applied sciences

MDPI

Article

A Provable and Secure Patient Electronic Health Record Fair Exchange Scheme for Health Information Systems

Ming-Te Chen † **and Tsung-Hung Lin** *,†

Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 41170, Taiwan; mtchen@ncut.edu.tw

* Correspondence: duke@ncut.edu.tw

† These authors contributed equally to this work.

Abstract: In recent years, several hospitals have begun using health information systems to maintain electronic health records (EHRs) for each patient. Traditionally, when a patient visits a new hospital for the first time, the hospital's help desk asks them to fill in relevant personal information on a piece of paper and verifies their identity on the spot. This patient will find that many of her personal electronic records are in many hospital's health information systems that she visited in the past, and each EHR in these hospital's information systems cannot be accessed or shared between these hospitals. This is inconvenient because this patient will again have to provide their personal information. This is time-consuming and not practical. Therefore, in this paper, we propose a practical and provable patient EHR fair exchange scheme for each patient. In this scheme, each patient can securely delegate the information system of a current hospital to a hospital certification authority (HCA) to apply migration evidence that can be used to transfer their EHR to another hospital. The delegated system can also establish a session key with other hospital systems for later data transmission, and each patient can protect their anonymity with the help of the HCA. Additionally, we also provide formal security proofs for forward secrecy and functional comparisons with other schemes.

Keywords: electronic health records; fair exchange; forward secrecy

Citation: Chen, M.-T.; Lin, T.-H. A Provable and Secure Patient Electronic Health Record Fair Exchange Scheme for Health Information Systems. *Appl. Sci.* **2021**, *11*, 2401. https://doi.org/10.3390/app11052401

Academic Editor: Federico Divina

Received: 25 January 2021
Accepted: 2 March 2021
Published: 8 March 2021

1. Introduction

In recent years, many research topics have arisen to make human life more convenient. An electronic health record (EHR) is an integrated personal medical record in health information systems. Many countries implement their own health information systems to help manage each patient's activities and keep track of their health. We can imagine a scenario in which a patient (let us call her Alice) plans to go to a new hospital and sees a doctor. In this situation, she may have to fill in her personal medical information another time when she attends a new hospital. In addition, if her doctor needs to know her medical treatment history from other hospitals, how she provides these records securely to her doctor needs to be considered. These problems are especially urgent. Our proposed scheme ensures the ease and security of data access and migration. Our approach proposes a practical and provable patient EHR fair exchange scheme with session key establishment for health information systems. Patients cannot only delegate the migration of their personal EHR to a desired hospital system from their current hospital health information system but also protect their privacy. Our mechanism provides secure data storage and the secure transfer of authorized information to a designated location. This study has two limitations. First, we assume that each patient's EHR record is well defined and appropriate for each healthcare facility. The process of electronic health information record transmission at each hospital provider is easily done by implementing our proposed scheme for secure encrypted transmission without considering issues such as different forms or file names or a lack of formatting details. Second, each facility will transfer or link the patient's EHR to

other facilities through patient consent or under a national policy when the patient requires better care at those facilities.

Summarizing all problems, we propose a high-level practical and provable patient EHR fair exchange scheme with key agreement for health information systems. Not only could a patient delegate the health information systems of the current hospital to migrate their personal EHR to the desired hospital system, but they can also keep their privacy. Our mechanism provides data storage and the secure transfer of authorized information to designated locations. What information can be authorized is beyond the scope of this study to determine. For example, whether COVID-19 patient privacy concerning patients' names, identities, and genetic sequences can be transmitted to different hospitals is beyond this study's scope. The mechanism presented here could guarantee data transfer and storage safely and securely. What is more, our scheme also provides a formal security proof in a random oracle model under chosen-ciphertext security.

This paper is organized as follows. Section 2 introduces related works. Section 3 deploys security definitions. Section 4 shows our proposed method. Section 5 describes our security analysis. Section 6 provides a security proof. Finally, Section 7 presents our conclusions.

2. Related Works

In this section, we surveyed some articles [1–4]. In [1], the author only mentioned how EHRs are used and managed. The author also talked about the EHR format that followed the definition of HL7 [5] and that performed well-known protocols to encode each patient's EHR from TCP/IP, MIME, HTTP(S), and SOAP.

In [2], the authors discussed several security requirements, such as EHR storage security, malicious code prevention, protected access right management, and other aspects to protect the health information system. However, they did not provide a practical scheme that would allow a patient to migrate their EHR to a health information system. We can imagine a scenario where a hospital only adopts the above simple protocols to develop its own health information system without any security mechanism. Additionally, it is not feasible for each patient to perform their own EHR exchange under this scheme.

On the other hand, in [5], the authors suggested that each patient's health records (or files) could be portably stored on a flash disk. This idea is appealing but is currently difficult to implement. There are many security issues to be handled, including portable device security and patient medical file access rights. However, more security mechanisms are needed to solve these kinds of security issues, which are beyond the scope of this research.

In addition, various patient authentication schemes of e-health systems have been proposed [6–10]. In [6,7], the schemes suffered a user impersonation attack and did not offer session key establishment with a formal security proof. The authors in [8–10] did not provide session key establishment with a forward secrecy proof. In [11,12], the authors each proposed a framework with a patient-centric access right in a blockchain environment. However, they did not provide a practical mechanism for each patient to perform EHR migration exchange securely.

Additionally, many studies are now examining the importance of personal privacy and data authorization. For example, the prevalence of COVID-19 has made many patients reluctant to disclose information about their infection, but government healthcare units want to control the trajectory of tracking these patients. A method of providing improvements in these mechanisms is the main motivation and purpose of our study. Therefore, in this paper, we emphasize providing a secure, simple, and complete mechanism for authorizing data transfer during personal information migration and demonstrate that our approach is secure and effective in practice through a professional information security authentication model.

Hence, we summarize and list here seven kinds of security attack when a patient attempts to migrate their personal information data through a traditional authentication model:

- Replay Attack Resistance: A malicious attacker intercepts the parameters used in the mutual authentication transaction successfully. They can then forward these parameters again to impersonate one party to communicate with other parties during the mutual authentication transaction and vice versa.
- Resist User Impersonation: A malicious attacker impersonates some party by replaying the intercepted signatures or random variables to other parties engaged in the mutual authentication transaction.
- Mutual Authentication: Without mutual authentication with other e-health systems, an attacker can pretend as a fake system to let other patients register and login. Then, patient's EHRs information cannot be stored securely in this storage location.
- Data Security Problem: An attacker intercepts the parameters, including ciphertext, successfully during the mutual authentication transaction, and they may then decrypt the intercepted ciphertext by adopting these intercepted parameters.
- Session Key Establishment with Forward Secrecy: Each party communicates a temporary symmetric key for data transmission after performing mutual authentication successfully. If the session key is easy to guess or derive by an attacker successfully without forward secrecy, then this attacker can derive the used session keys before the next mutual authentication phase.
- EHR Fair Exchange Problem: If there is packet loss or data loss during the EHR migration transaction, a patient's EHR can be lost when they attended a desired hospital. At this time, without their personal EHR, the e-health system of this hospital can delay their medical treatment in this situation.
- Patient's Anonymity Protection: During the EHR migration transaction, if a patient's EHR identity information is not protected well, then it could be exposed and intercepted by an attacker. Additionally, the patient's EHR information may be misused by an attacker further.

Our contribution is to offer an efficient provable and practical patient EHR fair exchange scheme so that each patient can migrate their personal EHR securely from one hospital to another and provides solutions to the above seven problems. We designed a secure patient EHR exchange protocol that can be integrated into the e-health information system of each hospital. The proposed scheme could also guarantee convenience, rapidity, and integrity. We constructed a high-level practical and provable patient EHR fair exchange scheme with key agreement for the health information system. A patient could not only delegate the current hospital's health information systems to migrate their personal EHR to the desired hospital system, but also keep their privacy. Additionally, our scheme demonstrates a formal security proof with light-weight computation for both authentication parties.

3. The Proposed Scheme

Our proposed scheme contains three stages: the migration registration phase, the EHR migration phase, and the data recovery phase.

3.1. Preliminary

In this subsection, we provide some definitions in our proposed scheme.

- n: A large prime number that forms a finite primes field with an order less than n.
- l: A security parameter that defines the hashed messages' length.
- V: The current medical organization.
- W: The patient's desired medical organization.
- U: A patient making an authentication request with a server V in a health information system.
- S: A server that accepts the registration of the patient request, the login request, and the password modification request in the health information system of hospital V.
- UID_i: A patient's real identity computed from social security numbers, where $i \in \{U\}$ in the certification.

- ID_i: A registration local identity that can link the UID_i of user $i \in \{U\}$.
- H_1, H_2: Two secure hash functions that each maps $Z_n^* \rightarrow \{0,1\}^l$ with collision-resistance and outputs the same l-bits hash strings.
- pw_U: A initial password that is chosen by a server V when a patient U has remotely registered on the server for the first time.
- E_{k_i}: A symmetric key encryption function for the party i under the symmetric key k_i, where $i \in \{U, V\}$.
- D_{k_i}: A symmetric key decryption function for the party i under the symmetric key k_i, where $i \in \{U, V\}$.
- ASE_{pk_i}: An asymmetric key encryption function for the party i, where $i \in \{U, V\}$.
- ASD_{sk_i}: An asymmetric key decryption function for the party i, where $i \in \{U, V\}$.
- EHR_i: A patient electronic health record (EHR) in one hospital organization, where $i \in U$.
- Bio_i: A biometric information value that is chosen uniformly from the party i, where $i \in \{U\}$.
- $Date$: A patient EHR migration limitation date period.
- $Cert_i$: A migration certification of the party i with ID_i registration and public keys for this ID_i, where $i \in \{U, V\}$.
- HCA: A hospital certification authority (HCA) that helps the patient to generate the patient migration permission signature to another hospital or medical center in the public key infrastructure (PKI).
- $Agree_{i->j}$: A patient-delegated EHR migration agreement document whereby the patient agrees that its own patient files can migrate from current hospital i to the desired one j, where $i, j \in \{V, W\}$.

3.2. The Migration Registration Phase

Before starting this phase, a patient (U) forwards (UID_U, ID_U) to a hospital certification authority (HCA) for migration certification registration with a secure channel. After receiving this identity (UID_U, ID_U), the HCA keeps this link information and generates $Cert_i$ certification with ID_i for EHR migration and forwards this $Cert_i$ to the patient U.

When U performs this phase with the server V of the current hospital, U forwards a patient migration registration request to a server (V). After receiving this request, the server V forwards this request to the HCA to help U obtain the permission signature from HCA. V first prepares two hash functions: one is H_1, and the other is H_2, where $H_1 : Z_n^* \rightarrow \{0,1\}^l$ and $H_2 : Z_n^* \rightarrow \{0,1\}^l$.

- First, U has prepared their biometric value Bio_U and computed a random value r_{B_U} with a random number $r''_U \in Z_n^*$, where $r_{B_U} = r''_U \oplus H_1(Bio_U)$. After the above is computed successfully, they forward their identity ID_U, which is computed from $H_1(r''_U||UID_U)$ and encrypted via real identity cipher-text $C_{HCA} = AE_{pk_{HCA}}(r''_U, Bio_U, UID_U, ID_U, Cert_U, EHR_U \oplus r''_U)$ with public key pk_U and their certificate $cert_U$ with their biometric information r_{B_U}, to the server V. It also generates the applicant-delegated migration signature S_U, where the signature S_U is to be the $Sig_U(Agree_{V->W}, UID_U, ID_U, EHR_U \oplus r''_U)$ with the registration identity ID_U, real identity UID_U, and EHR migration delegated agreement $Agree_{V->W}$ with final ciper-text $EHR_U \oplus r''_U$. U then prepares the random number r''_V, where $r''_V = r_V \oplus r_{B_U}$, and forwards these files to the server V with $(r''_V, C_{HCA}, S_U, Cert_U, Agree_{V->W})$.
- After V receives these messages, it can check the S_U with the above messages and forward the C_{HCA} to the HCA. When HCA has received this message with S_U, it can decrypt C_{HCA} to obtain all parameters. First, it can fetch the random value $r''_V \oplus r_{B_U} = r_V$ and verify the signature S_U with other parameters. If they are valid, it saves these files for data recovery usage. It then generates S_{HCA}, which is $Sig_{HCA}(S_U, Date, ID_U)$. Finally, it returns $(S_{HCA}, H_1((r_V + 1)||r_{HCA}), (r_V + 1) \oplus r_{HCA}, Date, ID_U, S_U)$ to the server V.

- V receives this message tuple, where one is $(S_{HCA}, H_1((r_V+1)|r_{HCA}))$ and the other is $((r_V+1) \oplus r_{HCA}, Date, ID_U, S_U)$, and it can verify the signature S_{HCA} with the above parameters and compute $r_{HCA} = (r_V+1) \oplus (r_V+1) \oplus r_{HCA}$. When the above messages are valid, V returns $H_1(r_{HCA}+1)$ and forwards it back to the server HCA. In addition, it also generates a signature $Sig_V(S_{HCA}, S_U, Agree_{V->W})$ as the receipt S_V and finishes this phase after forwarding S_V and S_{HCA}. We demonstrate in the Figure 1.

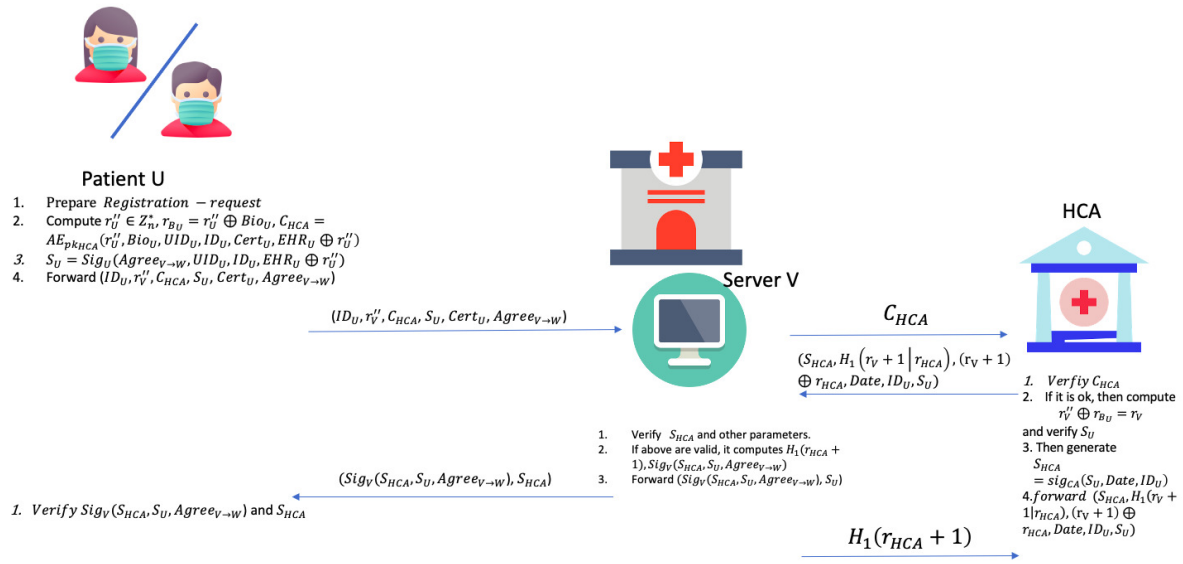

Figure 1. The migration registration phase.

3.3. The EHR Migration Phase

In this phase, the server V will behave according to the delegated agreement file $Agree_{V->W}$ with the signature S_U, and it prepares these messages as follows.

- First, it selects a random r_V^* and computes $C_V = AE_{pk_W}(r_V^*)$. It then forms the message tuple $(C_V, S_U, S_{HCA}, EHR_U \oplus r_U'', Date)$ for the server of the desired migration hospital W. When the system of W has received this migration agreement from V, it verifies this message tuple and decrypts C_V to obtain the challenge random number. If all signatures and parameters are valid, it generates the response random number $r_W^* \oplus H_2(r_V^*+1)$ and returns it to the server V.
- When V receives $r_W^* \oplus H_2(r_V^*+1)$, it decrypts it with r_V^*+1. If it is valid, it can obtain r_W^* and computes the response random number $H_2((r_W^*+1) \oplus r_U'')$, $(r_W^*+1) \oplus r_U''$ for the server of hospital W. Finally, W receives $H_2(r_W^*+1 \oplus r_U'')$, $(r_W^*+1) \oplus r_U''$ from V. It also verifies the message to check if it was returned by the real V. If true, it decrypts $(r_W^*+1) \oplus r_U''$ to obtain r_U'', computes the session key $ssk_{V,W} = H_1((r_W^*+1)||(r_V^*+1))$, and decrypts $EHR_U \oplus r_U''$ to fetch the patient's EHR file EHR_U with the random r_U''.
- After performing mutual authentication successfully, V can also use the session key $ssk_{V,W}$ to generate an cihper-text, such as $E_{ssk_{V,W}}(EHR_U)$, for the rest of the data transmission of the patient U's EHR. Figure 2 shows the scenario of this phase.

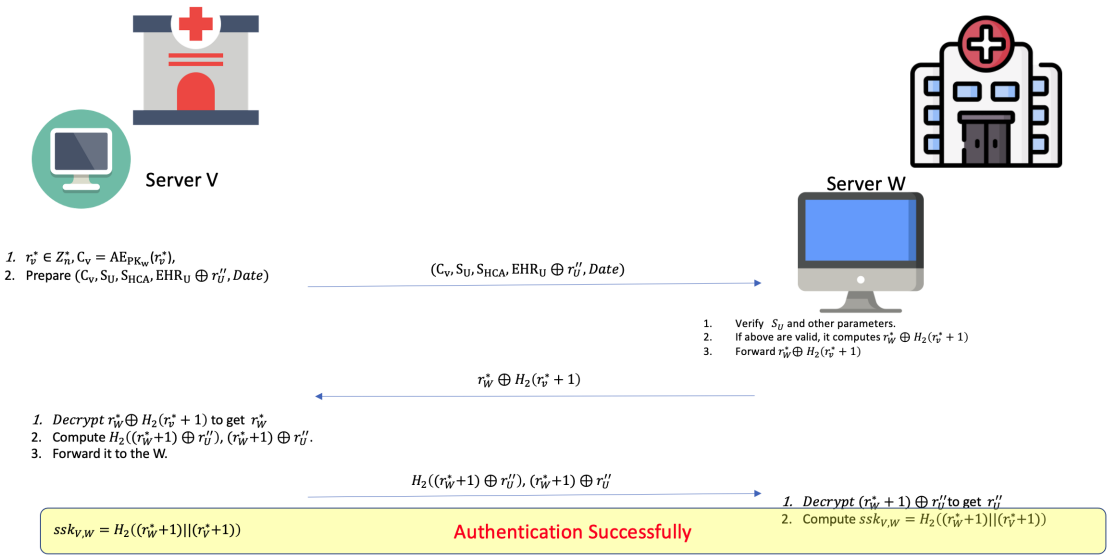

Figure 2. The EHR migration phase.

3.4. The Data Recovery Phase

In this phase, if there is some network packet loss or EHR data loss of the patient U after they have performed the EHR migration phase, then the e-health system of the hospital W cannot obtain the full U's personal EHR, so U can ask the HCA to deal with this situation.

- First, when a patient migrating to a hospital W that does not receive the EHR_U from the server V after querying the system of hospital W, then U can ask HCA to resolve the situation with some evidence $Sig_V(S_{HCA}, S_U, Agree_{V->W})$ with signatures S_{HCA}, S_U and $Agree_{V->W}$. After verifying the above signatures successfully, HCA can fetch the $EHR_U \oplus r_U''$ to the server of hospital W with $AE_{pk_W}(r_U'')$.
- W can then decrypt $AE_{pk_W}(r_U'')$ to obtain r_U'' and fetches the patient's EHR file EHR_U from the above $EHR_U \oplus r_U''$ by applying $\oplus$ with r_U''. At this time, this situation is solved with HCA's help if needed. This phase's scenario shows in the Figure 3.

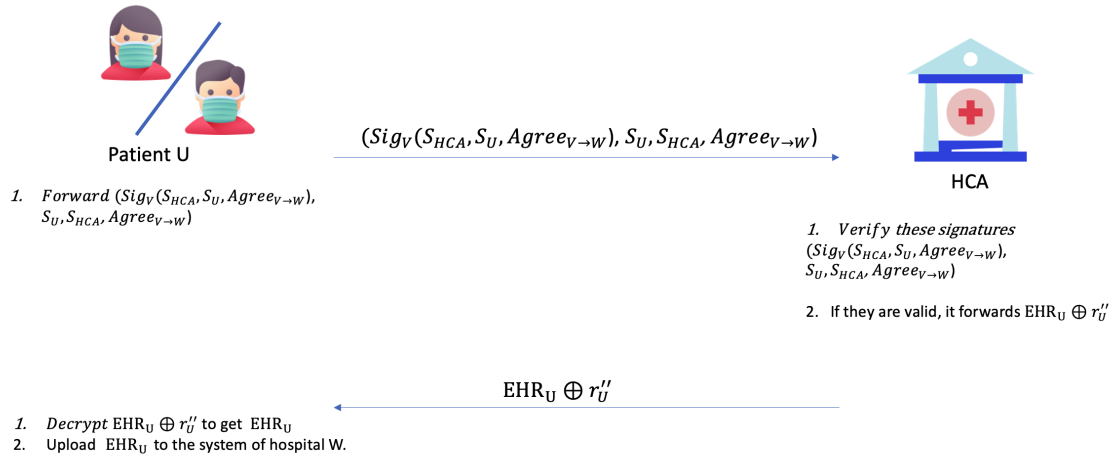

Figure 3. The data recovery phase.

4. Security Assumptions

4.1. Secure Digital Signature

In this scheme, we define a secure digital signature. In the beginning, we have that $Sig(\cdot)$ is a signature generation function that inputs a message m with a signer's secret key sk_i and outputs a signature S_i. We also assert this signature function is based on the RSA factoring hard problem or the discrete logarithm problem. We can then input a signature such as S_i with the signer's public key pk_i into the verification function $Ver(\cdot)$ and see what the output is. If the output is 1, then we can confirm the signature S_i is valid and signed by the signer i. In this scheme, we also assumed that the signature building block is under the RSA problem. If there is an attacker, we assume it as $\mathcal{F}^*$. If $\mathcal{F}^*$ can make a forged $l+1$ signature called $S'_{i,j+1}$ of some user $i \in \{U, V, HCA\}$ in at most $l$signature queries, and this signature can pass the verification $Ver(pk_i, S'_{i,j+1})$ successfully with non-negligible probability ε, then $\mathcal{F}^*$ can be used to break the RSA factoring problem. Thus,

$$Pr[S'_{i,j+1} \longleftarrow \mathcal{F}^*(Sig(sk_i, \cdot), Ver(pk_i, \cdot), i \in \{U, V, HCA\}) | Ver(S'_{i,j+1}) = 1] \geq \varepsilon. \quad (1)$$

4.2. Unforgeability

In this scheme, we define the secure digital signature scheme in the above. First, we define an attacker $\mathcal{F}^*$, whose ability is to forge a signature that can be verified successfully through the $Ver(\cdot)$ verification function with non-negligible probability ε'. We also define a simulator $\mathcal{D}$ that adopts $\mathcal{F}^*$'s ability to break the underlying hard problem (such as the RSA factoring problem) in the above secure signature scheme. After $\mathcal{D}$ is given the environment parameters $G(\cdot)$, it can start the protocol simulation with $\mathcal{F}^*$. $\mathcal{F}^*$ can make the signature queries to the $\mathcal{D}$. $\mathcal{D}$ will also output the signature back according to the received input m from $\mathcal{F}^*$ on some user i. After this simulation, if $\mathcal{F}^*$ generates a forged signature $S'_{i,j+1}$, the verification result of $S'_{i,j+1}$ is valid. We then have

$$\begin{aligned} &Pr[\mathcal{D}^{\mathcal{F}^* \longrightarrow S'_{i,j+1}} | \text{Use } S'_{i,j+1} \text{ to solve the RSA factoring problem}] \geq \\ &Pr[S'_{i,j+1} \longleftarrow \mathcal{F}^*(Sig(sk_i, \cdot), Ver(pk_i, \cdot), i \in \{U, V, HCA\}) | Ver(pk_i, S'_{i,j+1}) = 1] \geq \varepsilon'. \end{aligned} \quad (2)$$

In fact, if there is no attack $\mathcal{F}^*$ that can make a forged signature pass the verification successfully with non-negligible probability ε, then we cannot use $\mathcal{F}^*$ to solve the RSA factoring problem with non-negligible ε' probability.

Lemma 1 (Unforgeability)**.** *First, we define Sig, which is a secure digital signature function and equips two secure hash functions, H_1 and H_2, which can be replaced with two random oracles functions RO_1 and RO_2. In our proposed EHR scheme, we also define our proposed EHR scheme with unforgeability (Unf), which satisfies the following situations. In other words, if Sig is (t', ε') and unforgeable, then*

$$Adv^{Unf}_{\mathcal{F}^*, Sig^{H_1, H_2, RO_1, RO_2}}(\theta, t') \leq \frac{1}{2 \cdot I^3 \cdot q_s} + \varepsilon', \quad (3)$$

where t' is the maximum total experiment time, including an adversary execution time, I is an upper bound on the number of parties, at most signature oracle q_s, and ε' is taken over the coin flip of our EHR scheme.

4.3. Indistinguishability

We define an attacker $\mathcal{A}$ on the experiment **EXP** of our symmetric encryption/decryption functions (**SE**), which is a game controlled by the simulator $\mathcal{S}$. We also define two pseudo-random hash functions (ω_1 and ω_2), which are satisfied with the property we call "indistinguishability" (**Ind**), due to which the attacker $\mathcal{A}$ can make a hash query to ω_1 and ω_2 on the message M', which is chosen by $\mathcal{A}$. These functions act as real functions as our hash functions (H_1 and H_2), where $i \in \{U, V\}$. The simulator also can switch this function pair

to respond to each query made by $\mathcal{A}$ during the simulation rounds of the above experiment. Finally, the simulator $\mathcal{S}$ is given a challenge message target M chosen by the $\mathcal{A}$.

At this time, $\mathcal{S}$ makes a coin flip on b. If $b = 0$, $\mathcal{S}$ randomly chooses (ω_1, ω_2) to generate the hashed value of M and return it to $\mathcal{A}$. Otherwise, $\mathcal{S}$ forwards M to (H_1, H_2) to ask for the hash value. $\mathcal{A}$'s goal is to guess correctly the hashed value that is from (ω_1, ω_2) or (H_1, H_2) with non-negligibility probability.

Lemma 2 (Indistinguishability)**.** *In this lemma, our symmetric encryption/decryption functions satisfy the indistinguishably property if there is no attacker $\mathcal{A}$ that can guess the hashed value from the chosen (M) with more than $\frac{1}{2}$ with negligible probability ε^* under the t^* polynomial time bound. That is,*

$$|Pr[b' \longleftarrow \mathcal{F}^{(\omega_1,\omega_2,H_1,H_2)}(M)|b = b'] - \frac{1}{2}| \leq \varepsilon.$$

Therefore, we concluded that

$$Adv^{Ind}_{\mathcal{A},\boldsymbol{SE}}(\theta, t^*) \leq \frac{1}{2} + \varepsilon^*.$$

4.4. Indistinguishable-Chosen Cipher-Text Attack (Ind-CCA)

In this scheme, we define our proposed asymmetric encryption/decryption function (**ASE**), which satisfies the semantic security in the following definitions.

First, we define an attacker $\mathcal{A}$ that can ask encryption/decryption queries in our scheme, respectively. However, the attacker $\mathcal{A}$ can also make an encryption query to the chosen message that we define as M'. The attacker $\mathcal{A}$ can then also make a decryption query to the decryption oracle, whose task is to decrypt the cipher-text sent $\mathcal{A}$. Next, we define *Game*, which is the simulation of our proposed scheme that can equip many different oracles, and oracles can answer back to the adversary depending on the attacker's input messages. We also define some oracles, such as the encryption oracle $AE_{pk_T}(\cdot, \theta)$ with the security parameter θ. This encryption oracle can generate the ciphertext according to the received input M_b, where $b \in \{0, 1\}$. In addition, we also model the decryption oracle that receives the cipher-text C from the attacker $\mathcal{A}$ and returns the final decrypted message M to the attacker $\mathcal{A}$. In the following, we consider two situations involving $\mathcal{A}$.

Phase 1: In this phase, the attacker $\mathcal{A}$ can make the decryption and encryption queries on a chosen message (call it M'). I.e., if $\mathcal{A}$ makes an encryption query on the input message M', then $C' \longleftarrow AE_{pk_T}(M', \theta)$ returns to $\mathcal{A}$. At this time, $\mathcal{A}$ can also make the decryption query on cipher-text C', and the simulator will then forward this C' to the decryption oracle and return the final message M' back to the $\mathcal{A}$. Additionally, $\mathcal{A}$ can also make other kinds of queries, such as a hash query to the hash oracles.

Challenge: In this phase, if $\mathcal{A}$ has performed training on the above encryption/decryption query many times, then, in the following challenge phase, the attacker $\mathcal{A}$ will choose a challenge message pair (M_0^*, M_1^*) for the simulator for game playing. The simulator then will toss the coin on b after it receives this message pair. If the final output b is 1, then we can have $C^* \longleftarrow AE_{pk_T}(M_b^*, \theta)$. Otherwise, we have $C^* \longleftarrow AE_{pk_T}(M_{1-b}^*, \theta)$. After the attacker $\mathcal{A}$ has asked the cipher-text on the chosen target messages (M_0^*, M_1^*), the only restriction is that the $\mathcal{A}$ cannot ask the decryption oracle on the target message (M_0^*, M_1^*) with the input cipher-text C^*. This query can make the simulation fail due to the simulator cannot be able to tell the answer of cipher-text C^*. Except in the above query, $\mathcal{A}$ can make other kinds of queries on different messages.

Lemma 3. *In this lemma, we model the above actions as the game simulation steps, which we played with the attacker $\mathcal{A}$.*

$Game_{\mathcal{A},ASE}^{Ind-CCA-b}(\theta)$
Phase 1.
$T \in \{U, V\}, \{M_0, M_1\} \longleftarrow \mathcal{A}^{ASE_{pk_T}(\cdot,\theta), ASD_{sk_T}(\cdot,\theta), H_1(\cdot), H_2(\cdot)}$
Challenge Phase.
$b \in \{0,1\}, C^* \longleftarrow ASE_{pk_T}(M_b^*, \theta)$,
$b' \longleftarrow \mathcal{A}^{ASE_{pk_T}(\cdot,\theta)}(C^*, M_0^*, M_1^*)$
Return b'.

The advantage function of the adversary that $\mathcal{A}_{ASE}^{Ind-CCA}(\cdot,\theta)$ is defined as $Adv_{\mathcal{A},ASE}^{Ind-CCA}(\theta) = |Pr[Game_{\mathcal{A},ASE}^{Ind-CCA-1}(\theta) = 1] - Pr[Game_{\mathcal{A},ASE}^{Ind-CCA-0}(\theta) = 1]| < \frac{1}{2}|Pr[Game_{\mathcal{A},ASE}^{Ind-CCA-1}(\theta) = 1]| \leq \varepsilon'$.

4.5. Partner Function

In this definition, we define the partner function. We assume that there is an instance Π_i^k whose action is the same as player i in the k-th session, where $i, j \in \{U, V\}$ and $k \in N$, where N is the number for total players. Let the partner function be the instance of player j (call it $\Pi_j^{k'}$) in the k'-th session, where $i, j \in \{U, V\}$ and $k' \in N$. At this time, the instances Π_i^k and $\Pi_j^{k'}$ believe that each side is the real player $i, j \in \{U, V\}$ in the $k, k' \in N$ session, respectively. At this time, we can say that two instances Π_i^k and $\Pi_j^{k'}$ are partnered if the following statements are true:

1. Π_i^k's session identity is the same as the session identity of $\Pi_j^{k'}$.
2. p_i is the partner of $\Pi_j^{k'}$ in the session k' of $\Pi_j^{k'}$.
3. p_j is the partner of Π_i^k in the session k of Π_i^k.

4.6. Freshness

In this definition, we define freshness. We assume that there is an instance where Π_i^k is "fresh" if it satisfies the following conditions.

1. Π_i^k has not been queried the reveal query $Reveal(i, k)$.
2. There is a partner $\Pi_j^{k'}$ that is matched to partner Π_i^k by the partner function, and Π_i^k has not been queried the reveal query $Reveal(j, k')$.
3. The partner of Π_i^k is not the inside attacker during communication in the instance of the player j.

4.7. Forward Secrecy (FS)

Our proposed two factor patient authentication scheme is forward secrecy (FS) if $\mathcal{A}$ cannot compromise the past information, even if they have sent $Corrupt(i)$ (or $Corrupt(j)$) to the player i, where $i, j \in \{U, V\}$.

Theorem 1. *First, we assume that ASE is an indistinguishable-CCA (Ind-CCA) secure asymmetric encryption/decryption scheme and equips two secure hash functions, H_1 and H_2, which we can be replaced with two random oracle (RO) functions, respectively. We also assume that our proposed patient electronic health record exchange scheme (PEHRES) that is forward secure (FS) and unforgeable (Unf) also satisfies the following situations. In other words, if our proposed scheme is secure, then*

$$Adv_{PEHRES}^{FS,Unf,Ind-CCA}(\theta,t) \leq \frac{1}{2}(I^2 q_h q_e q_s (Adv_{ASE,\mathcal{D},C_{HCA}^*}^{Ind-CCA}(\theta,t')) + 1) + \frac{1}{2}(I^2 q_h q_e (Adv_{ASE,\mathcal{D},C_V^*}^{Ind-CCA}(\theta,t')) + 1) + \frac{1}{2}((Iq_h)^2 Adv_{\mathcal{A},SE}^{Ind}(\theta,t^*) + 1) + (I^3 q_s) Adv_{Sig,\mathcal{S},\mathcal{F}}^{Unf}(\theta,t^*) + \varepsilon, t \leq t' + t^*, \quad (4)$$

where t is the total execution time, t' is the maximum total experiment time including an adversary execution time, t^* is the maximum total time to guess the real session key, I is an upper bound on the number of parties, with at most q_e encryption queries at most q_s decryption oracles, and q_h is an upper bound on the number of H_1 and H_2 queries in the experiment, where ε is a negligible advantage.

5. Security Analysis

In this section, we provide security analysis and functional analysis of our proposed scheme.

5.1. Replay Attack Resistance

In this EHR migration phase, we adopt random values r_U'', r_V^*, and r_W^* as our authentication challenge numbers. We assume an attacker can capture authentication messages among the protocol communication and may replay these captured messages to the server W to impersonate the patient U. First, the server V will check that this message was used before in some session before communicating with the server W. Hence, the server V will also check that one of these messages r_U'', r_V^*, and r_W^* was used before. If one of them was used, then it would close this session and save the record as the replay attack from V.

5.2. Resist User Impersonation Attack

In this proposed scheme, the adversary cannot replay any authentication message without the user U's biometric information Bio_U, and it also cannot guess the random number r_U'' successfully to impersonate the server V. Additionally, the adversary does not have the non-negligible probability to forge the patient's signature to the server V. In addition, the server V also checks the signature S_U to authenticate the patient U's identity in the migration registration phase. Thus, the adversary cannot have non-negligible probability to forge U's signature S_U under the RSA factoring problem. Therefore, our scheme can resist user impersonation attacks.

5.3. Provide Mutual Authentication

In the EHR migration phase, a patient U can delegate the server V to perform the EHR migration exchange with the system of the desired hospital W. Server V can perform the challenge response with the server of W, and they both communicate a session key for later usage after successful authentication. During the authentication rounds, V and W can check the freshness of random numbers (r_U'', r_V^*, and r_W^*). If one of them is to be replayed, V or W would find out and deny this session with the other party. Finally, it would close this phase and record that there was a replay attack in this EHR migration phase.

5.4. Provide Data Security

In the EHR migration phase, all random numbers are generated by these two parties and drop off when the authentication between them is successful. In addition, not only are r_U'', r_V^*, and r_W^* verified by these two parties V and W, but also they can also be response messages to confirm their respective identities.Hence, the adversary cannot have a non-negligible probability to replace each of these messages to pass the authentication process. In the data recovery phase, r_U'' is used to encrypt the patient's EHR, and the adversary does not have a non-negligible probability to obtain a patient's EHR, under the assumption that

the symmetric encryption/decryption function is indistinguishable for the adversary in a polynomial time bound.

5.5. Session Key Establishment

In the EHR migration phase, the server V and the server W can also communicate a common session key after they perform challenge-response authentication with each other successfully. Not only can this session key be used for later communication, but it can also provide for symmetric encryption/decryption usage. In the appendix, we provide a formal security proof of the session key.

5.6. Forward Secrecy Proof

In the EHR migration phase, a patient can delegate the server V to authenticate with the desired server of hospital W. They can then build the session key after successful authentication. In fact, they can use this session key to communicate with each other to transfer the patient U's EHRs or update the patient U's EHR. With this property, the system can reduce the communication bits and improve the efficiency of data transmission. In the appendix, we also provide a formal secrecy proof of the session key.

5.7. EHR Fair Exchange

In the EHR migration phase, if W does not receive the U's EHR from the V or if U's EHR is broken, then the patient U can perform the data recovery request to the HCA and ask the HCA for help to solve this situation by providing the above signatures and V's receipt to HCA. If the above signatures are valid, HCA performs the data recovery phase and forwards the encrypted patient's EHR to the system of the hospital W. Finally, the server of hospital W can also obtain the patient U's EHR under the help of HCA.

5.8. Offline Trusted Third Party

In the proposed scheme, we assume that there is a HCA and that it generates the patient's EHR migrating signature with a delegation document and performs data recovery. Here we can assume that the on-line device of the HCA can generate the signature after verifying the request party's signature in the migration registration phase. Only if there is a request coming in the data recovery phase would the HCA be on-line and solve this situation after verifying the request party's evidence, including the registration signatures and the related signatures. From the above setting, our trusted third party would not stay on-line all the time and just appears when it is needed. Additionally, only the HCA knows the link information (UID_U, ID_U) of the patient U. Therefore, the patient can prevent their real identity from being disclosed during the EHR migration transaction.

From the above security analysis properties, we take [10] as a reference and make comparisons with schemes from [6–10]. In the following, we provide some security analysis definitions for security comparison (Table 1).

Table 1. Security comparison.

Attributes	[6]	[7]	[8]	[9]	[10]	Ours
A1	Y	Y	Y	Y	Y	Y
A2	N	N	Y	Y	Y	Y
A3	Y	Y	Y	Y	Y	Y
A4	N	N	N	N	N	Y
A5	Y	Y	Y	Y	N	Y
A6	Y	N	N	N	N	Y
A7	N	N	N	N	N	Y
A8	N	N	N	N	N	Y

A1: Replay Attack Resistance; A2: Resist User Impersonation Attack; A3: Provide Mutual Authentication, A4: Provide Data Security; A5: Session Key Establishment, A6: Forward Secrecy Proof; A7: EHR Fair Exchange, A8: Offline Trusted Third Party.

5.9. Efficiency Comparisons

In this section, we evaluate our proposed scheme's efficiency. First, we assume that our scheme's parameter p is of 1024 bits for security consideration. We assume that H is the computation time of one hashing operation, Exp is the computation time of one modular exponential operation in a 1024 bit module, M is the computation time of one modular multiplication in a 1024 bit module, EC_M is the computation time of a number over an elliptic curve, and EC_P is the computation time of a bilinear pairing operation of two elements over an elliptic curve in [13–15]. We also let Sig, ASE, ADE, SE, and SD be the signature operation time, the asymmetric encryption time, the asymmetric decryption time, the symmetric encryption time, and the symmetric decryption time, respectively. We assume that our proposed scheme can be implemented on an elliptic curve over a 163-bit field and has the same security level of a 1024 bit public key crypto-system such as RSA or the Diffile-Hellman cryptosystem. We also assume that $Exp = 8.24EC_M$ for the ARM CPU to the processor in 200 Mhz [15]. We also determine certain relations from the following: $Exp \approx 240M = 600H \approx 3EC_P$, and $EC_A \approx 5M$ in [16–22].

Based on [23], a public key encryption/decryption operation time in an elliptic curve is approximately $1EC_A$ and $1EC_M + 1EC_A$, respectively. Therefore, our proposed scheme total computation time cost is about $9H + 3Sig + 14 \oplus + 2ASE + 1ADE \approx 60.075M + 14\oplus$. Due to the different properties of the above schemes, we omitted the efficiency comparisons and found some currently survey papers [11,24] that have the same functional properties as our proposed scheme.

In [11], the authors proposed a dynamic consent model of health data sharing using blockchain technology. They combine the consent representation models (DUO) and ADA-M [24] to let patients control their EHR sharing to match the request query with full access rights. Their method is designed for building an EHR platform but is not a practical mechanism for patients exchanging their EHRs with a formal security proof in a blockchain environment. In [12], the authors proposed an EHR with a patient-centric access right framework model by using blockchain technology. We think that this is a good idea for building health information exchange systematic modules with blockchain in the future, but they do not offer a practical solution for EHR migration currently, even in a blockchain environment. Our proposed scheme is established by the functional block such as the signature functions with other authentication functions. In future work, our proposed scheme could functionally add a smart contract function to generate a verifiable functional patient EHR block in blockchain network. Hence, our proposed scheme could be used in blockchain and non-blockchain environments.

In the efficiency evaluation of our scheme, we used a desktop with Ubuntu 20.04 with Intel(R) Core(TM) i7-8700 CPU @ 3.20 GHz CPU and 15 GB memory. The simulation

experiment was carried out using GO language, and the standard "crypto/elliptic" library was used. We simulated every phase 20 times, shown in Figures 4–6.

In the future, we will discuss the forged HCA problem [25] and other applications such as neural network environments for COVID-19 patients [26] exchanging their EHRs. We hope to have a good solution to the above problems.

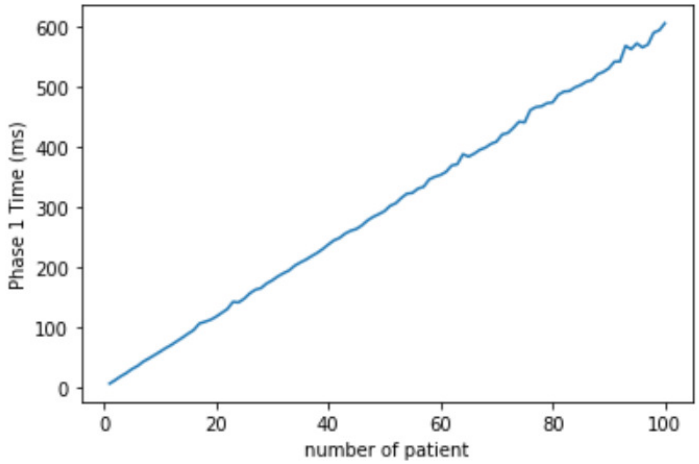

Figure 4. The migration registration phase simulation.

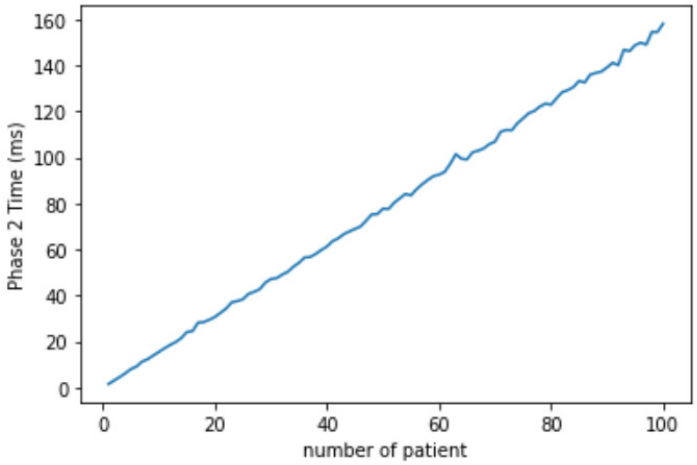

Figure 5. The EHR migration phase simulation.

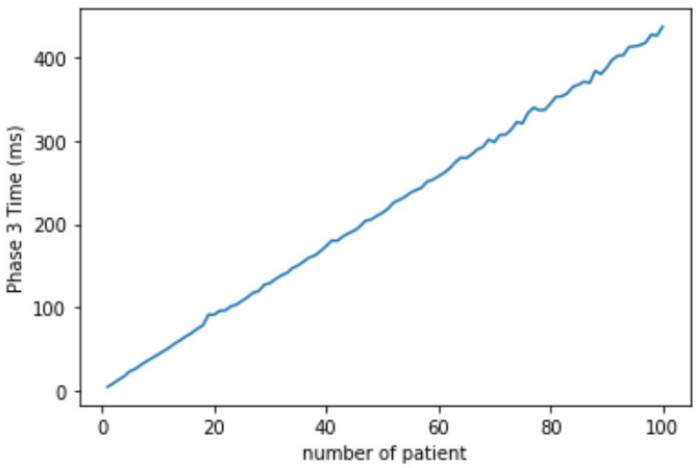

Figure 6. The data recovery phase simulation.

6. Security Proof

In this section, we continue to demonstrate what an adversary is and its probability. We model the *Game*, our scheme simulation steps, and the related oracle responses.

An adversary (call it $\mathcal{S}$) can control all communication messages in this scheme. The adversary can obtain related information by sending oracles. A *Game* is the simulation of our proposed scheme, which can equip all kinds of oracles, and oracles can reply back according to the adversary's questions. There is also another adversary (call it $\mathcal{S}$) that controls the simulation and takes $\mathcal{A}$'s ability to break the hard problem defined in the security definition.

Let *Game* be a "game", the simulation of our scheme, where the adversary $\mathcal{A}$ can ask queries to the oracles, and the oracles can answer back to the adversary. The following are query types that an adversary can make in the game.

- Send query $Send(i, k, M)$(or $Send(j, k', M)$): this query models an adversary that can send message M to the authentication party i(or j), where $i, j \in \{U, V\}$ in the k- (or k')-th session, where k and k' are two different session numbers in N.
- Reveal query $Reveal(i, k)$(or $Reveal(j, k)$): this query is used to model a situation that exposes a session key of Π_i^k(or $\Pi_j^{k'}$) to an adversary, where $i, j \in \{U, V\}$ and $k, k' \in N$ in the k(or k')-the session.
- Encryption query: this query is used to model that an adversary can obtain a ciphertext C' on the input of a chosen message M'.
- Decryption query: this query is used for modelling an adversary that can obtain a plain-text M' on the input of a cipher-text C'.
- Corrupt query $Corrupt(i)$(or $Corrupt(j)$): this query is used to expose the private key of the player p_i(or p_j) to the adversary, where $i, j \in \{U, V\}$.
- Hash query: this query depends on what the input is; the simulator then returns the related output to the attacker.
- Test query $Test(i, k)$: this query is used to define the advantage of an adversary. When the adversary $\mathcal{A}$ has finished all of the above queries to the oracle, they can make this test query on an instance Π_i^k(or $\Pi_j^{k'}$) to the simulator. At this time, the simulator will flip a coin b. If b is 1, then the real session key is $ssk_{i,j}^k$. Otherwise, it returns a random string chosen uniformly from $\{0, 1\}^*$. The adversary is only allowed to ask for the "fresh" instance of a player in the above simulation.

Proof of Theorem 1. First, we assume that there is an adversary $\mathcal{A}$ that attempts to attack our patient EHR exchange scheme ($PEHRES$) in the forward secure sense. We then let dis be the event at which $\mathcal{A}$ can distinguish at least one ciphertext in $PEHRES$ with non-negligible probability. At the same time, we also let $forge$ be the event at which the adversary $\mathcal{D}$ can forge the signature of our PES with non-negligible probability. We assume that

$$Pr_{\mathcal{A}}[b = b'] \leq Pr_{\mathcal{A}}[b = b' \wedge \overline{dis} \wedge \overline{forge}] + Pr_{\mathcal{A}}[dis] + Pr_{\mathcal{A}}[forge],$$

where b and b' are coin flips chosen by the simulator and the attacker $\mathcal{A}$, respectively.

We also assume that

$$Pr_{\mathcal{F}^*}[forge] \leq Pr_{\mathcal{F}^*}[\mathcal{F}^* \rightarrow S_U^* | Ver(S_U^*) = 1] + Pr_{\mathcal{F}^*}[\mathcal{F}^* \rightarrow S_{HCA}^* | Ver(S_{HCA}^*) = 1],$$

where S_U^* and S_{HCA}^* are signatures forged by the attacker $\mathcal{F}^*$, respectively. We then use three lemmas to complete this security proof in the following.

Lemma 4. *We assume that there is no event such that the attacker $\mathcal{A}$ can distinguish the ciphertext C^* with non-negligible probability*

$$Pr_{\mathcal{A}}[dis] \leq \frac{1}{2}(I^2 q_h q_e q_s (Adv_{ASE,\mathcal{D},C^*_{HCA}}^{Ind-CCA}(\theta, t')) + 1) + \frac{1}{2}(I^2 q_h q_e (Adv_{ASE,\mathcal{D},C^*_V}^{Ind-CCA}(\theta, t')) + 1),$$

in the polynomial time bound t' under the above Ind-CCA security definition with q_h hash queries, at most q_e encryption queries, and at most q_s decryption queries, respectively.

Proof of Lemma 4. We assume that $Pr[dis]$ is a non-negligible probability in the simulation game. We can then construct an attacker $\mathcal{D}$ whose work is to distinguish the cipher-text under the Ind-CCA encryption/decryption scheme. There is also an attacker $\mathcal{F}$ whose goal is to break the encryption/decryption of our proposed scheme SE. Next, we construct $\mathcal{D}$ as the simulator that simulates the attacking environment in which $\mathcal{F}$ can mount its attack. First, $\mathcal{D}$ simulates an encryption oracle $SE_{pk_i}(\cdot, \theta)$, where $i \in \{U, V\}$, and generates the C' to the attacker $\mathcal{F}$ on the plain-texts (M') chosen by the attacker $\mathcal{D}$ in the selected instance $\Pi_{i^*}^k$, where the partner of $\Pi_{i^*}^k$ is p_{j^*}. In addition, $\mathcal{D}$ also simulates the decryption oracle to answer the decryption query issued by the attacker $\mathcal{D}$. We consider the following steps. First, $\mathcal{D}$ prepares all hash functions, including H_1 and H_2, two hash functions with collision-resistance. It also generates the instances $i^*, j^* \longleftarrow [1, ..., I-1]$ of each player i, where $i \in \{U, V\}$. It can make the above two hash queries l^* times, where $l^* \longleftarrow [1, ..., q_h]$.

Hash query

In this hash query phase, the simulator also responds to all kinds of hash queries in each stage.

- In the migration registration phase, the simulator generates the corresponding hashed value to U and V. It prepares the $H_1(Bio_U)$ for the patient U as the registration token. At the same time, the simulator chooses r''_U and makes the ciphertext $C_{HCA} = (r''_U, Bio_U, UID_U, Cert_U, ID_U, EHR_U \oplus r''_U)$ for $\mathcal{F}$.
- Next, the simulator has to simulate the signature S_U and S_{HCA} from the signing oracle. It then forwards (S_U, S_{HCA}) to $\mathcal{F}$. The simulator then computes $Sig_V(S_{HCA}, S_U, Agree_{V->W})$ as the response receipt of the instance of player V.
- In the EHR migration phase, the simulator can simulate the hashed values $H_2(r_V^* + 1)$ and $H_2((r_W^* + 1) \oplus r''_U)$ to $\mathcal{F}$, which forwards $r_V^* + 1$, $(r_W^* + 1)$, and r''_U as challenge random numbers.

Phase 1

- In the migration registration phase, the attacker $\mathcal{F}$ can issue the encryption query on the chosen message $M' = (r''_U, Bio_U, UID_U, Cert_U, ID_U, EHR_U \oplus r''_U)$. $\mathcal{D}$ can then forward this M' to the encryption oracle and pass the final result C' to the $\mathcal{F}$.
- The attacker $\mathcal{F}$ can issue the decryption query on the ciphertext C'. $\mathcal{D}$ can then forward this C' to the decryption oracle and pass the final message $M' = (r''_U, Bio_U, UID_U, Cert_U, ID_U, EHR_U \oplus r''_U)$ from the oracle output to the $\mathcal{F}$.
- In the EHR migration phase, the attacker $\mathcal{F}$ can ask for the $M'' = r_V^*$ encryption result C''_V and the decryption result of M''. $\mathcal{D}$ can forward C''_V and M'' to the attacker $\mathcal{F}$.

Challenge

- In this phase, $\mathcal{D}$ can generate the ciphertext C^*_{HCA} by querying the asymmetric encryption oracle $ASE_{pk_T}(M_b^*, \theta)$ with the coin flip b on the target message pair (M_0^*, M_1^*) chosen by the attacker $\mathcal{F}$. If b=1, C is computed from $ASE_{pk_T}(M_1^*, \theta)$, where $T \in \{U^*, V^*\}$. Otherwise, it returns the ciphertext C^* to $\mathcal{F}$, where $C^*_{HCA} \longleftarrow ASE_{pk_T}(M_0^*, \theta)$. The only restriction is that $\mathcal{F}$ cannot ask for the decryption query on the ciphertext C^*_{HCA}. On the other hand, we also consider that the ciphertext C^*_V is the same situation. We also set up the target message pair (M_0^{**}, M_1^{**}) chosen by the attacker $\mathcal{F}$. If b'=1, C is computed from $ASE_{pk_T}(M_1^{**}, \theta)$, where $T \in \{W^*\}$. Otherwise, it returns the ciphertext C^*_V to $\mathcal{F}$, where $C^*_V \longleftarrow ASE_{pk_T}(M_0^{**}, \theta)$.
- We assume that this event dis happens with respect to the instance $\Pi_{i^*}^{l^*}$ of the player i, where its partner player is p_{j^*}. At this time, $\mathcal{F}$ finally outputs its own guessing bit b'. Otherwise, the system stops this authentication stage and aborts this simulation.

Finally, $\mathcal{F}$ has a set with instances of players i^* and j^* with q_h total hash queries , at most q_e encryption queries, and q_s decryption queries. At this time, $\mathcal{D}$ does not fail in the simulation environment with $\mathcal{F}$'s correct guessing, where $b = b'$ has non-negligible probability. The following equation will then hold:

$$\begin{aligned} Adv^{Ind-CCA}_{ASE,\mathcal{D},C^*_{HCA}}(\theta, t') \leq \\ \frac{1}{I^2 q_h q_e q_s}(Pr[Game^{Ind-CCA-1}_{ASE,\mathcal{F}}(\theta) = 1] - Pr[Game^{Ind-CCA-0}_{ASE,\mathcal{F}}(\theta) = 1]) = \\ \frac{1}{I^2 q_h q_e q_s}(Pr[Game^{Ind-CCA-1}_{ASE,\mathcal{F}}(\theta) = 1] - (1 - Pr[Game^{Ind-CCA-1}_{ASE,\mathcal{F}}(\theta) = 1])) = \\ \frac{1}{I^2 q_h q_e q_s}(2(Pr[Game^{Ind-CCA-1}_{ASE,\mathcal{F}}(\theta) = 1]) - 1). \end{aligned} \tag{5}$$

In the ciphertext C^*_V simulation game, we have the same simulation as above. Therefore, we omitted the simulation, but we also conclude that

$$Adv^{Ind-CCA}_{ASE,\mathcal{D},C^*_V}(\theta, t') \leq \frac{1}{I^2 q_h q_e}(2(Pr[Game^{Ind-CCA-1}_{ASE,\mathcal{F}}(\theta) = 1]) - 1) \tag{6}$$

We then can summarize the total probability as follows:

$$\begin{aligned} Pr_{\mathcal{A}}[dis] \leq Pr[Game^{Ind-CCA-1}_{ASE,\mathcal{F}}(\theta) = 1] \leq \\ \frac{1}{2}(I^2 q_h q_e q_s (Adv^{Ind-CCA}_{ASE,\mathcal{D},C^*_{HCA}}(\theta, t')) + 1) + \frac{1}{2}(I^2 q_h q_e (Adv^{Ind-CCA}_{ASE,\mathcal{D},C^*_V}(\theta, t')) + 1). \end{aligned} \tag{7}$$

□

Lemma 5. *Before we prove this lemma, we assume that there is no attacker $\mathcal{A}$ that can guess the real session key in the event that the ciphertext C^* generated by the symmetric encryption (SE) functions cannot be distinguished by $\mathcal{A}$ correctly with non-negligible probability. We then have*

$$Pr_{\mathcal{A}}[b = b' \wedge \overline{dis} \wedge \overline{forge}] \leq \frac{1}{2}((Iq_h)^2 Adv^{Ind}_{\mathcal{A},SE}(\theta, t^*) + 1),$$

in the polynomial time t^* under the random oracle (RO) assumption with total q_h hash queries.

Proof of Lemma 5. In this proof, we construct another simulator $\mathcal{C}$ that also simulates the attacking environment for $\mathcal{A}$ mounting its attack. Finally, if $\mathcal{A}$ can guess the real session key successfully with the non-negligible property, then we can use $\mathcal{A}$ to break the random oracle assumption.

- First, we assume that $\mathcal{C}$ is given the system parameters $(G, g, q, H_1, H_2, t_{i^*}, t_{j^*})$. It starts to choose public key/secret key pairs for all parties except for p_{i^*} and p_{j^*}. Next, $\mathcal{C}$ selects other protocol parameters such as $(i^*, j^*) \longleftarrow [1, ..., I-1]$ and $t_1, t_2 \longleftarrow [1, ..., q_h]$. At this time, we also let the target identities i^* and j^* be the instances of the patient U and the system V, respectively.
- After the above environment is set up completely, $\mathcal{C}$ starts to simulate the following oracle queries and related hash functions.
- First, $\mathcal{C}$ sets parameters $(i, j, r_i, r_j, AE_{pk_T}(M_b, \theta))$, where r_i, r_j are two random numbers, and H_1,H_2 are these two collision-resistance hash functions. In addition, $\mathcal{C}$ adopts θ as the security parameter. First, $\mathcal{C}$ prepares two nonce challenge numbers r_i and r_j in the t_1-th and t_2-th session, respectively.
- During this simulation, for each query issued from $\mathcal{A}$, $\mathcal{C}$ answers it as follows:
- It takes $(i, j, H_1, H_2, \omega_1, \omega_2, r_i, r_j)$ as its input and responds to each *Send* query in the instance Π^k_i in the k-th session on the message M, where ω_1 and ω_2 are two pseudo-random functions with the same length of the hash oracles H_1 and H_2, respectively.

□

Hash Query

In this hash query phase, the simulator can answer all kinds of harsh queries in each stage, as follows:

- In the migration registration phase, the simulator also computes the initial biometric information value ($r_{B_U} \oplus H_1(Bio_U)$) for the patient U and keeps them in the oracle simulation database. If the $\mathcal{A}$ makes the hash query on ($r_V + 1||r_{HCA}$) and $r_{HCA} + 1$, $\mathcal{C}$ also forwards them to the hash oracle and returns the response hashed value to $\mathcal{A}$.
- In the EHR migration phase, $\mathcal{A}$ asks for the $r_V^* + 1$, $(r_W^* + 1) \oplus r_U''$, and $(r_W^* + 1)||(r_V^* + 1)$ hash values of the H_2 function, and $\mathcal{C}$ also forwards them to the hash oracle and lets the hash oracle generate the corresponding hash values to $\mathcal{A}$.
- If $\mathcal{C}$ receives the $Corrupt(i)$ query, then it returns the private key to $\mathcal{A}$. If $\mathcal{C}$ receives the $Reveal(i)$ query, it checks if $i \neq i^*$ or $j \neq j^*$, then $\mathcal{D}$ computes the session key $ssk_{i,j} = H_1(\omega_1(r_W^* + 1))||\omega_2((r_V^* + 1)))$, where $r_W^* + 1$ and $r_V^* + 1$ are chosen from ω_1 and ω_2 functions, respectively. On the other hand, if $i = i^*$, $j = j^*$, and $t_1 = t_2 = l$, then $\mathcal{C}$ computes $H_1((r_W^* + 1)||(r_V^* + 1))$ as the session key $ssk_{i,j}$ and delays this key in the response in the *Test* query.
- If the adversary $\mathcal{A}$ has finished the above queries, it can make the *Test* query to $\mathcal{C}$. At this time, $\mathcal{C}$ will check the instance session and player to see if $i = i^*$ and $j = j^*$, and $\mathcal{C}$ then tosses a coin b to answer the session key. If b=0, then it computes $ssk_{i,j} = H_1((r_W^* + 1)||(r_V^* + 1))$. Otherwise, it computes $ssk_{i,j} \longleftarrow \{0,1\}^*$ from the random pseudo-random function.

Finally, if $\mathcal{C}$ answers the *Test* query for $\Pi_{i^*}^{t_1}$ and $\Pi_{j^*}^{t_2}$ by using $(Z_n^*, H_1, H_2, \omega_1, \omega_2)$, and $\mathcal{A}$ does not fail in guessing b', then $\mathcal{A}$ answers the session key depending on its coin flip b'. We can have

$$\begin{aligned} &Adv_{\mathcal{C},\mathcal{A},SE}^{H_1,H_2,\omega_1,\omega_2}(\theta,t) = \\ &Pr[\mathcal{C}(Z_n^*, H_1, H_2, \omega_1, \omega_2) = 1 | ssk_{i,j} = H_1((r_W^* + 1)||(r_V^* + 1)))] - \\ &Pr[\mathcal{C}(Z_n^*, H_1, H_2, \omega_1, \omega_2) = 1 | ssk_{i,j} \longleftarrow \{0,1\}^*, t \in Z_q^*] \leq \\ &\frac{1}{(Iq_h)^2}(Pr[\mathcal{A}(\cdot) = 1 | ssk_{i^*,j^*} \text{ is real in } Test \text{ query}] - Pr[\mathcal{A}(\cdot) = 1 | ssk_{i^*,j^*} \text{ is random in } Test \text{ query}]) \leq \\ &\frac{1}{(Iq_h)^2}(2Pr_{\mathcal{A}}[b = b' \wedge \overline{dis} \wedge \overline{forge}] - 1). \end{aligned} \tag{8}$$

Finally, we could conclude that

$$Pr_{\mathcal{A}}[b = b' \wedge \overline{dis} \wedge \overline{forge}] \leq \frac{1}{2}((Iq_h)^2 Adv_{\mathcal{C},\mathcal{A},SE}^{H_1,H_2,\omega_1,\omega_2}(\theta,t) + 1). \tag{9}$$

Lemma 6. *Before we prove Lemma 1, we assume that there is no event such that the attacker $\mathcal{F}^*$ can forge the signature S_U of patient U with non-negligible probability*

$$Pr_{\mathcal{F}}[forge] \leq (I^3 q_s(Adv_{Sig,\mathcal{S},\mathcal{F}}^{Unf}(\theta, t^*),$$

in the polynomial time bound t^* under the above Ind-CCA security definition with q_h hash queries, at most q_e encryption queries, and at most q_s decryption queries, respectively.

Proof of Lemma 6. In this lemma proof, we start to prove our above **Lemma 1** (Unforgeability). To start the proof of **Lemma 1** (Unforgeability), we defined the **Game** as

the simulation game that runs as the proposed protocol controlled by the simulator $\mathcal{S}$. We define **Game** as follows.

$\mathbf{Game}^{Unf}_{\mathcal{A},Sig}(\theta, t)$
Phase 1.
$\mathcal{F} \longleftarrow \{M\}^l$
$S_i \longleftarrow \mathcal{F}^{Sig(sk_i,M_i),RO_1,RO_2,H_1,H_2}(\theta, t)$
Challenge Phase.
$i \in \{U, HCA\}, M^* \longleftarrow \mathcal{F}(M)$,
Loop j=1 to l
$S'_{i,j} \longleftarrow \mathcal{F}^{Sig(sk_i)}(\theta, t)(M^*)$
If (Ver($S'_{i,j+1}$) == 1 and $S'_{i,j+1} \notin S_{i,j}$).
Break
Return $S'_{i,j+1}$.
else if (j <= l)
goto Loop

We first define the simulator $\mathcal{S}$ as the simulator that is given in the RSA factoring problem, and we assume there is an attacker $\mathcal{F}$ whose goal is to forge a valid signature on the Sig function block.

The simulator $\mathcal{S}$ first chooses the security parameter l with the message space M^l. The $\mathcal{S}$ also selects two collision-resistance hash functions that map from $Z^*_n \longrightarrow \{0,1\}$ and two hash oracles RO_1 and RO_2, respectively. After setting up the system parameter, the $\mathcal{S}$ simulates each phase in the proposed EHR scheme. In the migration registration phase, the attacker $\mathcal{F}$ can impersonate the patient U to ask for U's signature request S_U on the desired message. When $\mathcal{S}$ has received this request, it takes the message as input and outputs the signature S_U with the help of the above secure digital signature function Sig. It then returns this S_U back to the $\mathcal{F}$. The $\mathcal{F}$ can also continue to ask the hospital certification center HCA's signature on the received signature S_{HCA} of patient P. It also receives the message tuple $(S_U, Date, ID_U)$ and outputs the signature S_{HCA} back to $\mathcal{F}$. In addition, it is the same situation when $\mathcal{F}$ asks the signature of V. The $\mathcal{S}$ also returns S_V back to $\mathcal{F}$.

In these phases, the $\mathcal{F}$ will make the signature request in the above situation. The $\mathcal{S}$ starts the **Challenge phase** and forwards the message M_i, where $i \in \{U, V, HCA\}$. The $\mathcal{F}$ can forge l+1 signatures $S'_{i,j}$, where $S'_{i,j+1} \notin S_{i,j} \longleftarrow Sig(sk_i, M^*)$, where $i \in \{U, V, HCA\}$ and $j = 1 \sim l$ after l signature queries. Finally, this forged signature also passes the verification $Ver(S'_{i,j+1})$ successfully. We then can use $\mathcal{F}$'s ability to find a solution to the RSA factoring problem. Thus, we have

$$\begin{aligned} Adv^{Unf}_{Sig,\mathcal{S},\mathcal{F}}(\theta, t^*) \geq & |Pr[S'_{i,j+1} \longleftarrow \mathcal{F}^{Unf}_{Sig}(M^*), Ver(S'_{i,j+1} = 1)]| \\ = & \frac{1}{I^3 q_s}(Pr[S'_{i,j+1} \longleftarrow \mathcal{F}^{Unf}_{Sig}(M^*), Ver(S'_{i,j+1} = 1)]). \end{aligned} \tag{10}$$

Finally, we could conclude that

$$Pr_{\mathcal{F}}[forge] \leq (I^3 q_s) Adv^{Unf}_{Sig,\mathcal{S},\mathcal{F}}(\theta, t^*).$$

□

After summarizing the above three lemmas, we can conclude that $\frac{1}{2}(I^2 q_h q_e q_s(Adv^{Ind-CCA}_{SE,\mathcal{D},C^*_{HCA}}(\theta, t')) + 1) + \frac{1}{2}(I^2 q_h q_e(Adv^{Ind-CCA}_{SE,\mathcal{D},C^*_V}(\theta, t')) + 1) + \frac{1}{2}((Iq_h)^2 Adv^{Ind}_{\mathcal{A},SE}(\theta, t^*) + 1) + (I^3 q_s) Adv^{Unf}_{Sig,\mathcal{S},\mathcal{F}}(\theta, t^*)$. □

7. Conclusions

We propose a practical and provable patient EHR fair exchange scheme with key agreement for e-health information systems. Not only does our scheme offer a solution for the seven problems described in Section 2, when a patient attempts to migrate their personal information data to another hospital, but they can also maintain their anonymity during the data migration transaction. In addition, Table 1 shows a security and functional comparison with other related papers. It is obvious that our proposed scheme guarantees convenience, rapidity, and integrity.

Our mechanism provides secure data storage and the secure transfer of authorized information to designated locations. What information can be authorized, for example, whether COVID-19 patient privacy concerning patients' names, identities, and genetic sequences can be transmitted to different hospitals, is beyond this study's scope. This study guarantees secure data transfer and storage. Our scheme also provides a formal security proof in the random oracle model under chosen-ciphertext security. Our approach focuses on the security and privacy protection of patient EHRs rather than on the design of electronic health systems. It not only serves as a high-level functional module for integrity but also provides an efficient and contactless data transfer method that allows for medical data aggregation and protects patient anonymity, especially relevant in the context of the global COVID-19 pandemic. In the future, we will extend our scheme to be applicable for COVID-19 patient EHR exchange in a neural network environment.

Author Contributions: Conceptualization, M.-T.C.; Formal analysis, M.-T.C. and T.-H.L.; Methodology, M.-T.C. and T.-H.L.; Writing—original draft, M.-T.C.; Writing—review & editing, T.-H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: This study was supported in part by grants from the Ministry of Science and Technology of the Republic of China (Grant No. MOST 109-2221-E-167-028-MY2).

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

References

1. Saranummi, N. In the spotlight: health information systems. *PHR Value Based Healthc.* **2011**, *2*, 15–17. [CrossRef] [PubMed]
2. Chiuchisan, I.; Balan, D.G.; Geman, O.; Chiuchisan, I.; Gordin, I. A Security Approach for Health Care Information Systems. In Proceedings of the E-Health and Bioengineering Conference (EHB), Sinaia, Romania, 22–24 June 2017; pp. 721–724.
3. Appari, A.; Johnson, M. Information security and privacy in healthcare: Current state of research. *Int. J. Internet Enterp. Manag.* **2010**, *4*, 279–284. [CrossRef]
4. Mahmoud, A.M.; Zeki, A.M. Security issues with health care information technology. *Int. J. Sci. Res. (IJSR)* **2015**, *12*, 1021–1024.
5. Health Level Seven. Available online: http://www.hl7.org/implement/standards/ansiapproved.cfm (accessed on 21 December 2020).
6. Das, A.K.; Goswami, A. A secure and efficient uniqueness- and-anonymity-preserving remote user authentication scheme for connected health care. *J. Med. Syst.* **2013**, *3*, 9948. [CrossRef] [PubMed]
7. He, D.; Kumar, N.; Chen, J.; Lee, C.C.; Chilamkurti, N.; Yeo, S.S. Robust anonymous authentication protocol for health- care applications using wireless medical sensor networks. *Multimed. Syst.* **2015**, *1*, 49–60. [CrossRef]
8. Amin, R.; Islam, S.K.H.; Biswas, G.P.; Khan, M.K.; Kumar, N. A robust and anonymous patient monitoring system using wireless medical sensor networks. *Future Gener. Comput. Syst.* **2018**, *80*, 483–495. [CrossRef]
9. Zhang, L.; Zhang, Y.; Tang, S.; Luo, H. Privacy protection for e-health systems by means of dynamic authentication and three-factor key agreement. *IEEE Trans. Ind. Electron.* **2017**, *65*, 2795–2805. [CrossRef]
10. Kaul, S.D.; Murty, V.K.; Hatzinakos, D. Secure and privacy preserving biometric based user authentication with data access control system in the healthcare environment. In Proceedings of the 2020 International Conference on Cyberworlds (CW), Caen, France, 29 Septemeber–1 October 2020; pp. 249–256.
11. Jaiman, V.; Urovi, V. A Consent Model for Blockchain-Based Health Data Sharing Platforms. *IEEE Access* **2020**, *8*, 143734–143745. [CrossRef]

12. Zhuang, Y.; Sheets, L.R.; Chen, Y.W.; Shae, Z.Y.; Tsai, J.J.P.; Shyu, C.R. A Patient-Centric Health Information Exchange Framework Using Blockchain Technology. *IEEE J. Biomed. Health Inform.* **2020**, *24*, 2169–2176. [CrossRef] [PubMed]
13. Jurisic, A.; Menezes, A.J. Elliptic Curves and Cryptography; pp. 1–13. Available online: http://http://www.cs.nthu.edu.tw/~cchen/CS4351/jurisic.pdf (accessed on 22 December 2020).
14. Koblitz, N.; Menezes, A.; Vanstone, S. The state of Elliptic curve cryptography. *Des. Codes Cryptgography* **2000**, *19*, 173–193. [CrossRef]
15. Lauter, K. the Advantages of Elliptic curve cryptography for wireless security. *IEEE Wirel. Commun.* **2004**, *11*, 62–67. [CrossRef]
16. LI, Z., Higgins, J., Clement, M., Performance of Finite Field Arithmetic in an Elliptic Curve Cryptosystem. In Proceedings of the 9th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'01), Cincinnati, OH, USA, 15–18 August 2001; pp. 249–256.
17. Ramachanfdran, A.; Zhou, Z.; Huang, D. Computing cryptography algorithm in Portable and embedded devices. In Proceedings of the IEEE International Conference on Portable Information Devices, Orlando, FL, USA, 25–29 May 2007; pp. 1–7.
18. Schneier, B. *Applied Cryptography*, 2nd ed.; John Wiley & Sons: Hoboken, NJ, USA, 1996.
19. Takashima, K. Scaling Security of Elliptic Curves with Fast Pairing Using Efficient Endomorphisms. *IEICE Trans. Fundam.* **2007**, *E90-A*, 152–159. [CrossRef]
20. Bertinoi, G.; Breveglieri, L.; Chen, L.; Fragneto, P.; Harrison, K.; Pelosi, G. A pairing SW implementation for smart cards. *J. Syst. Softw.* **2008**, *81*, 1240–1247. [CrossRef]
21. Hankerson, D.; Menezes, A.; Scott, M. Software Implementation of pairings. *Identity-Based Cryptogr. Cryptol. Inf. Secur.* **2008**, *2*. [CrossRef]
22. Hohenberger, S. Advances in Signatures, Encryption, and E-cash from Bilnear Groups. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2006.
23. Juang, W.S.; Lei, C.L.; Liaw, H.T.; Nien,W.K. Robust and Efficient Three-party User Authentication and key agreement Using Bilinear pairings. *Int. J. Innov. Comput. Inf. Control.* **2010**, *6*, 763–772.
24. Woolley, J.P.; Kirby, E.; Leslie, J.; Jeanson, F.; Cabili, M.N.; Rushton, G.; Hazard, J.G.; Ladas, V.; Veal, C.D.; Gibson, S.J.; et al. Responsible sharing of biomedical data and biospecimens via the automatable discovery and access Matrix (ADA-M). *NPJ Genom. Med.* **2018**, *3*, 7. [CrossRef] [PubMed]
25. Chakraborty, T.; Jajodia, S.; Katz, J.; Picariello, A.; Sperli, G.; Subrahmanian, V.S. FORGE: A fake online repository generation engine for cyber deception. *IEEE Trans. Dependable Secur. Comput.* **2019**. [CrossRef]
26. La Gatta, V.; Moscato, V.; Postiglione, M.; Sperli, G. An epidemiological neural network exploiting dynamic graph structured data applied to the COVID-19 outbreak. *IEEE Trans. Big Data.* **2020**, *7*, 45–55. [CrossRef]