Abstract: The advent and rapid rise of network technology and cloud computing have led to new opportunities for ushering in a new era in telehealth. Thanks to the Internet of Things (IoT) and advances in 5G communication, telehealth is expanding and shows no signs of slowing down. It provides patients including elderly and disabled patients with convenient and easy access to healthcare services across space and time. However, the continuous real-time transmission of health information over networks also exposes private data to the risk of being intercepted by third parties. The privacy of the primary individual patient must be managed under the protection of the patient’s anonymous key while storing, transferring, sharing, and adding privacy rights. A question arises: How can we design a secure communication environment for remote access control to personal privacy matters? The patient’s electronic medical record is protected by the patient’s private key, and our scheme provides a real anonymous design for the patient with absolute autonomy over their privacy. Each update of the cloud medical records is patient-led and performed in a secure tunnel. As a result, this study reveals that the cloud-based iPatient privacy copyright management fully controlled by an individual patient is indeed safe and effective.

Keywords: iPatient; telehealth; electronic patient record; privacy copyright management

1. Introduction

Personal handwritten medical records are being phased out, and electronic personal medical records are now commonplace. An iPatient, which refers to a patient-led, privacy and medical record copyright issue, is rising to the challenge of privacy security. Most medical records kept by hospitals or private clinics are viewed and distributed without the patient’s authorization. As such, many medical agencies have begun to pay attention to the importance of this issue and have begun to design a medical record database system that is centrally managed and shared. Most of the current patient privacy and medical record copyrights based on medical priority can be read by medical personnel with unlimited access only by a one-time authorization. Such privacy protection is obviously deficient. Anyone maliciously intending to obtain the privacy of others’ medical records can easily illegally obtain relevant information. Hence, how to let patients fully control their own personal privacy and medical record copyright has become an important issue in today’s privacy-sensitive information-dominated society. With the advent of the cloud computing era, most data are stored in cloud servers; therefore, we call these data management solutions “iPatient privacy copyright cloud management.”

Additionally, telehealth has received considerably increased attention. In this medical treatment approach, all the medical prescriptions, personal privacy, medical record copyright, etc., are transmitted on the public network and the data are stored in the cloud system. The main topics in this paper include the following: How to ensure telehealth visits occur in a secure transmission environment, how to securely transfer data and securely process data, and how to ensure that all actions related to patient privacy are performed with the patient’s full authorization. The management of general medical
records is mainly through a secure tunnel to encrypt and decrypt data. The tunnel is constructed by a cryptographic asymmetric public key system or symmetric key agreement. Asymmetric public-key encryption and decryption methods need to maintain huge public and private key pairing creation and storage costs. Accordingly, in recent years, symmetric encryption and decryption by a session key agreement mechanism has managed privacy issues. The main point of the key agreement mechanism is to design a set of zero-knowledge-based communications. The two communication parties cannot transmit information that is derived or reversibly decrypted to obtain the private key, and also cannot reveal personal identities during the medical communication process. This paper proposes, under the above premises, an application to provide medical personal privacy and a copyright access mechanism that is managed in the cloud, where the entire process is conducted in a patient-controlled authorization environment, which can be seen in Figure 1. The paper is laid out as follows. Section 2 introduces the related work. Our protocols are found in Section 3. Security analysis is shown in Section 4. Section 5 shows performance comparisons. Section 6 illustrates the conclusions.

Figure 1. iPatient privacy copyright cloud management system scenario map.

2. Related Work

In this study, we used a symmetric key to encrypt and decrypt all transmitted medical record copyrights. Therefore, we established a secure tunnel for all transmitted data and designed a symmetric key agreement mechanism. The key agreement was roughly divided into two parts, namely offline processing and online processing. In the case of offline processing, the user had to use a token that was unavailable to others to initialize the key agreement. Prior studies used personal identity IDs and passwords for preliminary verification, which is often referred to as the two-factor verification method [1,2].

When coupled with other auxiliary mechanisms or biological characteristics, they are often referred to as three-factor authentication [3,4]. However, new research mostly used fuzzy hashing conducting one-way exclusive pairing for biological characteristics, changing slightly each time [5,6]. Our research used fuzzy hashing plus identity anonymity; thus, it is a single-factor dynamic anonymous offline processing method [3,4,7]. Offline processing security vulnerability issues include smart card loss or brute force password cracking attacks, and many prior studies have also proposed possible vulnerabilities and solutions [8–10]. Alternatively, in the case of offline processing, a zero-knowledge information exchange key agreement was performed through the public network [11]. In the case of the key agreement mechanism, many scholars had proposed related agreements [8,11–15]. In 2002, using fingerprints as passwords and using a remote login authentication through smart cards and biometrics were common [9]. Two years later, the improperly designed communication process posed a risk of masquerade attacks even though biometrics are hard to counterfeit [16].

In 2012, a study addressed a medical information system for an integrated electronic patient record (EPR) with a one-way operation hash function as the main security mechanism. The study also reduced the time of encryption and decryption operations through multiplications [17]. However, it was still possible to carry out impersonation attacks although prior studies provided irreversible
encryption and decryption operations for medical information systems. Impersonators can obtain fake user information by hacking the database [12]. In the case of user anonymity, in 2017, some scholars addressed anonymous mechanisms [3,4,7,14,15]. However, our study found that the anonymity in many studies was pseudonymous because the same alias was also easily traced to an association with the real identity. Our study will design one real untraceable anonymous agreement. As telehealth has received substantially increased attention, cloud data communication and storage security for telehealth have become quite important issues [3,18–23]. There are many possible malicious attacks in traditional communication security issues such as man-in-the-middle, spoofing, and replay attacks [23–26]. Recently, Lin [27] proposed an authorized sharing mechanism for telehealth. Several weaknesses existed in Lin’s scheme. Lin’s scheme only discussed security protocols transferring data from the data centers to the hospital and lacked a design for storage or modification. Hospitals and data centers did not know the user’s location and the real address, and consequently, they were unable to directly send communication messages. Additionally, the immutable pseudonyms were not truly anonymous and the medical record update did not establish a user-led secure channel, and data were probably at risk of being tampered with or counterfeited [27].

The main points of our study focus on the patients’ complete anonymity but fully controlled through a zero-knowledge key agreement mechanism withstanding existing malicious communication attacks. All sensitive information is processed and transmitted after the secure tunnel is established. We designed an iPatient privacy copyright cloud management system, the details of which are shown in Figure 2.

In this section, we describe our iPatient copyright privacy cloud management protocols. We apply the patient personal biometric signal and lightweight biometric hash function to encipher the security for the patient’s private copyright. Our proposed protocol contains three phases, namely the initialization phase, registration phase, and electronic patient record sharing and updating phase. A scenario of iPatient copyright privacy cloud management is illustrated in Figure 2. Table 1 lists the notations used in the proposed protocols.

3. The Proposed Mechanism

Figure 2. Our proposed iPatient privacy copyright cloud management system.
Table 1. Notations table.

| Notations | Description |
|-----------|-------------|
| $B_i$     | The biometric signal of $P_i$ |
| CA        | Certificate Authority |
| $D_j$     | j-th Hospital Doctor/Clinic |
| $D_K(.)$  | Symmetric decryption function using key $K$ |
| $E_K(.)$  | Symmetric encryption function using key $K$ |
| EPRS      | Electronic Patient Record Server |
| $h(.)$    | One-way hash function |
| $H(.)$    | One-way fuzzy hash function suitable for biometric ID |
| $ID_{Pi}$ | Patient $P_i$'s identity |
| $ID_{DJ}$ | Hospital Doctor/Clinic $D_j$'s identity |
| $ID_e$    | EPRS’s identity |
| $P_i$     | i-th Patient |
| $R_i$     | Electronic patient record of $P_i$ |
| $T_{CA}$  | The secret token of CA |
| $\oplus$ | Exclusive-OR operation |

3.1. Initialization Phase

In our mechanism, there are three roles. Each role must be registered and initialized with a certificate authority (CA) to obtain a passcode for verification. In the first phase, the electronic patient record server (EPRS) performs initial registration information with a CA, as seen in the following two stages.

Stage 1. The EPRS registers on-site the first time with a CA. The CA generates the identifier $ID_e$ for the EPRS. The CA computes the token $EC = h(ID_e \oplus T_{CA})$, stores $ID_e$ in its CA members database, and sends message tuple $\{ID_e, EC\}$ to the EPRS via a secure channel.

Stage 2. The EPRS receives the message tuple $\{ID_e, EC\}$ from $P_i$ and then calculates the secret token $V_i = ID_{Pi} \oplus H(B_i)$, generates a random number $R_{Pi}$, computes secret token $S_i = R_{Pi} \oplus h(T_{CA})$, and $M_1 = R_{Pi} \oplus h(ID_e \oplus T_{CA})$. Then, the CA stores $\{V_i, ST_i, S_i\}$ to the smart card and stores the patient ID $ID_{Pi}$ to its ID list table. The CA then sends the smart card back to the $P_i$ through the secure channel and sends $\{ST_i, M_1\}$ to the EPRS over the public channel.

Stage 3. The EPRS receives the message $\{ST_i, M_1\}$ from the CA and then computes $R_{Pi} = M_1 \oplus EC$, $ID_{Pi} = ST_i \oplus R_{Pi}$, creates a new patient record EPR $\{R_i\}$ for $ID_{Pi}$, encrypts $R_i$ using $RP_i\{E_{RPi}(R_i)\}$, and stores $ID_{Pi}, E_{RPi}(R_i)$ to EPRS’s database.

3.2. Registration Phase

In the second phase, two roles, namely patients and remote hospital doctors or clinics, need to register at the CA. We describe these two roles separately in the following subsections.

3.2.1. Patients Registration Phase

Patients must register the first time with the CA in the prior EPRS registration phase and create patient accounts on the EPRS in the following three stages.

Stage 1. $P_i$ keys in onsite his/her identity $ID_{Pi}$ and imprints the fingerprint $B_i$; then, the CA obtains $\{ID_{Pi}, H(B_i)\}$.

Stage 2. The CA receives the message $\{ID_{Pi}, H(B_i)\}$ from $P_i$ and then calculates the secret token $V_i = ID_{Pi} \oplus H(B_i)$, generates a random number $R_{Pi}$, computes secret token $S_i = R_{Pi} \oplus h(T_{CA})$, and $M_1 = R_{Pi} \oplus h(ID_e \oplus T_{CA})$. Then, the CA stores $\{V_i, ST_i, S_i\}$ to the smart card and stores the patient ID $ID_{Pi}$ to its ID list table. The CA then sends the smart card back to the $P_i$ through the secure channel and sends $\{ST_i, M_1\}$ to the EPRS over the public channel.

Stage 3. The EPRS receives the message $\{ST_i, M_1\}$ from the CA and then computes $R_{Pi} = M_1 \oplus EC$, $ID_{Pi} = ST_i \oplus R_{Pi}$, creates a new patient record EPR $\{R_i\}$ for $ID_{Pi}$, encrypts $R_i$ using $RP_i\{E_{RPi}(R_i)\}$, and stores $ID_{Pi}, E_{RPi}(R_i)$ to EPRS’s database.

3.2.2. Remote Hospital Doctors or Clinics Registration Phase

The hospital doctors or clinics must also register the first time with the CA in the prior patient’s registration phase by the following two stages.
Stage 1. Doctor $D_j$ registers on-site the first time with a CA. The CA generates the identifier $ID_j$ for the $D_j$ and then computes $DC_j = h((ID_j) \oplus T_{CA})$, stores $DC_j$ to a medical card of doctor $D_j$, and then sends back to the doctor $D_j$ via a secure channel.

Stage 2. $D_j$ receives the personal medical card from the CA and keeps the medical card on hand.

3. Secure Electronic Patient Record Storing, Sharing, and Updating Communication Phase

This phase involves the stages of health medical privacy creation, sharing, and storage. In our iPatient access control model, doctors or clinics have read and updated permissions, while authorized professionals have only read permissions. Only patients themselves can access the encrypted personal privacy of the cloud server. Additionally, all data stored in the cloud are also protected by the patient’s private key. All decrypted data must be transmitted in a secure tunnel established through a key agreement. Below, we show six strategies in the iPatient access control model and show the strategy diagram in Figure 3.

![Figure 3. Our iPatient access control model.](image_url)

1. The privacy of all patients is protected and stored in a cloud database via the patient’s private key.
2. Anyone who wants to access patient privacy records must obtain the patient’s authorization and obtain the records of the patient via patient downloading from the cloud database.
3. In the case of protecting the privacy of the cloud database, only patients can access via key agreement to establish a secure tunnel. After the tunnel is established, the transmitted data still need to be encrypted with the session key before it can be transmitted securely.
4. Doctors modify, update patient records, and send them to the patient via the session key. After the patient decrypts the updated patient records, s/he will use the personal private key to encrypt them, and then encrypt them with the session key to upload them to the cloud database server. Once the cloud data server receives and decodes them with the session key, it will update them to the patient records database.
5. Other authorized professionals excluding doctors can only read the patient records transmitted by the patient through the secure tunnel after they have been authenticated by the key agreement. They cannot modify or update patient records.
6. Our design for iPatient is personally led and we want to establish a privacy sharing mechanism that is completely patient-led.

Below, we introduce the steps we designed. First, we perform the three-party key agreement process in order to build a secure tunnel, and then decrypt, add, and modify the medical records in a secure tunnel.

Stage 1. $P_i$ imprints the biometric $B_i$ and then deciphers his/her identity by $ID_{P_i} = V_i \oplus H(B_i)$, deciphers $R_{P_i}^{old} = ST_i \oplus ID_{P_i}$, $NID_{P_i} = ID_{P_i} \oplus R_{P_i}^{new}$, and generates a new random number $R_{P_i}^{new}$, $M_1 = R_{P_i}^{old} \oplus S_i \oplus R_{P_i}^{new}$. $P_i$ then sends message ($NID_{P_i}, NID_{D_j}, M_1$) to the CA via a public channel.
Stage 2. After receiving the message from the $P_i$, the CA deciphers $R_{Pi}^{\text{new}} = M_1 \oplus h(T_{CA})$, $ID_{Pi} = NID_{Pi} \oplus R_{Pi}^{\text{new}}$, and checks the patient identity $ID_{Pi}$ from the CA’s ID table. If the legal list cannot find the $ID_{Pi}$, the request is terminated. Then, the CA calculates $ID_{Di} = NID_{Di} \oplus R_{Pi}^{\text{new}}$, computes $M_2 = R_{Pi}^{\text{new}} \oplus h(ID_i \oplus T_{CA})$, $M_3 = R_{Pi}^{\text{new}} \oplus h(ID_{Di} \oplus T_{CA})$, and then sends a request of message and parameters tuple $\{NID_{Pi}, NID_{Di}, M_2\}$ to the EPRS through the public channel. Then, the CA sends $\{NID_{Pi}, M_3\}$ to the doctor $D_i$ via a public channel.

Stage 3. After receiving the message $\{NID_{Pi}, NID_{Di}, M_2\}$ from the CA, the EPRS computes $R_{Pi}^{\text{new}} = M_2 \oplus EC$, $ID_{Pi} = NID_{Pi} \oplus R_{Pi}^{\text{new}}$, and checks $ID_{Pi}$ from the EPRS database. Then, the EPRS generates a random number $R_{EPRS}$, generates secret token $M_4 = R_{Pi}^{\text{new}} \oplus R_{EPRS}$, and then sends $M_4$ to doctor $D_i$ via a public channel.

Stage 4. After receiving the message $\{NID_{Pi}, M_3\}$ from the CA and $M_4$ from the EPRS, $D_i$ computes $R_{Pi}^{\text{new}} = M_3 \oplus DC$, $ID_{Pi} = NID_{Pi} \oplus R_{Pi}^{\text{new}}$, generates a random number $R_{D_i}$, computes $R_{EPRS} = M_4 \oplus R_{Pi}^{\text{new}}$, $M_5 = R_{D_i} \oplus R_{EPRS}$, and then sends $\{M_5, NID_{Pi}\}$ to the CA and $\{M_5\}$ to the EPRS via a public channel. Then, $D_i$ owns the session key $SK = h(R_{Pi}^{\text{new}} \oplus R_{EPRS} \oplus R_{D_i})$.

Stage 5. The EPRS receives the message $\{M_5\}$ from $D_i$, computes $R_{D_i} = M_5 \oplus R_{EPRS}$, and then computes session key $SK = h(R_{Pi}^{\text{new}} \oplus R_{EPRS} \oplus R_{D_i})$.

Stage 6. The CA receives the message $\{M_5, NID_{Pi}\}$ from doctor $D_i$ and then sends $M_5$ to $P_i$ by checking $NID_{Pi}$.

Stage 7. After receiving the message from the CA, $P_i$ computes session key $SK = h(R_{Pi}^{\text{new}} \oplus M_5)$, updates secret tokens $ST_i = ST_i \oplus R_{Pi}^{\text{new}} \oplus R_{Pi}^{\text{old}}$ and $S_i = S_i \oplus R_{Pi}^{\text{new}} \oplus R_{Pi}^{\text{old}}$, and then finishes key agreement.

After the key agreement process is done, the secure tunnel among $P_i$, the EPRS, and $D_i$ is established. The three roles can use the same one-time session key for secure communication. Next, the patient privacy history update process under the secure channel is detailed in Stage 8 to Stage 12.

Stage 8. The EPRS uses the secret session key $SK$ to encrypt the privacy record of the $ID_{Pi}$ patient, $M_6 = \{E_{SK}(E_{R}(R_i))\}$, and then sends $M_6$ to patient $P_i$.

Stage 9. $P_i$ receives the encrypted electronic patient record $\{E_{SK}(E_{R}(R_i))\}$ and decrypts $E_{SK}(E_{R}(R_i))$ using session key $SK$ to obtain $E_{R}(R_i)$. Then, $P_i$ applies $R_{Pi}^{\text{old}}$ to decipher the original $R_i = D_{R_{Pi}^{\text{old}}}(E_{R}(R_i))$ and then enciphers the original $R_i$ with session key $SK$ to message $M_7 = E_{SK}(R_i)$. Then, $P_i$ sends $M_7$ to doctor $D_i$.

Stage 10. The doctor $D_i$ receives the message $M_7$ from $P_i$ and deciphers the original record $R_i = D_{SK}(M_7)$. When the telehealth service is done, the doctor generates a new record $R_{i}^{\text{new}}$ and then enciphers the new record $R_{i}^{\text{new}}$ using session key $SK$ to obtain message $M_8 = E_{SK}(R_{i}^{\text{new}})$. Then, the doctor $D_i$ sends encrypted privacy record $M_8$ to $P_i$.

Stage 11. $P_i$ receives the message from $D_i$ and uses session key SK to obtain $R_{i}^{\text{new}} = D_{SK}(M_8)$. Then, $P_i$ enciphers himself the privacy record to obtain $M_9 = E_{SK}(E_{R_{Pi}^{\text{new}}}(R_{i}^{\text{new}}))$ and sends $M_9$ to the EPRS.

Stage 12. The EPRS receives the message $M_9$ from $P_i$ and then computes $M_{10} = D_{SK}(M_9)$. The EPRS updates $ID_{Pi}$’s electronic patient record using $M_{10}$ $\{E_{R_{Pi}^{\text{new}}}(R_{i}^{\text{new}})\}$ and then finishes the data sharing and data update phase.

3.4. Algorithms for Key Agreement, Data Encryption/Decryption, and Updating

In this subsection, we also give another presentation to describe our protocols. We use different roles to divide the algorithms into four independent but concurrent procedures. As described from Algorithms 1-4.
Algorithm 1: Procedure of Patient $P_i$

1. $B_i = \text{imprints (}P_i\text{'s biometric)}$  // Input patient’s biometric
2. $ID_{P_i} = V_i \oplus H(B_i)$  // Retrieves patient’s identity
3. $R_{P_i}^{\text{old}} = ST_i \oplus ID_{P_i}$  // Retrieves prior random number
4. $R_{P_i}^{\text{new}} = \text{Random()}$  // Generates a new random number
5. $NID_{P_i} = ID_{P_i} \oplus R_{P_i}^{\text{new}}$  // Generates new nick ID
6. $M_4 = R_{P_i}^{\text{old}} \oplus S_i \oplus R_{P_i}^{\text{new}}$  // Replaces $S_i$ to $M_4$ with new random number
7. Sends $\{ \text{[}NID_{P_i}, NID_{D_j}, M_4\text{]}\}$, CA)  // Sends communication message to CA
8. while (!receive from CA) { do nothing }  // Waiting for the response from CA
9. $SK = h(R_{P_i}^{\text{new}}@EPRS@R_{D})$  // Computes session key $h(R_{P_i}^{\text{new}}@EPRS@R_{D})$
10. $ST_i = ST_i \oplus R_{P_i}^{\text{new}} \oplus R_{P_i}^{\text{old}}$  // Updates secret tokens $ST_i$
11. $S_i = S_i \oplus R_{P_i}^{\text{new}} \oplus R_{P_i}^{\text{old}}$  // Updates secret tokens $S_i$
12. while (! receive from CA) { do nothing }  // The following statements for privacy data processing in secure tunnel
13. $E_g(R_i) = D_{SK}(E_{SK}(E_{SK}(R_i)))$  // Deciphers message using session key $SK$
14. $R_i = D_{SK}^{\text{old}}(E_{SK}(R_i))$  // Deciphers record using private key $R_{P_i}^{\text{old}}$
15. $M_7 = E_{SK}(R_i)$  // Enciphers the original $R_i$ with session key $SK$
16. Sends $\{ \text{[}M_7, D_j\text{]}\}$  // Sends cyber record to doctor $D_j$
17. while (! receive from $D_j$) { do nothing }  // Waiting for the response from $D_j$
18. $R_{P_i}^{\text{new}} = D_{SK}(M_8)$  // Deciphers new record using session key $SK$
19. $M_9 = E_{SK}(E_{EPRS}(R_{P_i}^{\text{new}}))$  // $P_i$ enciphers privacy using private key $R_{P_i}^{\text{new}}$
20. Sends $\{ \text{[}M_9\text{]}, EPRS\}$  // Sends cyber record to $EPRS$

Algorithm 2: Procedure of CA

1. while (! receive from Patient $P_i$) { do nothing }  // Waiting for the ack from Patient $P_i$
2. $R_{P_i}^{\text{new}} = M_1@h(T_{CA})$  // Deciphers random number
3. $ID_{P_i} = NID_{P_i}@R_{P_i}^{\text{new}}$  // Retrieves patient’s ID
4. if (! exist (ID_{P_i})) exit()  // Checks if ID_{P_i} is illegal or not, if not exist session
5. $ID_{D_j} = NID_{D_j}@R_{P_i}^{\text{new}}$  // Retrieves doctor’s ID
6. $M_2 = R_{P_i}^{\text{new}}@h(\text{ID}_{D_j}@T_{CA})$  // Builds exchange message $M_2$
7. $M_3 = R_{P_i}^{\text{new}}@h(\text{ID}_{D_j}@T_{CA})$  // Builds exchange message $M_3$
8. Sends $\{ \text{[}NID_{P_i}, NID_{D_j}, M_2\text{]}, EPRS\}$  // Sends communication message to EPRS
9. Sends $\{ \text{[}NID_{P_i}, M_3\text{]}, D_j\}$  // Sends communication message to $D_j$
10. while (!receive (M_3)) { do nothing }  // Waiting for the response from $D_j$
11. Sends $\{ \text{[}M_5\text{]}, whois(NID_{P_i})\}$  // Forwards communication message to $P_i$

Algorithm 3: Procedure of doctor $D_j$

1. while (! receive from $P_i$) { do nothing }  // Waiting for the ack from CA
2. $R_{P_i}^{\text{new}} = M_3@DC$  // Deciphers random number $R_{P_i}^{\text{new}}$
3. while (! receive from $M_3$) { do nothing }  // Waiting for the ack from EPRS
4. $R_2 = \text{Random()}$  // Generates a new random number
5. $R_{EPRS}^{\text{new}} = M_4@R_{P_i}^{\text{new}}$  // Deciphers random number $R_{EPRS}^{\text{new}}$
6. $M_5 = R_2@R_{EPRS}$  // Builds exchange message $M_5$
7. Sends $\{ \text{[}M_5, NID_{P_i}\text{]}, CA\}$  // Forwards communication message to CA
8. Sends $\{ \text{[}M_5\text{]}, EPRS\}$  // Forwards communication message to EPRS
9. $SK = h(R_{P_i}^{\text{new}}@R_{EPRS}@R_{D})$  // Computes session key $h(R_{P_i}^{\text{new}}@R_{EPRS}@R_{D})$
10. while (! receive from $P_i$) { do nothing }  // Waiting for the response from $P_i$
11. $R_i = D_{SK}(M_5)$  // Deciphers the original record $R_i$ using $SK$
12. $R_i^{\text{new}} = \text{update } R_i^{\text{old}}$ ( )  // Telehealth service
13. $M_4 = E_{SK}(R_i^{\text{new}})$  // Enciphers the $R_i^{\text{new}}$ with session key $SK$
14. Sends $\{ \text{[}M_8\text{]}, P_i\}$  // Sends cyber record to $P_i$
Algorithm 4: Procedure of doctor EPRS

1. while (! (receive (M2))) { do nothing } // Waiting for the ack {NID\textsubscript{P}, NID\textsubscript{Dj}, M2} from CA
2. \( R^{\text{new}}_D = M_2 \oplus \text{EC} \) // Deciphers random number \( R^{\text{new}}_D \)
3. \( ID_{P} = NID_{P} \oplus R^{\text{new}}_D \) // Retrieves patient’s ID
4. if (! exist (ID\textsubscript{P})) exit() // Checks if ID\textsubscript{P} is illegal or not, if not exist session
5. \( R_{EPRS} = \text{Random}() \) // Generates a new random number
6. \( M_4 = R^{\text{new}}_P \oplus R_{EPRS} \) // Builds exchange message \( M_4 \)
7. Sends \{ \{M_4\}, Dj\} // Forwards communication message to \( D_j \)
8. while (! (receive (M5))) { do nothing } // Waiting for the ack \{M5\} from \( D_j \)
9. \( SK = h(R^{\text{new}}_P@R_{EPRS}@h(M_5@R_{EPRS})) \) // Computes session key \( h(R^{\text{new}}_P@R_{EPRS}@R_{D}) \)

// The following statements for privacy data processing in secure tunnel
10. \( M_6 = [E_{SK}(E_K(R_i))] \) // EPRS encrypts the privacy record of the ID\textsubscript{P}
11. Sends \{ \{M_6\}, P \} // Sends encrypted privacy of \( P_i \) to \( P_i \)
12. while (! (receive from \( P_i \))) { do nothing } // Waiting for the response from \( P_i \)
13. \( M_{10} = D_{SK}(M_9) \) // Decipher \( M_9 \) using session key \( SK \)
14. Updates (M\textsubscript{10}, ID\textsubscript{P}) // EPRS updates ID\textsubscript{P}'s electronic patient record

4. Security Analysis

In this section, we highlight some of the attributes and well-known attack types in the encryption and decryption process of long-distance data transmission and describe, in detail, the reasons why our designed protocol can resist attacks. The defenses of user anonymity, stolen or lost smart card attack, man-in-the-middle attack, replay attack, tamper attack, insider privilege attack, verify table stolen attack, off-line password guessing attack, and known key security are illustrated as follows.

4.1. User Anonymity

In our protocol, the patient \( P_i \) uses a one-time random number as a key to wrap the patient and the clinician’s original ID in each communication. The alias ID of \( P_i \) is \( NID_{P_i} = ID_{P_i} \oplus R^{\text{new}}_P \), and the alias ID of \( D_j \) is \( ID_{D_j} = ID_{D_j} \oplus R^{\text{new}}_D \). Each time the ID is protected by a random value, it can be ensured that the ID is not derived and the malicious attacker cannot track a specific nickname. Additionally, the CA can check the ID list to verify that the user’s nickname is legal. Therefore, the mechanism we proposed can correctly implement user anonymity.

4.2. Stolen or Lost Smart Card Attack

When a user’s personal smart card is lost or stolen, a malicious attacker accessing the card can retrieve the information stored in the smart card. At this time, if the attacker cannot be detected, the attacker can use the smart card and the information in the smart card to simulate a remote legal user and pass the authentication.

In our proposed mechanism, only three parameters \( V_i, ST_i, \) and \( S_i \) are stored in the smart card. In our design, the ID is the main secret. If the user’s personal biosignal \( B_i \) cannot be obtained after obtaining \( V_i \), the ID cannot be obtained. \( ST_i \) and \( S_i \) use the updated \( R_P \), and CA private information to protect the ID. Therefore, it is impossible to analyze the biological characteristic \( B_i \) to analyze the ID. Therefore, when a malicious attacker steals the patient’s smart card, the attacker still cannot obtain the patient’s real identity and CA’s private key by verifying \( V_i \). Accordingly, there is little likelihood of decryption; consequently, we confirm that the proposed mechanism can prevent a data breach from theft or lost smart card attacks.

4.3. Man-In-The-Middle Attack

The person illegally retrieving messages during the communication period is called a man-in-the-middle attack. A middleman will modify or falsify content before it is delivered to the recipient. In our proposed protocol, the key communication messages are \( M_1, M_2, \) and \( M_3 \).
The message $M_1$ that the patient communicates with the CA is protected by the CA's private secret $T_{CA}$. If the attacker tampers with this message, after receiving the maliciously modified message, the CA's $R_{P_i}^{new}$ will be incorrect. In addition, when the $M_2$ message is modified by a malicious attack, the EPRS cannot resolve the real $R_{P_i}^{new}$ through its privacy token $EC$. The message $M_3$ transmitted from the CA to the doctor is protected by the doctor's secret value $DC_j$. If the message is modified by a malicious attack, the doctor is also unable to obtain the real $R_{P_i}^{new}$. Regardless of whether the content of $M_1$, $M_2$, or $M_3$ is maliciously modified, the final session key will be different, and a shared secret tunnel cannot be generated. Our proposed mechanism indeed resists man-in-the-middle attacks.

4.4. Replay Attack

Replay attacks can also be viewed as man-in-the-middle attacks. This type of attack does not need to modify or tamper with the communication information, but directly retransmits the original information. If the malicious attacker obtains the message sent from the user to the CA, the malicious attacker can directly replay the authentication without verifying the message. Generally, resolving this issue is offset by a timestamp. In our proposed mechanism, timestamps are replaced by dynamic parameters formed by one-time random numbers. If a malicious attacker replays the user’s messages $\{NID_{Pi}, NIDD_j, M_1\}$, sending them to the certificate authority again, s/he may obtain the message $M_4$ from the electronic medical record server. However, the malicious attacker cannot know the one-time random number $RP_{i,new}$ generated by the patient and will receive the wrong message to create the wrong session key. Hence, our proposed mechanism can defend against replay attacks.

4.5. Tamper Attack

Tamper attacks can also be viewed as man-in-the-middle attacks. In our design, if a malicious attacker tampers with the communication between CA and Pi, the CA will not be able to obtain the real $R_{P_i}^{new}$, because $M_1$ has been tampered with by a malicious attacker. When the CA uses the wrong $ID_{Pi}$, the CA is unable to obtain the real $ID_{Pi}$. $R_{P_i}^{new}$ is not actually $P_i$, because the message has been tampered with by a malicious attacker and the CA will reject the key agreement request. Additionally, when a malicious attacker maliciously modifies the message, $\{NID_{Pi}, NIDD_j, M_2\}$ is sent from the CA to the EPRS. If message $M_2$ has been maliciously modified so that EPRS checks the EPR database, the EPRS will not find the real $R_i$ using $ID_{Pi}$. As a result, the EPRS will stop the key agreement request. In short, the mechanism we propose can prevent tampering attacks.

4.6. Insider Privilege Attack

Internal privileged attacks mean that malicious attackers deliberately register as legitimate members, so they may obtain relevant authentication information. They impersonate other legitimate users, perform illegal secret authentication, and obtain the keys of certificate authorities. In our proposed protocols, the internal information available to each user will be protected in $V_i$, $ST_i$, and $S_i$. The secret tokens in $S_i$ are protected by the patient’s biosignal $H(B_i)$ and the one-time random number for updating $ST_i$ and $S_i$. Hence, if internal malicious attackers obtain their own $V_i$, $ST_i$, and $S_i$, they cannot know the biological characteristics of the pre-impersonator, and the secret information is updated randomly in each session, so impersonation verification cannot be performed. Additionally, during the patient registration phase, CA’s secret token is hash-encrypted by a one-way hash function. Even if an internally privileged malicious attacker can easily obtain $h(T_{CA})$, s/he cannot obtain the CA’s real secret token $T_{CA}$. Based on the one-way hash function and the security of biological characteristics, it shows that our proposed mechanism can resist internal privilege attacks.

4.7. Verify Table Stolen Attack

The verify table refers to the list used to confirm the relevant identity in the authentication server. A malicious person can invade the authentication server. A malicious insider can steal the verification form from the certificate authority through his own authority. Then, they obtain confidential
information of other users from the table to impersonate other legitimate identities. In our proposed protocols, the certificate authority only stores a list of legitimate user ID$_{Pi}$ and does not store other linkable messages for the user. It only provides the electronic medical record sharing stage, allowing the certificate authority to verify the user’s identity through the ID list. As such, our proposed mechanism will not allow attackers to perform other attacks even if the verification table is stolen.

4.8. Off-Line Password Guessing Attack

Generally, for the sake of saving communication costs, the system first verifies the client information and confirms that communication content is correct before communicating with the remote site. As such, malicious attackers often use brute force offline password guessing after obtaining the smart card. In our proposed protocols, patients do not use a password to register with a certification authority during the registration phase. In the medical record sharing stage, there is a patient’s personal biometric protection communication message. The biometric signal is unique for everyone, so a malicious attacker cannot obtain the secret $B_i$. Additionally, the CA checks the ID list and rejects the illegal ID$_{Pi}$, so offline attacks will not obtain the real ID. We confirm that the proposed mechanism can resist offline password guessing attacks.

4.9. Known Key Secrecy

Known key secrecy is often mistaken for perfect forward secrecy because malicious users may obtain the session key or any secret information through analysis and predict the next session key or member privacy. In our proposed protocol, the session key used by patients, the EPRS, and doctors in the communication is generated by a one-time random number $SK = h (R_{Pi, new} \oplus R_{EPRS} \oplus R_D)$, where the random number $R_{Pi, new}$ is generated by the patient randomly. The number $R_{EPRS}$ is generated by the electronic medical record server, and the random number $R_D$ is generated by the remote doctor. These one-time random numbers are irregular. Each parameter is randomly regenerated. Consequently, even if the attacker obtains any such confidential information in this session, the attacker cannot calculate the next session key, so the proposed protocol can provide known key secrecy.

5. Performance Comparisons

In this section, we compare recent electronic medical solutions by their computational and communication costs. Comparisons of Wen et al. [7], Li et al. [12], Jung et al. [13], Wu et al. [17], Lin et al. [28], Lee et al. [29], and our method are illustrated in Table 2, a performance comparison table, where $T_H$ denotes the execution time of the biometric-hash function operation, $T_h$ denotes the execution time of the one-way hash function operation, $T_{xor}$ denotes the execution time of the Exclusive-OR (short for XOR) operation, $T_M$ denotes the execution time of multiplication operations, and $T_{qr}$ denotes the execution time of computing a square root modulo $n$. $T_{xor}$, $T_h$, and $T_H$ belong to lightweight operations. According to the description of [15,23], the calculation of the XOR operation is negligible, and the $T_h \approx T_H$ time cost is similar.

| Schemes | Phases | Registration Phase | Login/Authentication Phase | Password Change Phase | Total Cost |
|---------|--------|--------------------|---------------------------|-----------------------|------------|
| Wen [7] | 4$T_h$ + 2$T_{xor}$ | (6$T_h$ + 2$T_{xor}$) + 1$T_qr$ + 1$T_M$ | 4$T_h$ + 2$T_{xor}$ | (14$T_h$ + 6$T_{xor}$) + 1$T_qr$ + 1$T_M$ |
| Lee [12] | 4$T_h$ + 2$T_{xor}$ | 12$T_h$ + 9$T_{xor}$ | 4$T_h$ + 3$T_{xor}$ | (20$T_h$ + 14$T_{xor}$) |
| Jung [13] | 3$T_h$ + 2$T_{xor}$ + 2$T_M$ | 13$T_h$ + 2$T_{xor}$ + 9$T_{xor}$ | 2$T_h$ + 2$T_{xor}$ | (24$T_h$ + 10$T_{xor}$) |
| Wu [17] | 2$T_h$ + 1$T_M$ | 10$T_h$ + 1$T_{xor}$ + 2$T_M$ | 1$T_h$ | (13$T_h$ + 2$T_{xor}$) + 2$T_M$ |
| Lin [28] | 1$T_h$ + 1$T_{xor}$ | 9$T_h$ + 26$T_{xor}$ + 4$T_M$ | N/A | (9$T_h$ + 26$T_{xor}$) + 4$T_M$ |
| Lee [29] | 6$T_h$ + 1$T_{xor}$ + 15$T_{xor}$ | 7$T_h$ + 1$T_{xor}$ + 13$T_{xor}$ | 8$T_h$ + 1$T_{xor}$ + 16$T_{xor}$ | (24$T_h$ + 44$T_{xor}$) |
| Proposed | 3$T_h$ + 1$T_M$ + 7$T_{xor}$ | 6$T_h$ + 1$T_M$ + 22$T_{xor}$ | 4$T_{xor}$ | (11$T_h$ + 33$T_{xor}$) |

$T_H$: Denotes execution time of biometric-hash function operation; $T_h$: Denotes execution time of one-way hash function operation; $T_{xor}$: Denotes execution time of XOR operation; $T_M$: Denotes execution time of multiplication operations; $T_{qr}$: Denotes execution time of computing a square root modulo $n$. $T_L$: Denotes execution time of total lightweight computing.
As lightweight computing has gradually become the main issue of encryption and decryption considerations, more and more articles employ it. As the computing time of lightweight computing is relatively short, the variation in the discrepancy was found to be negligible. Even though the computing time in our paper is slightly shorter than others, this is not our focus. We emphasize that we can still avoid various well-known malicious attacks under nearly differential calculation time, implement a genuine dynamic anonymous mechanism, and employ exclusive one-single-factor input, namely reasonable biometric features, to avoid long strings of memories and inputs.

In Figure 4 and Table 2, we mainly focus on the lightweight operation, which mainly includes XOR and hash operations. Though our proposed mechanism is not the best in XOR operation performance, our hash function operation performance is the best. In the case of total cost, the operation computation time of bitwise XOR has less hash function, so our proposed mechanism uses the least cost and has better performance. As mentioned in Table 2, our login phase computation time is \( (6T_b + 11T_H + 22T_{xor}) \approx (7T_b + 22T_{xor}) \) and the total computation time required for our proposed mechanism is \( (9T_b + 2T_H + 33T_{xor}) \approx (11T_b + 33T_{xor}) \) relative time.

![Figure 4. Performance comparison charts.](image)

Finally, a security comparison of the proposed solution with related research for different known attacks, user anonymous, man-in-the-middle attack, replay attack, stolen or loss smart card attack, tamper attack, insider attack, verify table stolen attack, off-line password guessing attack, and known key security are shown in Table 3. Based on the security analysis, we observe that the mechanism proposed in this study is to determine that it can resist known attacks, and the results are desirable.

| Trait       | Wen [7] | Lee [12] | Jung [13] | Wu [17] | Lin [28] | Lee [29] | Proposed |
|-------------|---------|----------|-----------|---------|----------|----------|----------|
| T1          | Yes     | Yes      | No        | Yes     | No       | Yes      | Yes      |
| T2          | Yes     | Yes      | Yes       | Yes     | Yes      | Yes      | Yes      |
| T3          | Yes     | Yes      | Yes       | Yes     | No       | Yes      | Yes      |
| T4          | Yes     | Yes      | Yes       | No      | Yes      | Yes      | Yes      |
| T5          | Yes     | No       | Yes       | Yes     | Yes      | Yes      | Yes      |
| T6          | Yes     | Yes      | Yes       | Yes     | Yes      | Yes      | Yes      |
| T7          | Yes     | Yes      | Yes       | No      | Yes      | Yes      | Yes      |
| T8          | Yes     | No       | Yes       | Yes     | Yes      | Yes      | Yes      |
| T9          | Yes     | Yes      | Yes       | Yes     | Yes      | Yes      | Yes      |
| T10         | No      | Yes      | Yes       | Yes     | No       | Yes      | Yes      |

T1: User anonymous. T2: Man-in-the-middle attack. T3: Replay attack. T4: Stolen or loss smart Card attack. T5: Tamper attack. T6: Insider attack. T7: Verify table stolen attack. T8: Off-line password guessing attack. T9: Known key security. T10: Lightweight computing cost.

6. Conclusions

Thanks to the Internet of Things (IoT) and advances in 5G communication, telehealth is expanding and shows no signs of slowing down. It provides patients with convenience and easy access to
healthcare services across space and time. However, the continuous real-time transmission of health information over networks also exposes the patients’ privacy copyright data to the risk of a data breach. In this paper, we propose a personal electronic medical record copyright storage and transmission protection, authorization, and sharing management mechanism. The patient’s electronic medical record is protected by the patient’s private key and our protocol provides a real anonymous design for the patient possessing absolute autonomy over their privacy. Our mechanism ensures that only the certificate authority center knows the real name of the patient and the corresponding transmission location, and no one else knows it; therefore, it may achieve true anonymity. The secret token is updated every time, i.e., the pseudonym is different every time, so patients can individually control their own electronic medical records and use them safely in telehealth environments. In this study, each update of the cloud medical records is patient-led and performed in a secure tunnel. Our proposed mechanism has strict security controls over either in the telehealth process or in the storage and renewal process. As such, our proposed security analysis guarantees comprehensive security and anonymity protection during telehealth data and privacy transmission.

Author Contributions: Y.-J.K. was the principal researcher of this study. She contributed substantially to the conception, the design, the methodology, the analysis, and the interpretation of this study. She drafted the original manuscript, edited the critical revision of the manuscript, and proofread the revised paper. She performed the final approval of the manuscript to be submitted. J.-C.S. performed project administration. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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

References

1. Nguyen, L.; Bellucci, E. Electronic health records implementation: An evaluation of information system impact and contingency factors. *Int. J. Med. Inf.* 2014, 83, 779–796. [CrossRef] [PubMed]
2. Pomputius, A.F. A Review of Two-Factor Authentication: Suggested Security Effort Moves to Mandatory. *Med. Ref. Serv. Q.* 2018, 37, 397–402. [CrossRef] [PubMed]
3. Tan, Z. A user anonymity preserving three-factor authentication scheme for telecare medicine information systems. *J. Med. Syst.* 2014, 38, 16. [CrossRef] [PubMed]
4. Li, X.; Niu, J.; Kumari, S.; Wu, F.; Sangaiah, A.K.; Choo, K.-K.R. A three-factor anonymous authentication scheme for wireless sensor networks in internet of things environments. *J. Netw. Comput. Appl.* 2018, 103, 194–204. [CrossRef]
5. Healy, J.; Chambers, D. Approximate $k$-Mer Matching Using Fuzzy Hash Maps. *IEEE ACM Trans. Comput. Biol. Bioinform.* 2014, 11, 258–264.
6. Barman, H.; Hubert, P.H.; Chattopadhyay, S.; Samanta, D. A Secure Authentication Protocol for Multi-Server-Based E-Healthcare Using a Fuzzy Commitment Scheme. *IEEE Access* 2019, 7, 12557–12574. [CrossRef]
7. Wen, F. A more secure anonymous user authentication scheme for the integrated EPR information system. *J. Med. Syst.* 2014, 38, 42. [CrossRef]
8. Chuang, M.-C.; Chen, M.C. An anonymous multi-server authenticated key agreement scheme based on trust computing using smart cards and biometrics. *Expert Syst. Appl.* 2014, 41, 1411–1418. [CrossRef]
9. Lee, J.K.; Ryu, S.R.; Yoo, K.Y. Fingerprint-based remote user authentication scheme using smart cards. *Electron. Lett.* 2002, 38, 554–555. [CrossRef]
10. Messerges, T.S.; Dabbish, E.A.; Sloan, R.H. Examining smart-card security under the threat of power analysis attacks. *IEEE Trans. Comput.* 2002, 51, 541–552. [CrossRef]
11. Allam, A.M.; Ibrahim, I.I.; Ali, I.A.; Elsawy, A.E.H. Efficient zero-knowledge identification scheme with secret key exchange. In Proceedings of the 2003 46th Midwest Symposium on Circuits and Systems, Cairo, Egypt, 27–30 December 2003; pp. 516–519.
12. Lee, T.F.; Chang, I.-P.; Lin, T.-H.; Wang, C.-C. A Secure and Efficient Password-Based User Authentication Scheme Using Smart Cards for the Integrated EPR Information System. *J. Med. Syst.* 2013, 37, 9941. [CrossRef]
13. Jung, J.; Kang, D.; Lee, D.; Won, D. An improved and secure anonymous biometric-based user authentication with key agreement scheme for the integrated EPR information system. *PLoS ONE* 2017, 12, e0169414. [CrossRef]

14. Jung, J.; Moon, J.; Lee, D.; Won, D. Efficient and Security Enhanced Anonymous Authentication with Key Agreement Scheme in Wireless Sensor Networks. *Sensors* 2017, 17, 644. [CrossRef]

15. Wazid, M.; Das, A.K.; Kumari, S.; Li, X.; Wu, F. Design of an efficient and provably secure anonymity preserving three-factor user authentication and key agreement scheme for TMIS. *Secur. Commun. Netw.* 2016, 9, 1983–2001. [CrossRef]

16. Lin, C.H.; Lai, Y.Y. A flexible biometrics remote user authentication scheme. *Comput. Stand. Interfaces* 2004, 27, 19–23. [CrossRef]

17. Wu, Z.-Y.; Chung, Y.; Lai, F.; Chen, T.-S. A password-based user authentication scheme for the integrated EPR information system. *J. Med. Syst.* 2012, 36, 631–638. [CrossRef] [PubMed]

18. Rho, M.J.; Choi, I.Y.; Lee, J. Predictive factors of Telehealth service acceptance and behavioral intention of physicians. *Int. J. Med. Inform.* 2014, 83, 559–571. [CrossRef] [PubMed]

19. Zhang, L.; Zhu, S.; Tang, S. Privacy Protection for Telecare Medicine Information Systems Using a Chaotic Map-Based Three-Factor Authenticated Key Agreement Scheme. *IEEE J. Biomed. Health Inform.* 2017, 21, 465–475. [CrossRef]

20. Li, S.; Wang, C.; Lu, W.; Lin, Y.; Yen, D. Design and implementation of a telecare information platform. *J. Med. Syst.* 2012, 36, 1629–1650. [CrossRef]

21. Perednia, D. Telehealth technology and clinical applications. *JAMA J. Am. Med. Assoc.* 1995, 273, 483–488. [CrossRef]

22. Mair, F.S.; Haycox, A.; May, C.; Williams, T. A review of Telehealth cost-effectiveness studies. *J. Telehealth Telecare* 2000, 6, 38–40. [CrossRef] [PubMed]

23. Li, X.; Zheng, Z.; Zhang, X. A Secure Authentication and Key Agreement Protocol for Telecare Medicine Information System. In Proceedings of the 2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies, Cambridge, UK, 9–11 September 2015; pp. 275–281.

24. Chen, H.; Ge, L.; Xie, L. A User Authentication Scheme Based on Elliptic Curves Cryptography for Wireless Ad Hoc Networks. *Sensors* 2015, 15, 17057–17075. [CrossRef] [PubMed]

25. Choi, Y.; Lee, D.; Kim, J.; Jung, J.; Nam, J.; Won, D. Security Enhanced User Authentication Protocol for Wireless Sensor Networks Using Elliptic Curves Cryptography. *Sensors* 2014, 14, 10081–10106. [CrossRef] [PubMed]

26. Esposito, C.; Ciampi, M.; Pietro, G. An event-based notification approach for the delivery of patient medical information. *Inf. Syst.* 2014, 39, 22–44. [CrossRef]

27. Lin, J.H. An Electronic Medical Record Authorized Sharing Mechanism for Telemedicine. Master’s Thesis, National Chin-Yi University of Technology, Taiwan, July 2019.

28. Lin, T.H.; Tsung, C.K.; Lee, T.F.; Wang, Z.B. Round-Efficient Authenticated Key Agreement Scheme Based on Extended Chaotic Maps for Group Cloud Meeting. *Sensors* 2017, 17, 2793. [CrossRef]

29. Lee, T.F.; Diao, Y.-Y.; Hsieh, Y.-P. A ticket-based multi-server biometric authentication scheme using extended chaotic maps for telecare medical information systems. *Multimed. Tools Appl.* 2019, 78, 31649–31672. [CrossRef]

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).