applied sciences

MDPI

Article

CLAP-PRE: Certificateless Autonomous Path Proxy Re-Encryption for Data Sharing in the Cloud

Chengdong Ren, Xiaolei Dong *, Jiachen Shen, Zhenfu Cao and Yuanjian Zhou

Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai 200062, China; 51194501090@stu.ecnu.edu.cn (C.R.); jcshen@sei.ecnu.edu.cn (J.S.); zfcao@sei.ecnu.edu.cn (Z.C.); 52194501028@stu.ecnu.edu.cn (Y.Z.)

* Correspondence: dongxiaolei@sei.ecnu.edu.cn

Abstract: In e-health systems, patients encrypt their personal health data for privacy purposes and upload them to the cloud. There exists a need for sharing patient health data with doctors for healing purposes in one's own preferred order. To achieve this fine-gained access control to delegation paths, some researchers have designed a new proxy re-encryption (PRE) scheme called autonomous path proxy re-encryption (AP-PRE), where the delegator can control the whole delegation path in a multi-hop delegation process. In this paper, we introduce a certificateless autonomous path proxy re-encryption (CLAP-PRE) using multilinear maps, which holds both the properties (i.e., certificateless, autonomous path) of certificateless encryption and autonomous path proxy re-encryption. In the proposed scheme, (a) each user has two public keys (user's identity and traditional public key) with corresponding private keys, and (b) each ciphertext is first re-encrypted from a public key encryption (PKE) scheme to an identity-based encryption (IBE) scheme and then transformed in the IBE scheme. Our scheme is an IND-CPA secure CLAP-PRE scheme under the *k*-multilinear decisional Diffie–Hellman (*k*-MDDH) assumption in the random oracle model.

Keywords: data sharing; autonomous path; proxy re-encryption; certificateless; multi-hop; IND-CPA

Citation: Ren, C.; Dong, X.; Shen, J.; Cao, Z.; Zhou, Y. CLAP-PRE: Certificateless Autonomous Path Proxy Re-Encryption for Data Sharing in the Cloud. *Appl. Sci.* **2022**, *12*, 4353. https://doi.org/10.3390/app12094353

Academic Editor: Eui-Nam Huh

Received: 1 March 2022
Accepted: 22 April 2022
Published: 25 April 2022

1. Introduction

1.1. Background

In recent years, the rapid development of electronic medicine has benefitted from the development of Internet of Things and mobile communication equipment. Personal health data can be collected in a comprehensive way by using wireless sensors. Through the communication network, personal health data can be transmitted to the data center and then transmitted to remote experts or doctors with some additional information for healing purposes through the cloud server. By this way, the communication overhead of the patient is reduced. With this benefit, an increasing number of patients outsource their personal health data to data centers. However, the crux of the matter is data privacy. As we all know, personal health data contains a lot of sensitive information, such as patients' identity, case report and inspection report. Malicious users or data centers can easily access this privacy data if stored in plaintext. To achieve privacy, these data must be stored in data centers with encrypted forms. Due to the encrypted data, sharing data with the experts or doctors would be a significant issue.

A common solution is that patients download and decrypt the encrypted data locally, then encrypt the data with a corresponding public key. In contrast to inefficient methods, proxy re-encryption (PRE) has been proposed as a better solution. The concept of proxy re-encryption was introduced in 1998 by Blaze et al. [1]. Proxy re-encryption is used to deal with the ciphertext transformation in the same data encryption system. Ateniese et al. [2] classified PRE schemes into four different types. If the ciphertext can be transformed more than once, i.e., from A to B, and then from B to C, we call this multi-hop; otherwise, it is

a single-hop. If the ciphertext can be transformed from A to B, at the same time, from B to A, we call this bidirectional; otherwise, it is unidirectional. In a proxy re-encryption scheme, data are stored in encrypted form. In order to share the data with experts or doctors, the patient generates a re-encryption key and sends it to the cloud server. With the re-encryption key, the cloud server can transform the ciphertext under the patient's public key to that under the expert's or doctor's public key without knowing the plaintext. Solutions for proxy re-encryption have been proposed in the literature [2–5]. In a multi-hop proxy re-encryption scheme, the delegatee i may further delegates the decryption privilege to another delegatee, say j. In this case, it is possible that there is no relation between delegatee j and the delegator. This means, in a multi-hop proxy re-encryption scheme, the ciphertext may be re-encrypted to a delegatee who is unknown to the delegator. This may leak the privacy of the data, so a multi-hop PRE scheme is desired in which the delegator can control all of the delegatees. A cryptographic solution to fulfill the above requirements is autonomous path proxy re-encryption (AP-PRE) [6].

For a better understanding of the autonomous path proxy re-encryption, we illustrate it with the Figure 1 in the following scenario: suppose that A is a patient who wants to make an online treatment in hospital H. Making an online treatment requires some information which is sensitive. Patient A just wants to share the information with his/her favourite doctors, so A makes a list of his/her favourite doctors. For the privacy of the information, A uses his/her public key to encrypt the information and generates the corresponding re-encryption keys of his/her doctor list. Then, A uploads the encrypted information and re-encryption keys to the proxy. We suppose that the priority of the doctors in the list are from high to low. Therefore, the proxy re-encrypts the information from D_1 to D_n. If one of the doctors has no time to deal with the online treatment, the process of re-encryption continues; otherwise, the process of re-encryption stops.

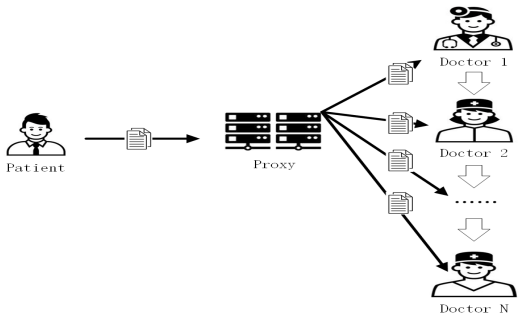

Figure 1. Electronic Health Records Sharing with Patient's Preferred Doctors.

In this scenario, Cao et al. [6] constructed a flexible PRE scheme called autonomous path proxy re-encryption (AP-PRE), where the delegator could control the next delegatee if the previous delegatee of his choice could not complete the decryption. In Cao's scheme [6], (a) there must be a certificate authority (CA) which is fully trusted and responsible for generating, distributing and managing certifications for users, and (b) Figure 2 shows that the delegation path can be forked by a malicious data user when sharing data multiple times with the same delegation path.

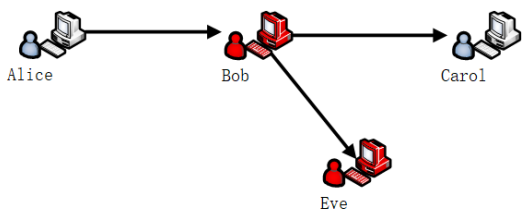

Figure 2. Forked By Malicious Data User.

Here, we show how to fork the delegation path. Recall the re-encryption key from $j-1 \rightarrow j$, and the ciphertext for j, where $j \geq 2$, the re-encryption key, is formed as $(g^{r_j}, X_j \cdot e(g_1, pk_{i_j}^{r_j}), \frac{H(X_j)}{H(X_{j-1})})$ and the ciphertext is $g^r, m \cdot e(g^r, H(X_j)), g^{r_j}, X_j \cdot e(g_1, pk_{i_j}^{r_j})$ in [6]. User j decrypts $g^{r_j}, X_j \cdot e(g_1, pk_{i_j}^{r_j})$ to get X_j, then decrypts $m \cdot e(g^r, H(X_j)), g^r$ to get message m. When sharing data with the same delegation path, the malicious user j can generate the re-encryption key using X_j. For example, the re-encryption key from $j \rightarrow (j+1)'$ is $(g^{r_{(j+1)'}}, X_{(j+1)'} \cdot e(g_1, pk_{i_{(j+1)'}}^{r_{(j+1)'}}), \frac{H(X_{(j+1)'})}{H(X_j)})$. Thus, the malicious data user can fork the delegation path when sharing data with the same delegation path.

1.2. Our Techniques

The main reason why the malicious data user can fork the delegation path is that the re-encryption keys are irrelevant to the data owner's private key apart from the first re-encryption key. In our scheme, the re-encrypted ciphertext still maintains some secret value (x in our scheme) that belongs to the data owner. Thus, in order to decrypt successfully, the re-encryption keys must include the secret value. Anyone who does not know the secret value can not generate a valid re-encryption key even if he is a valid data user.

1.3. Our Contributions

In this paper, we address the above challenge by proposing a certificateless autonomous path proxy re-encryption (CLAP-PRE). So far as we know, this is the first certificateless autonomous path proxy re-encryption scheme. Our CLAP-PRE scheme is based on the AP-PRE scheme recently proposed by Cao et al. [6]. It is also worth showing the following features of our constructions:

- Certificateless. To achieve the goal of removing the certificate authority, we introduce the certificateless public key cryptography to construct our CLAP-PRE scheme;
- Stronger Autonomous Path. This is the most important contribution. We show how to fork the delegation path in Section 1.1 and how to address it in Section 1.2. The delegation path can not be forked by the malicious data user when sharing data with the same delegation path;
- Non-interactive. For CLAP-PRE, user A can generate re-encryption keys for delegatees with only the delegatees' identities. The process of generating re-encryption keys does not need to interact with the delegatees.

1.4. Related Work

Since the concept of PRE was introduced by Blaze et al. [1], there have been many papers that have proposed PRE schemes with different properties to meet various application demands, and most of them focus on unidirectional PRE. The first CCA security unidirectional PRE scheme without random oracle was proposed in 2008 by Libert and Vergnaud [7]. Guo et al. [8] proposed a traceable unidirectional PRE scheme to prevent the proxy from abusing its re-encryption keys. Green and Ateniese [5] proposed the first identity-based PRE (IBPRE) scheme, in which the ciphertext can be re-encrypted from one identity to another identity. Later, many solutions [9,10] were proposed to meet this requirement. Xu et al. [11] introduced IBBE into IBE to construct an IBBE-based PRE scheme. There also exist many other extensions of the PRE scheme, such as time-based PRE [12], attribute-based PRE [13,14], function-based PRE [14], conditional PRE [15,16], etc.

Ciphertext only can be transformed in the same cryptosystem by deploying the above schemes. In fact, there are many different cryptosystems that have been deployed. A cross-domain re-encryption scheme is also a requirement. Deng et al. [17] linked the identity-based encryption and the identity-based broadcast encryption by allowing the transformation of a ciphertext of the IBE system into a ciphertext of the IBBE system. Jiang et al. [18] constructed a concrete CDSS protocol which allowed the ciphertext to transform from the PKE scheme into an IBE scheme. Obviously, CDSS satisfies the cross-domain property. Döt-

tling et al. [19] proposed a UPRE scheme. It is a general cross-domain proxy re-encryption scheme from an existing PKE scheme to another existing PKE scheme, which does not care about what the PKE scheme is.

However, these PRE schemes mainly provide ciphertext transformations, and the user cannot fully control the delegation path in these multi-hop PRE schemes. To address this problem, Cao et al. [6] constructed an autonomous path PRE scheme to enable a user to designate a path of his preferred data users. However, the scheme needs certificates for the user in the system (for more details, see [6]). All of the data users in the delegation path are selected by the user, and no one can branch off the designated delegation path with meaningful decryption. This paper aims to address autonomous path transformation in an identity-based setting.

1.5. Paper Organization

The paper is organized as follows. In Section 2, we introduce the preliminary knowledge of assumption which our CLAP-PRE scheme is based on. In Section 3, we describe its system and security model. In Section 4, the construction of our CLAP-PRE scheme is presented, followed by security proof and security analysis in Section 5. Finally, Section 6 contains the conclusion of this paper and future work.

2. Preliminaries

2.1. Autonomous Path Proxy Re-Encryption (AP-PRE)

An autonomous path proxy re-encryption is a new kind of proxy re-encryption scheme where the delegator can fully control the delegation path. It is a unidirectional and multi-hop proxy re-encryption. The delegator selects his preferred delegatees with different privileges. The delegator generates re-encryption keys for these delegatees. In an autonomous path proxy re-encryption scheme, the ciphertext under i's public key can only be transformed on the autonomous path Pa_i, which is designated by the delegator i. No one can branch off the delegation path with meaningful decryption. The original ciphertexts can not be inserted into the delegation path which is not designated by the delegator.

2.2. Certificateless Signature

Certificateless cryptography was first proposed by Al-Riyami and Paterson in [20]. Certificateless cryptography is used to fill the gap between public key cryptosystems and identity-based cryptosystems. Certificateless cryptography does not require a certificate authority (CA) to ensure the authenticity of public keys and only relies on a trusted third party (KGC) who has the master-key. In this paper, we use the certificateless signature scheme to guarantee the authenticity of public keys. We use the certificateless signature scheme proposed by Choi et al. [21], which satisfies the requirements of certificateless signature schemes as defined in [20].

2.3. Bilinear Map

We briefly review the necessary facts about bilinear maps and bilinear map groups. Let G and G_T be two cyclic multiplicative groups of the same prime order $q > 2^\lambda$, where λ is the security parameter, and let g be the generator of the group G. We define that $e : G \times G \to G_T$ is a bilinear or pairing map if it satisfies the following conditions:

- Bilinear: For all $a, b \in Z_q$, $g \in G$, $e(g^a, g^b) = e(g, g)^{ab}$;
- Non-degenerate: $e(g, g) \neq 1_{G_T}$, i.e., if $G =< g >$, then $G_T =< e(g, g) >$;
- Computable: The map e is efficiently computable.

Generally, the map e is obtained from Tate or Weil pairings.

2.4. Multilinear Maps

We briefly review multilinear maps and multilinear groups (for more details, see [22–24]). Consider the following setting: given a security parameter λ and prime $q > 2^\lambda$, a k-multilinear map consists of k cyclic groups $(G_1, \ldots, G_k)$ of prime order q, and g_i

is a generator of group G_i. There exists a set of bilinear maps $\{e_{ij} : G_i \times G_j \to G_{i+j}, |i, j \geq 1 \wedge i + j \leq k\}$. The k-multilinear map should satisfy the following conditions:

- Given that g_i and g_j, then $g_{i+j} = e_{ij}(g_i, g_j)$;
- For all $a, b \in Z_q^*$, $e_{ij}(g_i^a, g_j^b) = e_{ij}(g_i, g_j)^{ab}$;
- The map e_{ij} is efficiently computable.

Since the multilinear maps is clear to us, we omit the indexes of pairing $e_{i,j}$, i.e., $e(g_i^a, g_j^b) = g_{i+j}^{ab}$.

2.5. Complexity Assumptions

2.5.1. Computational Bilinear Diffie–Hellman Assumption

Consider two cyclic groups $G \times G \to G_T$ of prime order q and a map e. The CBDH problem states that, given a tuple (g, g^a, g^b, g^c), where g is a generator which is randomly chosen and random $a, b \in \{0, \ldots, q-1\}$, it is computationally intractable to compute the value $e(g, g)^{abc}$.

2.5.2. k-Multilinear Decisional Diffie–Hellman (k-MDDH) Assumption

The k-multilinear decisional Diffie–Hellman (k-MDDH) problem states the following: a challenger runs $PairGen(1^\lambda, k)$ to generate groups and generators of order q. Then, it chooses $b, a_1, a_2, \ldots, a_k \in Z_q$ randomly.

The assumption then states that given $g, g^b, g^{a_1}, \ldots, g^{a_k}$, it is hard to distinguish $T = g_k^{b \prod_{j=1}^{k} a_j}$ from a random group element in G_k with the non-negligible advantage ϵ (in the security parameter λ).

3. System and Security Model

3.1. System Model

Here, we use Figure 3 to describe the system model for the proposed CLAP-PRE scheme. The model contains three entities: the key generation center (KGC), the data owner (the data user) and the proxy.

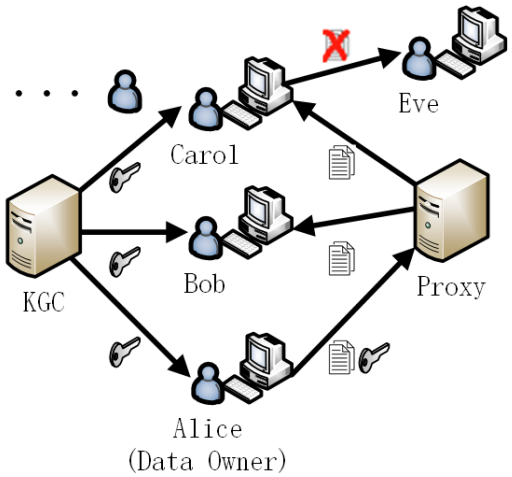

Figure 3. Data Sharing in Cloud.

3.1.1. Key Generation Center

The KGC provides a partial private key for the data owner and data users. It is wholly trusted. Different from the traditional certificateless encryption, we use the signature generated by the user to demonstrate that the identity and the public parameters belong to the same user.

3.1.2. Proxy

The proxy is equipped with a database that stores the encrypted data and the re-encryption keys for different data users generated by the different data owners. It performs the data re-encryption with corresponding re-encryption keys on the delegation path. The proxy is considered as honest but curious.

3.1.3. Data User/Data Owner

The data owner generates the delegation path along with the re-encryption keys and encrypted data, then uploads them to the proxy. The data user first queries for data access, then gets transformed encrypted data from the proxy. After receiving the encrypted data, the data user only can decrypt the data and has no right to transform the encrypted data to the user who is not in the delegation path. Note that in a multi-hop proxy re-encryption, the data user can transform the encrypted data to his own delegatee.

3.2. Security Model

In this part, we give the definition of the security model to prove that our scheme is an IND-CPA secure CLAP-PRE scheme under the *k*-multilinear decisional Diffie–Hellman (*k*-MDDH) assumption in the random oracle model.

- SETUP: The challenger generates the system's public parameters and gives them to the adversary;
- FIND PHASE: The adversary is allowed to make queries $\mathcal{O}_{pk}(ID_i)$, $\mathcal{O}_{sk}(ID_i)$, $\mathcal{O}_{pa}(ID_i)$, $\mathcal{O}_{rk}(ID_i)$;
  - On $\mathcal{O}_{pk}(ID_i)$, return $H_1(ID_i)$ and public parameters of ID_i;
  - On $\mathcal{O}_{sk}(ID_i)$, return the decryption keys of ID_i;
  - On $\mathcal{O}_{pa}(ID_i)$, return the delegation path of ID_i;
  - On $\mathcal{O}_{rk}(ID_i)$, return the re-encryption keys of delegation path generated by ID_i.

  At the end of this phase, the adversary submits two equal length messages M_0, M_1 and an identity ID. The adversary $\mathcal{A}$ is restricted to choices of ID such that the decryption keys of ID has not been queried on $\mathcal{O}_{sk}(ID_i)$, and the adversary $\mathcal{A}$ cannot translate the ciphertext from ID to ID', for which $\mathcal{A}$ holds the decryption keys by using re-encryption keys extracted during this phase;
- CHALLENGE: The challenger randomly choose a value $\mu \in \{0, 1\}$ and returns the ciphertext of M_μ under the delegator's public key ID;
- GUESS: $\mathcal{A}$ makes queries as in the FIND phase with the same restrictions. At the end of this phase, the adversary submits a guess, $\mu' \in \{0, 1\}$, of μ.

If $\mu' = \mu$, the $\mathcal{A}$ wins the game. $\mathcal{A}$'s advantage in the above game, $Adv_{\mathcal{A}}^{IND-CLAPPRE-CPA}$, is defined as $|Pr[\mu' = \mu] - 1/2|$. We say that the certificateless autonomous path proxy re-encryption scheme is IND-CLAPPRE-CPA-secure if for all probabilistic polynomial time algorithms $\mathcal{A}$, $Adv_{\mathcal{A}}^{IND-CLAPPRE-CPA} \leq v(\lambda)$.

4. CLAP-PRE Scheme

4.1. Definition

In this paper, we use λ to denote the security parameter. A CLAP-PRE scheme is a tuple of algorithms as follows.

- $Setup(1^\lambda) \rightarrow (par, msk)$: The algorithm's input is the system's security parameter λ, and the algorithm's output are the public parameters *par* of the cryptosystem and the master secret key *msk* of the cryptosystem;
- $PPKE(par, msk, ID_i) \rightarrow d_{ID_i}$: The algorithm's inputs are the public parameters *par* of the cryptosystem, the master secret key *msk* of the cryptosystem and a user's identity ID_i, the key generation center (KGC) outputs the partial private key d_{ID_i} for the user ID_i;

- $UKG(par, d_{ID_i}) \rightarrow (PK_{ID_i}, SK_{ID_i})$: The algorithm's inputs are the public parameters par of the cryptosystem and the partial private key d_{ID_i} generated by KGC, the user outputs the public key PK_{ID_i} and private key SK_{ID_i};
- $Proof(par, SK_{ID_i}) \rightarrow proof$: The algorithm's inputs are the public parameters par of the crptosystem and the private key SK_{ID_i}. The user outputs the $proof$ to guarantee the authenticity of public key;
- $Verify(par, proof) \rightarrow \{0, 1\}$: The algorithm's inputs are the public parameters par of the cryptosystem and the $proof$, as well as the algorithm outputs 0 or 1. With output value 1, we say that $proof$ is a valid proof. Additionally, this means that the identity ID_i and the corresponding public key PK_{ID_i} belong to the same user;
- $CreatePath(par, ID_i) \rightarrow (Pa_{ID_i}, l_{ID_i})$: The algorithm's inputs are the public parameters par of the cryptosystem and the identity ID_i of the delegator, and the algorithm's output are an autonomous delegation path Pa_{ID_i} of length l_{ID_i}. The autonomous delegation path $Pa_{ID_i} = \{ID_0 = ID_i, ID_1, \ldots, ID_{l_{ID_i}}\}$ designated by the delegator is a list of ordered l_{ID_i} different public keys. All of the keys in the sequence must be unique;
- $RKGen(par, Pa_{ID_i}) \rightarrow RK_{ID_i}$: The algorithm's inputs are the public parameters par of the cryptosystem and the delegation path Pa_{ID_i} designated by the delegator; the algorithm outputs l_{ID_i} re-encryption keys $(rk_{ID_i}^{0\rightarrow 1}, \ldots, rk_{ID_i}^{l_{ID_i}-1\rightarrow l_{ID_i}})$ and sends them to the proxy in a secure way;
- $Enc(par, ID_i, m) \rightarrow C_{ID_i}^0 = (c_{00}, c_{01}, c_{02}, c_{03})$: The algorithm's inputs are the public parameters par of the cryptosystem, the delegator's identity ID_i and a message m from the message space M, and the algorithm outputs the ciphertext $C_{ID_i}^0 = (c_{00}, c_{01}, c_{02}, c_{03})$ encrypted with the delegator's public key ID_i.
- $ReEnc(par, Pa_{ID_i}, rk_{ID_i}^{j\rightarrow j+1}, C_{ID_i}^j) \rightarrow C_{ID_i}^{j+1} = (c_{(j+1)0}, c_{(j+1)1}, c_{(j+1)2}, c_{(j+1)3})$: The algorithm's inputs are the public parameters par of the cryptosystem, the delegation path $Pa_{ID_i} = (ID_0 = ID_i, ID_1, \ldots, ID_{l_{ID_i}})$, the re-encryption key $rk_{ID_i}^{j\rightarrow j+1}$ from the delegatee j to the delegatee $j+1$ and the ciphertext $C_{ID_i}^j$ sent to the delegatee j. It outputs the ciphertext $C_{ID_i}^{j+1}$ sent to the delegatee $j+1$ who is in the delegation path Pa_{ID_i}.
- $Dec(par, SK_{ID_j}, C_{ID_i}^j) \rightarrow m$: The algorithm's inputs are the public parameters par of the cryptosystem, a ciphertext $C_{ID_i}^j$ and the corresponding secret key SK_{ID_j}; it outputs the plaintext m in the message space M.

4.2. System Flow

First, every user in the cryptosystem needs to send his own identity to KGC and request KGC to generate a partial private key. The KGC runs the algorithm $PPKE$ and sends the partial private key to the user in a secure way. After receiving the partial private key, the user runs the algorithm UKG to generate his own private key and public key. Then the user generates the proof of his public key using the algorithm $Proof$. Everyone can validate the proof using the algorithm $Verify$ without interacting with others.

Second, when a data owner wants to share his data with others, the data owner runs the algorithm *CreatePath* to create a delegation path in his preferred order. The data owner generates the ciphertext and re-encryption keys using the algorithm *Enc* and the algorithm *RKGen*. After that, the data owner sends the delegation path, the ciphertext and the re-encryption keys to the proxy.

Third, the proxy recovers the identities from the delegation path and the re-encryption keys from set RK_{ID_i}, then runs the algorithm *ReEnc* to re-encrypt the ciphertext. The proxy needs to query the delegatee if he wants to deal with the encrypted data. If so, the proxy sends the ciphertext to the delegatee; otherwise, the proxy finds the next delegatee along with the corresponding re-encryption key and repeats until a delegatee wants to decrypt.

4.3. Construction

Let $PairGen(1^\lambda)$ be an algorithm that, on input of a security parameter 1^λ, outputs a tuple $\gamma = (q, G_1, G_2, G_3, G_4, e_{ij})$, where G_1, G_2, G_3 and G_4 are the cyclic group with the same prime order q and $\{e_{ij} : G_i \times G_j \to G_{i+j}, |i, j \geq 1 \wedge i + j \leq 4\}$ are efficient non-degenerate multilinear maps such that for all $a, b \in Z_q, e(g_i^a, g_j^b) = e(g_i, g_j)^{ab}$.

- $Setup(1^\lambda) \to (par, msk)$: The algorithm runs the group generator algorithm $PairGen(1^\lambda)$ and gets the groups and the multilinear mapping description $\gamma = (q, G_1, G_2, G_3, G_4, e_{ij})$, $G_i =< g_i >$. The system's parameters are generated as follows. We choose a master secret key $\alpha \in Z_q$ randomly, set $A = g_2^\alpha \in G_2$ and choose random elements u_1 in G_1, B in G_2. Let $H_1 : \{0,1\}^* \to G_1$, $H_2 : \{0,1\}^* \to G_2$, $H_3 : G_2 \to Z_q^*$, $H_4 : \{0,1\}^* \to Z_q^*$ be four collision-resistant hash functions. Thus, we can get the public parameters $par = (q, G_1, G_2, G_3, G_4, e_{ij}, u_1, A, B, H_1, H_2, H_3, H_4)$ and the master secret key $msk = \alpha$, which is kept secretly by the key generation center (KGC);
- $PPKE(par, msk, ID_i) \to d_{ID_i}$: Upon input of the public parameters par of the cryptosystem, the system's master secret key msk and a user's identity ID_i, the key generation center (KGC) computes the partial private key as follows. First, the KGC chooses a random value $r \in Z_q$. Then KGC computes g_2^r, $B^\alpha e(u_1, H_1(ID_i))^r$ and $H_2(ID_i)^\alpha$. The partial private key of the user i with identity ID_i is formed as:

$$d_{ID_i} = \{B^\alpha e(u_1, H_1(ID_i))^r, g_2^r, H_2(ID_i)^\alpha\};$$

- $UKG(par, d_{ID_i}) \to (PK_{ID_i}, SK_{ID_i})$: Upon input of the public parameters par of the cryptosystem and the partial private key d_{ID_i} generated by KGC, the user computes PK_{ID_i} and SK_{ID_i} as follows. First, the user randomly chooses a secret value $x \in Z_q^*$ and sets $PK_{ID_i} = \{ID_i, X = g_2^x, Y = B^x\}$; $SK_{ID_i} = \{SK_{ID_i0}, SK_{ID_i1}, SK_{ID_i2}, SK_{ID_i3}\} = \{B^\alpha e(u_1, H_1(ID_i))^r, g_2^r, H_2(ID_i)^{\frac{\alpha}{xH_3(X)}}, x\}$.
- $Proof(par, SK_{ID_i}) \to proof$: Upon input of the public parameters par of the cryptosystem and the private key SK_{ID_i}, the user i randomly chooses a message p and a value s. The $proof$ is computed as follows:

$$U = H_2(ID_i)^s, V = (SK_{ID_i2})^{sH_4(p,U)},$$

$$proof = \{p, PK_{ID_i}, U, V\};$$

- $Verify(par, proof) \to \{0,1\}$: Upon input of the public parameters par of the cryptosystem and the $proof$, the algorithm outputs 1 if and only if:

$$e(V, X^{H_3(X)}) == e(U^{H_4(p,U)}, A),$$

and

$$e(X, B) == e(g_2, Y);$$

- $CreatePath(par, ID_i) \to (Pa_{ID_i}, l_{ID_i})$: Upon input of the public parameters par of the cryptosystem and the identity ID_i of the delegator, it outputs an autonomous delegation path Pa_{ID_i} of length l_{ID_i}. The autonomous delegation path $Pa_{ID_i} = \{ID_0 = ID_i, ID_1, \ldots, ID_{l_{ID_i}}\}$ designated by the delegator is a sequence of ordered l_{ID_i} different public keys. All of the keys in the sequence must be unique;
- $RKGen(par, Pa_{ID_i}) \to RK_{ID_i}$: Upon input of the public parameters par of the cryptosystem and the delegation path Pa_{ID_i} designated by the delegator, the re-encryption keys are computed as follows:
for $j = 1$,

$$rk_{ID_i}^{0 \to 1} = H_1(ID_1)^x,$$

for $j \geq 2$,

$$rk_{ID_i}^{j \to j+1} = (\frac{H_1(ID_{j+1})}{H_1(ID_j)})^x;$$

- $Enc(par, ID_i, m) \to C_{ID_i}^0$: To encrypt the message $m \in M$, the algorithm selects $t \leftarrow_R Z_q^*$ and computes

$$c_{00} = m \cdot e(A, Y)^t, c_{01} = B^t,$$
$$c_{02} = u_1^t, c_{03} = X^t,$$
$$C_0^{ID_i} = (c_{00}, c_{01}, c_{02}, c_{03}).$$

The second ciphertext can be transformed along with the delegation path;

- $ReEnc(par, rk_{ID_i}^{j \to j+1}, C_{ID_i}^j) \to C_{ID_i}^{j+1}$: To transform a ciphertext encrypted with the public key ID_j into the one encrypted with ID_{j+1} in the delegation path Pa_{ID_i}, the proxy computes as follows using the re-encryption key $rk_{ID_i}^{j \to j+1}$
for $j = 0$,

$$c_{10} = c_{00}, c_{11} = c_{02}, c_{12} = e(c_{02}, rk_{ID_i}^{0 \to 1}), c_{13} = c_{03}$$
$$C_{ID_i}^1 = (c_{10}, c_{11}, c_{12}, c_{13}).$$

for $j \geq 1$,

$$c_{(j+1)0} = c_{j0}, c_{(j+1)1} = c_{j1},$$
$$c_{(j+1)2} = c_{j2} \cdot e(c_{j1}, rk_{ID_i}^{j \to j+1}), c_{(j+1)3} = c_{j3}.$$
$$C_{ID_i}^{j+1} = (c_{(j+1)0}, c_{(j+1)1}, c_{(j+1)2}, c_{(j+1)3});$$

- $Dec(par, SK_{ID_j}, C) \to m$: For the original ciphertext formed as $C_{ID_j}^0 = (c_{00}, c_{01}, c_{02}, c_{03})$, the decryption is

$$m = \frac{c_{00}}{e(A^{SK_{ID_j 3}}, c_{01})}.$$

For the transformed ciphertext formed as $C_{ID_i}^j = (c_{j0}, c_{j1}, c_{j2}, c_{j3})$, the decryptiopn is

$$m = c_{j0} \cdot \frac{e(SK_{ID_j 3}, c_{j2})}{e(c_{j3}, SK_{ID_j 0})}.$$

The proof of correctness can be found in Appendix A.

5. Security

5.1. Security Proof

Our CLAP-PRE scheme in Section IV is IND-CLAPPRE-CPA secure under the k-MDDH assumption in the random oracle model.

Proof. Suppose $\mathcal{A}$ can break our CLAP-PRE scheme with the non-negligible advantage ϵ in polynomial time in Section IV. There must another adversary $\mathcal{B}$ that can break the k-MDDH assumption with a non-negligible advantage by interacting with $\mathcal{A}$.

Adversary $\mathcal{B}$ accepts a tuple $(g_1, g_1^a, g_1^b, g_1^c, g_1^d, g_1^e, T)$ as input. We say that the tuple $(g_1, g_1^a, g_1^b, g_1^c, g_1^d, g_1^e, g_4^{abcde})$ is a k-MDDH instance. $\mathcal{B}$ outputs 1 which indicates $T = g_4^{abcde}$; otherwise, $\mathcal{B}$ outputs 0. In order to simplify the proof of the scheme, we use the techniques proposed in [25,26] to construct our $\mathcal{O}_{pk}(ID_i)$ and $\mathcal{O}_{sk}(ID_i)$. $\mathcal{B}$ interacts with adversary $\mathcal{A}$ in the IND-CLAPPRE-CPA game as follows:

- SETUP. $\mathcal{B}$ runs $PairGen(1^\lambda)$ and gives the system's parameters $par = (q, G_1, G_2, G_3, G_4, e_{ij}, g_i, A = e(g_1^a, g_1^b) = g_2^{ab}, u_1 = g_1^f, B = e(u_1, g_1^c), H_1)$, $f \in_R Z_q^*$ to the adversary $\mathcal{A}$. The master secret key is ab which is unknown to $\mathcal{B}$.;

- FIND PHASE. In this phase, adversary $\mathcal{A}$ issues some queries, and $\mathcal{B}$ answers these queries as follows:
  - First, $\mathcal{B}$ takes $ID^* \leftarrow \{ID_1, ID_2, \ldots, ID_n\}$ at random. Adversary $\mathcal{B}$ guesses that ID^* would be challenged by adversary $\mathcal{A}$ in the following phase;
  - On a $\mathcal{O}_{pk}(ID_i)$ query, $\mathcal{B}$ first randomly selects $x_i, y_i \in Z_q^*$. If $ID_i = ID^*$, $\mathcal{B}$ sets $H_1(ID_i) = g_1^{y_i/f}$, $X = g_2^{dx_i} = e(g_1, g_1^d)^{x_i}$ and $Y = B^{dx_i} = e(u_1, g_1^c)^{dx_i} = e(g_1^d, g_1^c)^{fx_i}$. In this case, the user's private key is $\{B^{ab} e(u_1, H_1(ID_i))^r, g_2^r, dx_i\}$, which is unknown to $\mathcal{B}$. Otherwise, $\mathcal{B}$ sets $H_1(ID_i) = g_1^c g_1^{y_i/f}$, $X = g_2^{x_i}$ and $Y = B^{x_i}$.In this case, $\mathcal{B}$ does not know the user's private key;
  - On a $\mathcal{O}_{pa}(ID_i)$ query, for the initial query of identity ID_i, $\mathcal{B}$ creates a delegation path for ID_i and gives it to the adversary. Otherwise, $\mathcal{B}$ returns $\perp$;
  - On a $\mathcal{O}_{sk}(ID_i)$ query, if $ID_i = ID^*$, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ computes

$$\begin{aligned} e(u_1, H_1(ID_i))^r &= (B g_2^{y_i})^r \\ &= B^{ab} B^{-ab} (B g_2^{y_i})^r \\ &= B^{ab} A^{y_i} (B g_2^{y_i})^{r-ab} \\ B^{ab} (B g_2^{y_i})^{r-ab} &= A^{-y_i} e(u_1, H_1(ID_i))^r \\ g_2^{r-ab} &= g_2^r A^{-1} \end{aligned}$$

    and gives them and x_i to the adversary;
  - On a $\mathcal{O}_{rk}(ID_i)$ query, if $ID_i \neq ID^*$, this means adversary $\mathcal{B}$ knows the user's private key, and $\mathcal{B}$ runs $RKGen(par, Pa_{ID_i})$ to generate re-encryption keys. If $ID_i = ID^*$, this means adversary $\mathcal{B}$ does not know the user's private key, and $\mathcal{B}$ computes the re-encryption key rk^{ID_i} as follows:
    - $\mathcal{B}$ selects $Z_1, \ldots, Z_{l_{ID_i}} \leftarrow_R G_1$;
    - $\mathcal{B}$ sets $rk_{ID_i}^{0 \to 1} = Z_1$. Note that adversary $\mathcal{A}$ can not distinguish the real view and simulated view, because $rk_{ID_i}^{0 \to 1} = H_1(ID_1)^{SK_{ID_{i3}}} \in G_1$;
    - $\mathcal{B}$ sets $rk_{ID_i}^{j \to j+1} = \frac{Z_{j+1}}{Z_j}$.

    Finally, $\mathcal{B}$ gives the re-encryption keys to the adversary;
- CHALLENGE. Adversary $\mathcal{B}$ can end the QUERY phase at any time. $\mathcal{A}$ outputs a delegator user with identity ID and two messages $m_0, m_1 \in G_4$ of equal length. If $ID \neq ID^*$, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ selects $\mu \leftarrow \{0, 1\}$ at random. Finally, $\mathcal{B}$ computes the original challenge ciphertext and text as follows:

$$C_{ID}^0 = (m_\mu \cdot e(A, B^{dx})^e = m_\mu \cdot T^{fx}, B^e = e(g_1^e, g_0^c)^f,$$

$$u_1^e = g_1^{fe}, X^e = g_2^{dxe} = e(g_1, g_1)^{dxe} = e(g_1^d, g_1^e)^x)$$

$$C_{ID_i}^{ID} = (m_\mu \cdot e(A, B^{dx_i})^e = m_\mu \cdot T^{fx_i},$$

$$e(u_1^e, H_1(ID)^{dx_i}) = e(g_1^{ef}, g_1^{dyx_i/f})$$

$$u_1^e = g_1^{fe}, X^e = g_2^{dx_i e} = e(g_1, g_1)^{dx_i e} = e(g_1^d, g_1^e)^{x_i})$$

    C_{ID}^0 is a valid original ciphertext, and $C_{ID_i}^{ID}$ is a valid re-encrypted ciphertext in the system. At the end, $\mathcal{B}$ sends them to $\mathcal{A}$ as the challenge ciphertext;
- GUESS. Adversary $\mathcal{A}$ can repeat the FIND phase with the same restrictions. At the end of this phase, $\mathcal{A}$ returns a guess μ', where $\mu' \in \{0, 1\}$. If $\mu = \mu'$, $\mathcal{B}$ returns 1, it indicates that $T = g_4^{abcde}$; otherwise, $\mathcal{B}$ returns 0, in which case it indicates that T is a random element in G_4.

If $\mathcal{B}$ aborts in the simulation, $\mathcal{B}$ randomly returns 0 or 1. The adversary $\mathcal{A}$ can not distinguish the real world and the simulation. Thus, the challenge $\mathcal{B}$'s advantage in solving

the k-MDDH problem is at least $\frac{\epsilon}{n}(1-\frac{q}{n})$, where ϵ is the non-negligible probability with which $\mathcal{A}$ can break our CLAP-PRE scheme. □

5.2. Security Analysis

- We use the certificateless signature scheme to guarantee the authenticity of public keys. If the signature published by the user is valid, the authenticity of public key is verified. However, the certificateless signature scheme in [21] has only one public key (X in our scheme), so we use another equation to guarantee the authenticity of another public key (Y in our scheme). This equation demonstrates that X and Y have the same power x.
- Anyone who wants to fork the delegation path with meaningful decryption (i.e. from $j \rightarrow (j+1)'$) must first compute $rk_{ID_i}^{j\rightarrow(j+1)'} = (H_1(ID_{j-1}/H_1(ID_j)))^x$ and then compute $e(u_1^t, rk_{ID_i}^{j\rightarrow(j+1)'}) = e(u_1^t, (\frac{H_1(ID_{(j+1)'})}{H_1(ID_j)})^x)$. It is obvious that computing this is the same difficulty as CBDH problem without knowing x. That means no one can fork the delegation path with a meaningful decryption even if the data user is on the delegation path.

6. Conclusions

In this paper, we have presented a certificateless autonomous path proxy re-encryption scheme which combines the advantages of an autonomous path PRE and certificateless encryption. We first put forward the concept of double public keys in the autonomous path proxy re-encryption scheme. In order to fully control the delegation path, we only transform the ciphertext under one of the public keys, and the transformed ciphertext still includes the information of another public key, so that the ciphertext is still under control of the data owner.

Author Contributions: Writing—original draft preparation, C.R.; validation, Y.Z.; writing—review and editing, J.S.; supervision, X.D. and Z.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Key Research and Development Program of China (Grant No. 2020YFA0712300), in part by the National Natural Science Foundation of China (Grant No. 62172162, 62132005), in part by Shanghai Trusted Industry Internet Software Collaborative Innovation Center. Xiaolei Dong, Zhenfu Cao, and Jiachen Shen are the corresponding authors.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Appendix A. Correctness of CLAP-PRE Scheme

In this section, we will demonstrate the correctness of the proposed CLAP-PRE Scheme, including the verification of the proof and the decryption of the ciphertext.

- Verification:

$$\begin{aligned} e(V, X^{H_3(X)}) =& e((SK_{ID_i 2})^{sH_4(p,U)}, X^{H_3(X)}) \\ =& e((H_2(ID_i)^{\frac{\alpha}{xH_3(X)}})^{sH_4(p,U)}, (g_2^x)^{H_3(X)}) \\ =& e(H_2(ID_i)^{\alpha sH_4(p,U)}, g_2) \\ =& e(H_2(ID_i)^{sH_4(p,U)}, g_2^\alpha) \\ =& e((H_2(ID_i)^s)^{H_4(p,U)}, A) \\ =& e(U^{H_4(p,U)}, A) \end{aligned}$$

$$\begin{aligned} e(X, B) =& e(g_2^x, B) \\ =& e(g_2, B^x) \\ =& e(g_2, Y) \end{aligned}$$

- Decryption:
  - Case 1:

$$\begin{aligned} m =& \frac{c_{00}}{e(A^{SK_{ID_j 3}}, c_{01})} \\ =& \frac{m \cdot e(A, B^x)^t}{e(A^x, B^t)} \\ =& \frac{m \cdot e(A, B)^{xt}}{e(A, B)^{xt}} \end{aligned}$$

  - Case 2:

$$\begin{aligned} m =& c_{j0} \cdot \frac{e(SK_{ID_j 1}, c_{j2})}{e(c_{j3}, SK_{ID_j 0})} \\ =& c_{j0} \cdot \frac{e(g_2^r, e(u_1, H_1(ID_j))^{xt})}{e(X^t, B^\alpha e(u_1, H_1(ID_i))^r)} \\ =& c_{j0} \cdot \frac{e(g_2, e(u_1, H_1(ID_i)))^{rxt}}{e(A, B)^{xt} \cdot e(g_2, e(u_1, H_1(ID_i))^{rxt})} \\ =& \frac{m \cdot e(A, B)^{xt}}{e(A, B)^{xt}} \end{aligned}$$

References

1. Blaze, M.; Bleumer, G.; Strauss, M. Divertible Protocols and Atomic Proxy Cryptography. EUROCRYPT. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 1998; Volume 1403, pp. 127–144.
2. Ateniese, G.; Fu, K.; Green, M.; Hohenberger, S. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Trans. Inf. Syst. Secur.* **2006**, *9*, 1–30. [CrossRef]
3. Tang, F.; Li, H.; Chang, J. Multi-Hop Unidirectional Proxy Re-Encryption from Multilinear Maps. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **2015**, *98-A*, 762–766.
4. Wang, H.; Cao, Z. More efficient CCA-secure unidirectional proxy re-encryption schemes without random oracles. *Secur. Commun. Netw.* **2013**, *6*, 173–181. [CrossRef]
5. Green, M.; Ateniese, G. Identity-Based Proxy Re-encryption. ACNS. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4521, pp. 288–306.
6. Cao, Z.; Wang, H.; Zhao, Y. AP-PRE: Autonomous path proxy re-encryption and its applications. *IEEE Trans. Dependable Secur. Comput.* **2017**, *16*, 833–842. [CrossRef]
7. Libert, B.; Vergnaud, D. Unidirectional Chosen-Ciphertext Secure Proxy Re-encryption. Public Key Cryptography. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2008; Volume 4939, pp. 360–379.
8. Guo, H.; Zhang, Z.; Xu, J.; An, N.; Lan, X. Accountable Proxy Re-Encryption for Secure Data Sharing. *IEEE Trans. Dependable Secur. Comput.* **2021**, *18*, 145–159. [CrossRef]
9. Wang, H.; Cao, Z.; Wang, L. Multi-use and unidirectional identity-based proxy re-encryption schemes. *Inf. Sci.* **2010**, *180*, 4042–4059. [CrossRef]
10. Shao, J.; Cao, Z. Multi-use unidirectional identity-based proxy re-encryption from hierarchical identity-based encryption. *Inf. Sci.* **2012**, *206*, 83–95. [CrossRef]
11. Xu, P.; Jiao, T.; Wu, Q.; Wang, W.; Jin, H. Conditional Identity-Based Broadcast Proxy Re-Encryption and Its Application to Cloud Email. *IEEE Trans. Comput.* **2016**, *65*, 66–79. [CrossRef]
12. Yang, Y.; Ma, M. Conjunctive Keyword Search with Designated Tester and Timing Enabled Proxy Re-Encryption Function for E-Health Clouds. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 746–759. [CrossRef]
13. Ge, C.; Susilo, W.; Fang, L.; Wang, J.; Shi, Y. A CCA-secure key-policy attribute-based proxy re-encryption in the adaptive corruption model for dropbox data sharing system. *Des. Codes Cryptogr.* **2018**, *86*, 2587–2603. [CrossRef]
14. Liang, K.; Au, M.H.; Liu, J.K.; Susilo, W.; Wong, D.S.; Yang, G.; Phuong, T.V.X.; Xie, Q. A DFA-Based Functional Proxy Re-Encryption Scheme for Secure Public Cloud Data Sharing. *IEEE Trans. Inf. Forensics Secur.* **2014**, *9*, 1667–1680. [CrossRef]

15. Chu, C.; Weng, J.; Chow, S.S.M.; Zhou, J.; Deng, R.H. Conditional Proxy Broadcast Re-Encryption. ACISP. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5594, pp. 327–342.
16. Weng, J.; Deng, R.H.; Ding, X.; Chu, C.; Lai, J. Conditional proxy re-encryption secure against chosen-ciphertext attack. In Proceedings of the 4th International Symposium on Information, Computer, and Communications Security, Sydney, Australia, 10–12 March 2009; pp. 322–332.
17. Deng, H.; Qin, Z.; Wu, Q.; Guan, Z.; Deng, R.H.; Wang, Y.; Zhou, Y. Identity-Based Encryption Transformation for Flexible Sharing of Encrypted Data in Public Cloud. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3168–3180. [CrossRef]
18. Jiang, P.; Ning, J.; Liang, K.; Dong, C.; Chen, J.; Cao, Z. Encryption Switching Service: Securely Switch Your Encrypted Data to Another Format. *IEEE Trans. Serv. Comput.* **2021**, *14*, 1357–1369. [CrossRef]
19. Döttling, N.; Nishimaki, R. Universal Proxy Re-Encryption. Public Key Cryptography (1). In *Lecture Notes in Computer Science*; Springer: Cham, Switzerland, 2021; Volume 12710, pp. 512–542.
20. Al-Riyami, S.S.; Paterson, K.G. Certificateless Public Key Cryptography. ASIACRYPT. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2894, pp. 452–473.
21. Choi, K.Y.; Park, J.H.; Hwang, J.Y.; Lee, D.H. Efficient Certificateless Signature Schemes. ACNS. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4521, pp. 443–458.
22. Boneh, D.; Silverberg, A. Applications of Multilinear Forms to Cryptography. *IACR Cryptol. ePrint Arch.* **2002**, *324*, 80.
23. Coron, J.; Lepoint, T.; Tibouchi, M. Practical Multilinear Maps over the Integers. CRYPTO (1). In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 8042, pp. 476–493.
24. Garg, S.; Gentry, C.; Halevi, S. Candidate Multilinear Maps from Ideal Lattices. EUROCRYPT. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 7881, pp. 1–17.
25. Waters, B. Efficient Identity-Based Encryption without Random Oracles. EUROCRYPT. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3494, pp. 114–127.
26. Boneh, D.; Boyen, X. Secure Identity Based Encryption without Random Oracles. CRYPTO. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3152, pp. 443–459.