A} \) are listed in the following. We analyse the advantage difference in nearly games and bound them. In Game 0, it would be the real protocol.

**Game 0.** \( \mathcal{A} \) interacts with the initial security model.

**Game 1.** In this game, we modify \texttt{Execute} queries. When the states of \( \mathcal{P}_{i}, \mathcal{D}_{j} \), and \( \text{SID} \) are all void, \( \mathcal{B} \) simulates a real protocol but replaces \( s_{1}, sk_{\text{KC}}, s_{2}, sk_{\text{PC}}, s_{9}, sk_{\text{DC}}, s_{13} \) and \( sk_{\text{PC}} \) with random numbers in \( \mathbb{G} \).

**Lemma 1.** \(|\text{Adv}_{0} - \text{Adv}_{1}| \leq \text{negl.} \)

**Proof.** We just replace the \( sks \) of traditional DH protocol with random numbers. The advantage difference between two games is caused by DDH problem. And hence, Lemma 1 is proved by DDH assumption right.

**Game 2.** This game is based on Game 1 and we also modify \texttt{Execute} queries. When the states of \( \mathcal{P}_{i}, \mathcal{D}_{j} \), and \( \text{SID} \) are all void, \( \mathcal{B} \) simulates a real protocol but replaces \( s_{2}, s_{3}, s_{6}, s_{7}, s_{10}, s_{11}, s_{14}, s_{15} \) with uniform random numbers in the range of hash function.

**Lemma 2.** \(|\text{Adv}_{1} - \text{Adv}_{2}| \leq \text{negl.} \)

**Proof.** We just replace the real hash results with random numbers. Without the knowledge of inputs, the probability that \( \mathcal{A} \) can distinguish the real hash results and random numbers is less than the advantage of \( \mathcal{A} \) that captures the hash. And hence, if the hash function is secure, the probability is negligible.
Table 1: Comparisons of properties.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Resistance of impersonation attack</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Prevention of replay attack</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Preserving patient privacy</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Complete mutual authentication</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Perfect forward secrecy</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Perfect backward secrecy</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Proof. We just replace the encryption results by random numbers. Without the knowledge of inputs, the probability that $A_d$ can distinguish the real encryption results and random numbers is less than the advantage of $A_d$ that captures the CPA security symmetric encryption. If the symmetric encryption is secure, the probability is negligible.

In Game 6, we notice that $P_i$ and $D_j$ of test queries are all randomized. So in Game 6, the advantage of $A_d$ is zero. So, we can compute the $Adv_0$ as follows:

$$Adv_0 = |Adv_0 - Adv_\emptyset| \leq \epsilon_{DKH} + \epsilon_{hash} + \epsilon_{enc},$$

which is a negligible value.

4. Performance and Functionality Analysis

Herein, we evaluate the performance and functionality of the proposed scheme and compare it with three related schemes for cloud based telemedicine systems, including Chen et al.'s scheme [15], Chiou et al.'s scheme [16], and Cheng et al.'s scheme [20].

The comparisons on the key security properties among these systems are given in Table 1. It is visible that our scheme could achieve all security properties and it is superior to the rest three related schemes. Chen et al.'s scheme [15] fails to provide anonymity and complete mutual authentication, while Chiou et al.'s scheme [16] could not achieve the complete mutual authentication. Furthermore, Cheng et al.'s scheme [20] could not preserve users' privacy, complete mutual authentication, and confidentiality. Note that the proposed scheme offers important security features and it is better suitable for cloud based telemedicine environment.

Meanwhile, we present the comparisons of efficiency in terms of computation loading among these schemes in Table 2. Compared with the other three related schemes, the proposed scheme needs not perform the bilinear pairing and could provide more additional security features. Furthermore, our scheme achieves the provably security in the standard model.

More detailed efficiency comparisons are shown in Figures 6 and 7. We implement the cloud of authentication schemes for cloud based telemedicine systems in Python 3.5.2 using an Intel(R) Core(TM) i5-4590 CPU @ 3.30GHZ with 3300MB RAM and Ubuntu 16.04 system. The simulations of platform for healthcare center, patients, and doctors are...
Table 2: Comparisons of computation loading.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>HUP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>$T_s + T_a + 2T_m + 2T_p + T_E + 3T_h$</td>
<td>$T_s + 2T_p + 2T_E + 4T_h$</td>
<td>$2T_p + 2T_E + 4T_h + T_s + T_m$</td>
<td>$5T_p + 2T_E + T_s + 3T_h$</td>
</tr>
<tr>
<td>C</td>
<td>$2T_m + 2T_p + T_E + 3T_h$</td>
<td>$T_p + T_E + 3T_h$</td>
<td>$3T_h + T_p + T_E$</td>
<td>$4T_p + 2T_E + 2T_h$</td>
</tr>
<tr>
<td>P</td>
<td>$T_s + T_a + 2T_m + 2T_p + 2T_E + 3T_h$</td>
<td>$T_p + 2T_E + 7T_h$</td>
<td>$T_s + 5T_h$</td>
<td>$T_p + 2T_E + 2T_h$</td>
</tr>
<tr>
<td>C</td>
<td>$2T_m + 2T_p + T_E + 3T_h$</td>
<td>$T_p + 4T_h$</td>
<td>$4T_h$</td>
<td>$4T_p + 2T_E + 2T_h$</td>
</tr>
<tr>
<td>TP</td>
<td>$2T_s + 2T_m + 2T_p + 2T_E + 3T_h$</td>
<td>$2T_s + 2T_p + 2T_E + 4T_h$</td>
<td>$2T_s + 2T_p + 5T_h$</td>
<td>$5T_p + 4T_E + 2T_h$</td>
</tr>
<tr>
<td>C</td>
<td>$2T_m + 2T_E + 3T_h$</td>
<td>$T_p + 4T_h$</td>
<td>$4T_h$</td>
<td>$4T_p + 2T_E + 4T_h$</td>
</tr>
<tr>
<td>CP</td>
<td>N/A</td>
<td>$T_s + T_p + 2T_E + 4T_h$</td>
<td>$T_s + T_p + 2T_E + 4T_h$</td>
<td>$6T_p + 4T_E + 2T_h + T_s$</td>
</tr>
<tr>
<td>C</td>
<td>N/A</td>
<td>$T_p + 4T_h$</td>
<td>$4T_h$</td>
<td>$4T_p + 2T_E + 2T_h$</td>
</tr>
</tbody>
</table>

HUP: healthcare center uploading phase.
PUP: patient uploading phase.
TP: treatment phase.
CP: checking report phase.

$T_s$: time consumption for executing a hash function.
$T_p$: time consumption for executing the symmetric encryption/decryption operation.
$T_m$: time consumption for executing a modular exponent operation.
$T_p$: time consumption for executing a bilinear pairing operation.
$T_a$: time consumption for executing an asymmetric encryption/decryption operation.
$T_m$: time consumption for executing a multiplication operation.
implemented in Python 3.5.2 using an Intel(R) Core(TM) i5-4590 CPU at 1.65GHZ with 1540MB RAM and Ubuntu 16.04 system. The one-way hash function used is SHA-256, and the symmetric encryption/decryption algorithm is advanced encryption standard. We use the ElGamal signature scheme and ElGamal encryption scheme with 1024-bit security parameter for digital signature algorithm and the asymmetric encryption/decryption algorithm, respectively. Moreover, the bilinear paring is simulated in two MNT asymmetric groups, “MNT224”.

Figure 6 shows the main cost on the cloud computing of interacting with multiple patients and doctors for authentication simultaneously. It demonstrates that our proposal costs less time for the cloud to authenticate doctors and patients. Figure 7 illustrates the main cost on healthcare center, patients, doctors, and cloud for one round authentication in healthcare center uploading phase, patient uploading phase, treatment phase, and checking report phase, respectively.

From Figure 7 we can conclude that our scheme is the most efficient to finish one round mutual authentication.

5. Conclusion

In this article, we proposed an anonymous authentication scheme based on cloud for medical environment, which provided both data confidentiality and message authenticity. Subsequently, we stated that the proposed scheme was provably secure in the standard model. The comparisons with existing competitive protocols also observe that our scheme is suitable for the cloud based telecare medical information systems.

Data Availability

The data used to support the findings of this study are included within the article.
Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by NSFC (Grant Nos. 61602045, 61502044).

References
