electronics

MDPI

Article

Secure and Efficient Data Sharing Scheme Based on Certificateless Hybrid Signcryption for Cloud Storage

Wei Luo * and Wenping Ma

State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China; wp_ma@mail.xidian.edu.cn

* Correspondence: rovid008@163.com

Received: 23 April 2019; Accepted: 20 May 2019; Published: 27 May 2019

Abstract: As cloud service providers are not completely trusted, people are increasingly concerned about security issues such as data confidentiality and user privacy. In many existing schemes, the private key generator (PKG) generates a full private key for each user, which means that the PKG can forge a valid signature or decrypt the ciphertext. To address the issue, we first present a novel certificateless hybrid signcryption (CL-HSC) scheme without pairing, in which the PKG only generates the partial private keys for users. It is provably secure under the Elliptic Curve Computational Diffie-Hellman (EC-CDH) assumption in the random oracle model. Then, we propose a key derivation method by which the data owner only needs to maintain the master key to get rid of the complex key management. By combining our proposed CL-HSC scheme and the key derivation method, we present a secure and efficient data-sharing scheme for cloud storage, which can resist collusion attacks, spoofing attacks, and replay attacks and makes user revocation easier. In addition, compared with some existing schemes, our scheme has a lower computational complexity.

Keywords: cloud storage; data encryption; access control; authentication; certificateless hybrid signcryption

1. Introduction

With the development in networking technology and the increasing need for data storage and computing resources, cloud storage and cloud computing are gradually popularized. Users and companies can outsource data storage and computing to remote servers (called clouds), which greatly reduces the storage pressure and the calculation pressure of users and companies.

In the public cloud, the cloud service provider supplies a platform for all users who have registered in the cloud. Each user can store data and share data with others in the cloud. By using the cloud, it can be achieved easily that users access the data anytime and anywhere.

Although cloud storage brings many benefits, there are many security issues caused by the openness of the cloud platform and the lack of security mechanisms. Much of data stored in the cloud may be sensitive, such as, social networks and medical records. Thus, it is crucial to prevent unauthorized access to these data and to realize secure data sharing. At present, most cloud service providers do not take any security measure for protecting users' data, which means that users' data is visible to cloud service providers. They even abuse users' data to get more benefits. Furthermore, the attacker can attack the cloud servers to obtain users' private data. The confidentiality of data outsourced is important to the data owner. In addition, users may share their data with friends or the requesters. Therefore, a flexible secure access control scheme is absolutely needed.

Many existing schemes utilize the private key generator (PKG), which owns the master key used to generate all users' private keys. Therefore, the private key of any user is known to the PKG, which

can decrypt the ciphertexts or can impersonate some users. To solve this problem, Al-Riyami et al. proposed an efficient method to divide the private key into two parts [1].

In this study, we present a novel certificateless hybrid signcryption scheme with a low computation and propose a secure and efficient data-sharing scheme for cloud storage. Our main contributions are as follows:

- A secure cloud storage model is proposed, which employs a *semi-trusted third party* (STTP) to provide services, such as user management, key management, and data processing. Cloud storage servers are physically isolated, and only the STTP can directly access them.
- A novel certificateless hybrid signcryption scheme with a low computation is proposed.
- A key derivation method is constructed by using partial iteration to release users from the complex key management.

The rest of this paper is organized as follows: Section 2 introduces related works on cloud security. The proposed system model of cloud storage and assumptions are detailed in Section 3. Our proposed scheme is described in detail in Section 4. Section 5 presents a security analysis of our scheme. We discuss the performance of our scheme in Section 6. We conclude our paper in Section 7.

2. Related Works

In Reference [2], Chen et al. presented a comprehensive analysis of data security and privacy preserving issues associated with all phases of the data life cycle in cloud computing, discussed some schemes, and showed future research work. This paper focuses on the security of data storage and data sharing in cloud storages. The following is a review of some related research work, mainly involving the application of symmetric encryption, identity-based encryption, and attribute-based encryption in cloud storage.

Wang et al. proposed to encrypt every data block with a different key to achieve a flexible cryptography-based access control in Reference [3]. This scheme uses a symmetric cryptographic algorithm to encrypt data and employs the pre-shared secret key to achieve mutual authentication. However, this scheme assumes that the data owner shares the pairwise keys with the service provider and the end users, which requires a lot of storage space. Ramesh et al. introduced a encryption/decryption method based on a secure e-stream cipher ChaCha20 to protect the user's sensitive data for cloud storage in Reference [4].

Kumar et al. proposed a secure method based on elliptic curve cryptography (ECC) in Reference [5], which provides users secure storage and access to the data from the cloud storage servers. They adopt ECC to protect data files. Lu et al. presented a certificated-based proxy re-encryption scheme without pairing for data sharing in cloud storage, which combines the advantages of certificate-based encryption and proxy-based encryption in Reference [6]. However, this scheme does not perform user authentication. Li et al. proposed a new authentication protocol based on the identity-based hierarchical model for cloud computing in Reference [7]. Tang et al. in Reference [8] presented an inter-domain identity-based proxy re-encryption (IBPRE) scheme, which achieves secure data sharing between users in different domains. However, Han et al. proved the collusion attack against Tang's scheme in Reference [9] and proposed an identity-based data storage scheme that prevents collusion attacks and supports intra-domain and inter-domain queries. Wang et al. presented a new IBPRE+ to achieve controlled secure social cloud data sharing in Reference [10]. In the scheme, the re-encryption keys are generated by a delegator, which can reduce the computational complexity of the data owner.

In Reference [11], Ruj et al. presented a privacy-preserving authenticated access control scheme for cloud storage, which is decentralized and can achieve the authenticity of users without knowing users' identities. Later, they proposed a novel decentralized access control scheme for data storage in the cloud which supports anonymous authentication in Reference [12]. In this scheme, attribute-based encryption (ABE) was used to encrypt data and attribute-based signature (ABS) proposed by Maji et al. in Refrence [13] was used to achieve authentication. The existing work [14–17] on access control in

data outsourcing systems uses ABE to achieve security. However, all the schemes introduced above adopt the pairing operation, which requires much computational complexity. For the shared data files with hierarchical structures, Wang et al. presented an efficient data-sharing scheme based on ciphertext-policy attribute-based encryption (CP-ABE) in Reference [18]. Huang et al. claimed that they presented an identity-based private data-sharing scheme in online social networks in Reference [19]. However, this scheme was actually a data-sharing scheme based on ABE.

Since most of the above schemes are constructed based on bilinear pairing, their computational complexity is large. Therefore, in this paper, we design a data-sharing scheme based on a certificateless hybrid signcryption scheme without pairing.

3. Background

3.1. Assumptions

Before discussing in detail, we make some assumptions as follows:

- In our cloud storage model, we employ a semi-trusted third party (STTP) to manage and maintain cloud storage servers. Thus, the STTP is interested in viewing users' content but cannot modify it.
- Users registered can store their data in the cloud. Once the data owner verified requesters, they can read data stored in the cloud.

Elliptic Curve Computational Diffie-Hellman Problem (EC-CDH) [20]

Let G_p be an ECC group of order p, where p is a prime; the point P is the generator of G_p. The elliptic curve computational Diffie-Hellman problem in G_p: Given a random instance $(P, aP, bP) \in G_p$, compute abP. Let A^{EC-CDH} be an adversary. We define A^{EC-CDH}'s advantage in solving the EC-CDH by $Adv(A^{EC-CDH}) = Pr[A^{EC-CDH}(P, aP, bP) = abP]$.

3.2. System Model

Figure 1 shows an overview of our cloud storage model. The model contains four entities: the STTP, cloud storage servers, the data owner, and requesters. The STTP is the cloud administrator to manage and maintain cloud storage servers and users. Cloud storage servers are used to store and backup the ciphertext of users' data. The data owner stores the ciphertexts of data in the cloud storage servers, which can reduce the storage pressure and achieve accessing and sharing whenever and wherever. Requesters want to share the data stored in the cloud.

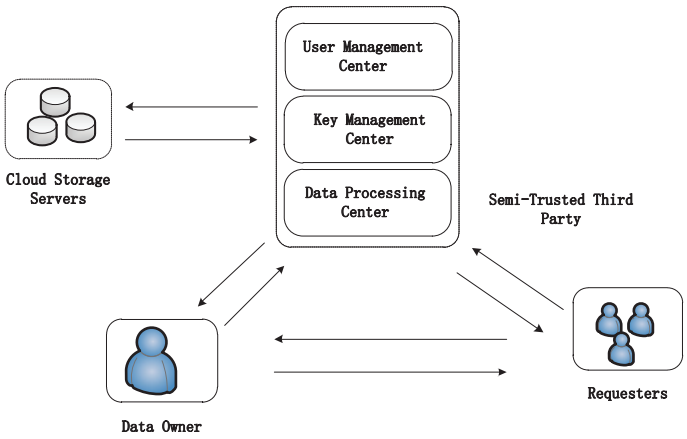

Figure 1. The system model.

The STTP is mainly composed of three parts: user management center, key management center, and data processing center. The user management center is responsible for the new user registration and user revocation. It always maintains the user list *UL* and the revocation list *RL* containing all users'

identities and public keys in the cloud storage system. The key management center is in charge of generating the partial private key for each user. The data processing center takes charge of storing and reading the ciphertext from cloud storage servers.

In our cloud storage model, only the STTP can directly access cloud storage servers, which can guarantee the security of cloud storage servers. To some extent, cloud storage servers are physically isolated.

4. Secure Data Sharing Scheme

We firstly describe our key derivation method and a novel certificateless hybrid signcryption scheme. Then, we present our secure data-sharing scheme based on the proposed CL-HSC algorithm and the re-encryption protocol in detail. The proposed scheme can provide data confidentiality, user authentication, message integrity, and non-repudiation.

4.1. Key Derivation Algorithm

Here, we employ the Advanced Encryption Standard (AES) algorithm for data encryption, which is mature, fast, and strongly secure [21]. The security of data mainly depends on the keys for the AES encryption algorithm. Due to the large number of data blocks, it is difficult for the data owner to manage encryption keys. To address this issue, we present a key derivation method, which employs a hash algorithm SHA256 to generate the encryption keys. The main flow chart of the data encryption keys generation algorithm is shown in Figure 2. We briefly describe the notations used in Figure 2 (See Table 1). The mathematical description of the algorithm is as follows:

$$V_i = SHA256(mk \parallel T_i), i = 1, 2, ..., n$$

$$K_i = SHA256(V_{i-1}) \oplus V_i, i = 1, 2, ..., n \quad (1)$$

where n denotes the number of files.

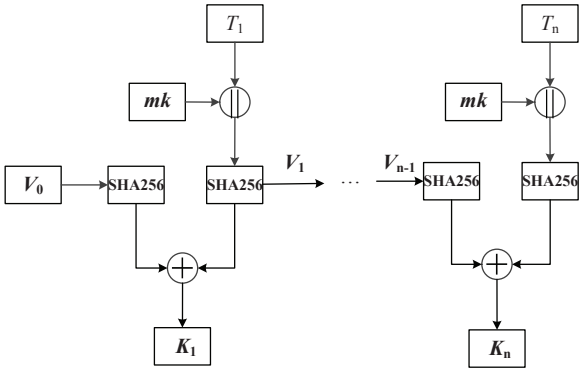

Figure 2. The Data Encryption Keys Generation Algorithm.

Table 1. The list of notations in Figure 2.

Notation	Meaning
V_0	a 256-bit random number
mk	the master key of user
T_i	time stamp
$\parallel$	the concatenation operation
SHA256	a hash algorithm
V_i	the intermediate value
$+$	the operation of XOR (exclusive OR)
K_i	i-th data encryption key

We adopt the partial iteration method to generate encryption keys so that the adjacent keys have a correlation. The master key and time stamp are added during each key generation to enhance the security of encryption keys.

4.2. A Novel Certificateless Hybrid Signcryption

Based on the schemes of References [20,22–24], we present a novel certificateless hybrid signcryption scheme, which is a certificateless public key cryptography [1].

The proposed CL-HSC scheme is comprised of six probabilistic polynomial-time algorithms: *setup*, *set secret value*, *extract partial private key*, *set private key*, *signcrypt*, and *de-signcrypt*.

Setup

This algorithm takes the security parameter λ as the input and returns the system parameters Ω and the master key *msk*. This algorithm is run by the STTP. Given λ, the STTP performs the following steps:

1. Choose a λ-bit prime p and return the tuple $\{p, F_p, G_p, P\}$, where G_p is an additive cyclic group consisting of points on an elliptic curve over the field F_p and P is the generator of G_p.
2. Choose the master key $x \in Z_p^*$ randomly, and compute the system public key $P_{pub} = xP$.
3. Choose cryptographic hash functions $H_0 : \{0,1\}^* \times G_p \rightarrow Z_p^*$, $H_1 : G_p \times G_p \rightarrow \{0,1\}^n$, $H_2 : G_p \times \{0,1\}^* \times \{0,1\}^* \times G_p \rightarrow Z_p^*$ and $H_3 : G_p \times \{0,1\}^* \times \{0,1\}^* \times G_p \rightarrow Z_p^*$. Here, n represents the length of the agreed symmetric key.
4. Publish $\Omega = \{F_p, G_p, P, P_{pub}, H_0, H_1, H_2, H_3\}$ as the system parameters, and keep the master key x secret.

Set Secret Value

Each user executes this algorithm. The user with an identity ID_i chooses $x_{ID_i} \in Z_p^*$ randomly as his secret value and generates the public key as $P_{ID_i} = x_{ID_i}P$.

Extract Partial Private Key

The STTP runs this algorithm, which takes his master key, the identity, and the public key of the user as the input and outputs the partial private key for the user. The STTP computes $d_{ID_i} = xH_0(ID_i, P_{ID_i}) \ mod \ p$ as the partial private key of the user.

The user can validate the partial private key by checking whether $d_{ID_i}P = H_0(ID_i, P_{ID_i})P_{pub}$ holds.

Set Private Key

The user takes the pair $sk_{ID_i} = (d_{ID_i}, x_{ID_i})$ as his full private key.

Signcrypt

This algorithm is executed by the sender ID_s to compute the signature and the symmetric key for the receiver ID_r. Given the time stamp τ which is used to prevent replay attacks, the sender obtains the signature φ_{ID_s} and the symmetric key K as follows:

1. Choose $l_{ID_s} \in Z_p^*$ randomly, and compute $S_{ID_s} = l_{ID_s}P$.
2. Compute $H = H_2(S_{ID_s}, \tau, ID_s, P_{ID_s})$, $H' = H_3(S_{ID_s}, \tau, ID_r, P_{ID_r})$, and $W_{ID_s} = d_{ID_s} + l_{ID_s} \cdot H + x_{ID_s} \cdot H' \ mod \ p$.
3. Compute $T_{ID_s} = l_{ID_s} \cdot H_0(ID_r, P_{ID_r})P_{pub}$ and $K = H_1(T_{ID_s}, l_{ID_s} \cdot P_{ID_r})$.
4. Output $\varphi_{ID_s} = (S_{ID_s}, W_{ID_s})$ and K.

De-signcrypt

Given the signature φ_{ID_s}, the time stamp τ, the signer's identity ID_s, and full public key pk_{ID_s}, the signature φ_{ID_s} is verified as follows:

1. Compute $H = H_2(S_{ID_s}, \tau, ID_s, P_{ID_s})$ and $H' = H_3(S_{ID_s}, \tau, ID_r, P_{ID_r})$.
2. If $W_{ID_s} \cdot P = H_0(ID_s, P_{ID_s}) \cdot P_{pub} + H \cdot S_{ID_s} + H' \cdot P_{ID_s}$, the signature φ_{ID_s} is valid. Then the receiver ID_r computes the agreed symmetric key $K = H_1(T_{ID_s}, x_{ID_r} \cdot S_{ID_s})$, where T_{ID_s} is computed by

$$\begin{aligned} T_{ID_s} &= d_{ID_r} \cdot S_{ID_s} \\ &= (xH_0(ID_r, P_{ID_r}) \ mod \ q) \cdot l_{ID_s} P \\ &= l_{ID_s} \cdot H_0(ID_r, P_{ID_r}) P_{pub} \end{aligned} \tag{2}$$

Otherwise, the output is invalid.

4.3. Scheme Construction

The proposed secure scheme mainly includes six phases: *user register*, *data store*, *data share*, *user revocation*, *data update*, and *data deletion*.

User Register

1. Generation of System Parameters: The STTP executes the *Setup* algorithm, publishes the system parameters Ω, and keeps the master key x secret.
2. Distribution of Keys: The STTP assigns a unique identifier ID_i to each user and runs the *Extract Partial Private Key* algorithm to generate the partial private key (sk_{ID_i}, pk_{ID_i}). Finally, the STTP generates a user list UL containing identifiers and public keys of registered users. It also creates a revocation list RL for managing the revoked users. In this phase, the STTP must communicate with users through a secure channel, e.g., Email.

Data Store

Once registered, users can store their data in the cloud. To protect privacy and confidentiality, users can encrypt their data and store the ciphertext to the cloud servers.

A symmetric cryptographic algorithm is relatively suitable for data encryption, which presents very good reading and writing performances. Thus, we employ the AES encryption algorithm to encrypt data. The data owner chooses the master key mk to generate encryption keys by using the proposed key derivation algorithm and encrypts each file F_i with encryption key K_i. Then, concatenate ciphertext and corresponding keyword and send the result and the signature $\varphi_{ID_O} = (S_{ID_O}, W_{ID_O})$ to the STTP. Adding the keywords is conducive to improving requesters' ability to quickly retrieve the ciphertext they want. Finally, the data owner encrypts the data encryption keys and corresponding keywords by using mk, keeps the ciphertext in the local, and deletes the plaintext. The mathematical description of the process is as follows:

$$C_{F_i} = AES256(K_i, F_i) \parallel kw_i, i = 1, 2, ..., n$$

$$C_K = AES256(mk, \bigcup_{i=1}^{n} kw_i \parallel K_i)$$

After receiving the signature φ_{ID_O}, the time stamp τ, and the ciphertext $C_{F_i} (i = 1, 2, ..., n)$, the STTP verifies the signature and then sends the ciphertext to the cloud storage servers.

Data Share

Based on the proposed CL-HSC algorithm and the idea of the re-encryption method [25–27], we propose a secure data-sharing scheme, which can provide data confidentiality and user authentication. Meanwhile, our scheme avoids directly sending data encryption keys to requesters, which makes user revocation easy.

1. The requester ID_R indexes the keywords kw_i in the cloud storage servers to find the corresponding ciphertext C_{F_i} and obtains the identifier ID_O and the public key pk_{ID_O} of the data owner from the STTP.
2. The requester ID_R computes a signature $\varphi_{ID_R} = (S_{ID_R}, W_{ID_R})$ and the shared key k_{RO} and sends $\{\varphi_{ID_R}, ID_R, kw_i\}$ to the data owner ID_O .
3. The data owner ID_O queries pk_{ID_R} from the STTP and validates the signature after receiving the request. Then, the data owner ID_O reads kw_i and makes a decision. If the data owner ID_O agrees to share data F_i with the requester ID_R, they compute the shared key k_{RO}. Otherwise, stop further operation.
4. The data owner ID_O sends a share message $Share(kw_i)$ to the STTP. The STTP gets the ciphertext C_{F_i} from cloud storage servers and sends C_{F_i} to the data owner.
5. The data owner obtains K_i from C_K with his/her master key mk and decrypts C_{F_i} by using K_i. Then, the data is re-encrypted by using the agreed key k_{RO} and sends the new ciphertext C'_{F_i} to the requester ID_R.
6. The requester ID_R receives ciphertext C'_{F_i} and decrypts it by using the agreed key k_{RO} to obtain the data. Simultaneously, an acknowledgement message *Acknowledge* is sent to the data owner ID_O.

User Revocation

Now, we discuss how to handle user revocation. Based on the data-share scheme, we propose a simple user revocation protocol as below.

1. Once a malicious user ID_i is found, the STTP revokes the ID_i from the user list UL and adds the ID_i to the revocation list RL. When a user is revoked, the STTP will reject providing any services for the ID_i. After a period of time, the STTP can delete the data of the revoked user.
2. The STTP sends a revocation message {Revoke, ID_i} to other users.
3. After receiving the message, users firstly validate the message. If the message is valid, users delete the agreed key and stop further operations with the revoked user ID_i.

Data Update

The data owner may need to update the data stored in the cloud servers. The data update operation is described as follows:

1. The data owner ID_O sends a data update message {Update, kw_i, ID_O, φ_{ID_O}} to the STTP.
2. After receiving the message, the STTP validates φ_{ID_O}. If the message is valid, the STTP sends the ciphertext C_{F_i} to the data owner ID_O.
3. The data owner ID_O decrypts the ciphertext C_{F_i} by using K_i. Then, the data owner ID_O can update the data F_i. After updating, the data owner encrypts the modified data again and sends the new ciphertext to the STTP.
4. The STTP receives the ciphertext and stores it to the cloud servers.

Data Deletion

In addition, the data owner maybe want to delete some data stored in the cloud servers. The data deletion operation is described as follows:

1. The data owner ID_O sends a data deletion message {Delete, kw_i, ID_O, φ_{ID_O}} to the STTP.
2. After receiving the message, the STTP validates the φ_{ID_O}. If the message is valid, the STTP deletes the ciphertext C_{F_i} and sends an acknowledgement message *Acknowledge* to the data owner ID_O.

5. Security Analysis

In this section, we discuss the security of our data-sharing scheme, which mainly depends on data encryption keys and the proposed CL-HSC scheme. We prove that our proposed data-sharing scheme is collusion resistant and spoofing attacks resistant.

5.1. Security of Data Encryption Keys

In our proposed key derivation method, the master key, the initial vector, and the time stamp are used to generate data encryption keys. We associate the adjacent keys to enhance the security. Each key is generated by XORing two hash results, one of which is the hash result of a part of the former key.

Wang et al. introduced a key derivation method based on binary tree [3], and the mathematical description of the key derivation method is as follows:

- left child of $K(i,j)$: $K((i+1),(2*j-1)) = hash(K(i,j)||(2*j-1)||K(i,j))$.
- right child of $K(i,j)$: $K((i+1),(2*j)) = hash(K(i,j)||(2*j)||K(i,j))$.

This method cannot provide the security of data encryption keys. Once the adversary has obtained the $K(0,1)$, the other data encryption keys can be obtained by the derivation formulae. In our key generation algorithm, each data encryption key cannot be directly generated by other data encryption keys. Furthermore, data encryption keys are confidential for the STTP and requesters in our scheme.

5.2. Security of the Proposed CL-HSC Scheme

The security of our proposed CL-HSC scheme is based on the computation assumption EC-CDH. To prove the security of a CL-HSC scheme, two types of adversaries are defined in Reference [1]. We show that our proposed CL-HSC scheme without pairing operations is satisfied with an existential unforgeability against adaptive chosen messages and identity attacks (EUF-CMA), which can be guaranteed by the following theorems. The proof process is similar to the one in Reference [20,28] and will not be discussed here.

Theorem 1. *If a forger $\mathcal{F}_I$ can break the EUF-CMA-I security of our CL-HSC scheme with a non-negligible advantage ϵ, asking q_{ppk} extract partial private key queries, q_{sv} set secret value queries, and q_{H_i} random oracle queries to $H_i(i = 0,1,2,3)$, then an algorithm $\mathcal{C}$ can be constructed that can solve the EC-CDH problem with the following advantage:*

$$\epsilon' \geq (\epsilon - \frac{1}{2^{\lambda-1}})(1 - \frac{q_{ppk}}{q_{H_0}})(1 - \frac{q_{sv}}{q_{H_0}})\frac{1}{q_{H_0} - (q_{ppk} + q_{sv})}$$

Theorem 2. *If a forger $\mathcal{F}_{II}$ can break the EUF-CMA-II security of the CL-HSC scheme with a non-negligible advantage ϵ, asking q_{sv} set secret value queries, q_{rpk} replace public key queries, and q_{H_i} random oracle queries to $H_i(i = 0,1,2,3)$, then an algorithm $\mathcal{C}$ can be constructed that can solve the EC-CDH problem with the following advantage:*

$$\epsilon' \geq (\epsilon - \frac{1}{2^{\lambda-1}})(1 - \frac{q_{rpk}}{q_{H_0}})(1 - \frac{q_{sv}}{q_{H_0}})\frac{1}{q_{H_0} - (q_{rpk} + q_{sv})}$$

5.3. Security of Our Proposed Data-Sharing Scheme

Theorem 3. *Our proposed data-sharing scheme can resist the collusion attack.*

Proof of Theorem 3. In our cloud storage system, data are encrypted by using AES256, which can guarantee the confidentiality of users' data. Currently, there are no known practical attacks to decrypt AES-encrypted data. The data encryption keys are encrypted by the master key, so only the data owner can obtain them. Therefore, the requester cannot collude with the STTP to decrypt the ciphertext. In addition, when sharing data, the original ciphertext is re-encrypted by the agreed key. This means that multiple requesters cannot collude to decrypt the ciphertext. □

Theorem 4. *Our proposed data-sharing scheme is spoofing attack resistant and replay attack resistant.*

Proof of Theorem 4. In our data-sharing scheme, mutual authentication is needed before undergoing a further operation. The signature φ_{ID_i} is computed by using the user ID_i's full private key sk_{ID_i}, which is secret to others, even the STTP. The STTP only generates the partial public/private key pair for all users, which can prevent the STTP from forging the signature to obtain the shared key and the data. Thus, the signature cannot be forged, which can effectively resist the spoofing attacks. Meanwhile, the time stamp τ is used to generate the signature, which can ensure that our scheme is replay attack resistant. □

6. Performance Analysis

6.1. Experiment

To evaluate the performance of our proposed CL-HSC scheme, we performed experimental simulations using the PBC library [29] and the OpenSSL library. The experiments were conducted on a Windows 7 system with an Inter(R) Core(TM) i5-3470 CPU at 3.20 GHz and 4.00 GB RAM. We choose the Type A pairings to implement the three schemes. We conducted five simulation experiments for each scheme and used the average run time as the final result.

From the simulation results (shown in Figure 3), it can be seen that although our scheme requires more execution time in Unsigncrypt, our scheme has a significant advantage in KegGen, PartialKeyGen, and Signcrypt. Compared with the schemes in Reference [20,24], our scheme has a lower total running time.

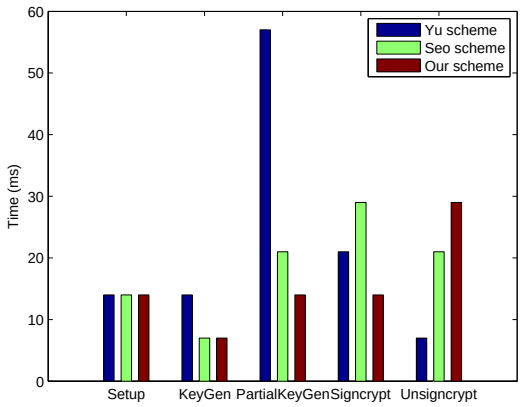

Figure 3. The experiment results.

6.2. Comparison with Other Existing Schemes

Table 2 presents the notations used in different operations.

Table 2. The notations for different operations.

Notation	Meaning
PM	Scalar point multiplication on G_p
T_H	Time to a normal hash function H
T_{H_0}	Time to hash function H_0
T_{H_1}	Time to hash function H_1
T_{H_2}	Time to hash function H_2
T_{H_3}	Time to hash function H_3
T_E	Time to AES256 algorithm
T_h	Time to SHA256 algorithm
$\|q\|$	Length of the binary representation of q
$\|C_{F_i}\|$	Size of the ciphertext C_{F_i}
SKC	Symmetric Key Cryptography

First, we make a comparison based on the work mechanism and functions as shown in Table 3. The scheme of Reference [3] and our scheme are centralized architecture, but the scheme of Reference [12] is decentralized and has multiple key distribution centers (KDCs) or STTPs for key management. In fact, our scheme also can be extended to a multi-domain environment and each sub-domain has a STTP as a domain agent. The scheme of Reference [12] supports multiple reads and writes on the data stored in cloud. For security and copyright considerations, our scheme only allows the data owner to write data. Also, our scheme provides authentication and user revocation.

Table 3. Comparison of Our Scheme with Other Existing Secure Schemes.

Schemes	Structure	Write/Read	Access Control	Authentication	User Revocation
[3]	Centralized	1-W-M-R	SKC	Yes	No
[12]	Decentralized	M-W-M-R	ABE	Yes	Yes
Our Scheme	Centralized	1-W-M-R	SKC	Yes	Yes

Then, we compare the computational complexity of our scheme with that of other existing schemes as shown in Table 4–6. Here, we assume that the AES256 algorithm and a normal hash function H are used in Reference [3].

In Table 4, we show the computational complexity of different schemes in the data store phase or creating a file. The complexity of our scheme is roughly the same as that of Reference [3]. However, the scheme of Reference [12] based on bilinear maps has a high computational complexity, which is positively correlated with the dimension of the access matrix. Note that our scheme and that of Reference [3] are in the case of n files but that of Reference [12] is in the case of one file.

Table 4. Comparison of Computational Complexity and Size of Ciphertext in the Data Store Phase.

Schemes	Data Owner	Cloud (STTP)	Size of Ciphertext
[3]	$(n+2^{\lceil \log_2^n \rceil})T_H + (n+1)T_E$	$T_H + T_E$	$\sum \lvert C_{F_i} \rvert$
[12]	$(3m+1)E_0 + 2mE_T + \tau_H$ $(2l+2)E_1 + 2tE_2 + \tau_{H'}$	$(l+2t)\tau_{\hat{p}} + l(E_1+E_2) + \tau_{H'}$	$2m\lvert G_0 \rvert + m\lvert G_T \rvert + m^2 +$ $\lvert MSG \rvert + (l+t+2)\lvert G_1 \rvert$
Our Scheme	$2nT_h + (n+1)T_E$ $PM + T_{H_2} + T_{H_3}$	$T_{H_0} + T_{H_2} + T_{H_3} + 3PM$	$2\lvert q \rvert + \sum \lvert C_{F_i} \rvert$

In Table 5, we show the computational complexity of different schemes in the data share phase. Since the scheme of Reference [12] is based on ABE, which allows users whose attributes meet the access policy to access data stored in cloud, the data owner and the STTP do not need to do anything in the data share phase. Compared with the schemes of Reference [3,12], our scheme has a high computational complexity. However, in our scheme, the data owner has strict control over data in the cloud.

Table 5. Comparison of Computational Complexity in the Data Share Phase.

Schemes	Data Owner	Cloud (STTP)	Requester
[3]	$3T_E + 3T_H$	$T_E + T_H$	$2T_E + T_H$
[12]	0	0	$2m\tau_p + \tau_H + O(mh)$
Our Scheme	$T_{H_0} + T_{H_1} + T_{H_2} + T_{H_3} + 4PM + 2T_E$	0	$T_{H_0} + T_{H_1} + T_{H_2} + T_{H_3} + 2PM + T_E$

In Table 6, we show the computational complexity of different schemes in the data update phase. This process is the same as the data store phase. Here, we just discuss the case of one file. Although the computational complexity of the scheme in Reference [3] is less than our scheme, the data owner needs to maintain two key trees in the block-update phase. Furthermore, the data owner sends the data encryption keys to the requesters directly in the scheme of Reference [3], which leads to much computational complexity in the phases of block deletion, block update, and revocation of access right.

Table 6. Comparison of Computational Complexity in the Data Update Phase.

Schemes	User	Cloud (STTP)
[3]	$3T_E + T_H$ (Case1) $3T_E + 3T_H$ (Case2)	$T_E + T_H$
[12]	$(3m+1)E_0 + 2mE_T + \tau_H$ $(2l+2)E_1 + 2tE_2 + \tau_{H'}$	$(l+2t)\tau_{\hat{p}} + l(E_1 + E_2) + \tau_{H'}$
Our Scheme	$PM + T_{H_2} + T_{H_3} + 2T_E$	$T_{H_0} + T_{H_2} + T_{H_3} + 3PM$

7. Conclusions

In this paper, we firstly proposed a key derivation method by which the data owner just needs to maintain the master key. Then, we combined the proposed CL-HSC scheme and a re-encryption protocol to propose a secure data share scheme, which is satisfied with confidentiality, unforgeability, and user-centricity. Since the proposed CL-HSC scheme do not contain pairing operation, it is low cost and low complexity. As our future work, we will consider the problem of secure data sharing for multiple recipients.

Author Contributions: Conceptualization, W.L.; methodology, W.L.; software, W.L.; validation, W.L.; formal analysis, W.L.; investigation, W.L.; resources, W.L.; data curation, W.L.; writing—original draft preparation, W.L.; writing—review and editing, W.L.; visualization, W.L.; supervision, W.M.; project administration, W.M.; funding acquisition, W.M.

Funding: This research was funded by the National Key R&D Program of China (No. 2017YFB0802400), the National Natural Science Foundation of China under grant No. 61373171, and the 111 Project under grant No. B08038.

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

References

1. Al-Riyami, S.S.; Paterson, K.G. Certificateless public key cryptography. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, 30 November 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 452–473.
2. Chen, D.; Zhao, H. Data security and privacy protection issues in cloud computing. In Proceedings of the 2012 International Conference on Computer Science and Electronics Engineering, Hangzhou, China, 23–25 March 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 647–651.
3. Wang, W.; Li, Z.; Owens, R.; Bhargava, B. Secure and efficient access to outsourced data. In Proceedings of the 2009 ACM workshop on Cloud Computing Security (CCSW '09), Chicago, IL, USA, 13 November 2009; ACM: New York, NY, USA, 2009; pp. 55–66.
4. Ramesh, D.; Mishra, R.; Edla, D.R. Secure Data Storage in Cloud: An e-Stream Cipher-Based Secure and Dynamic Updation Policy. *Arab. J. Sci. Eng.* **2017**, *42*, 873–883. [CrossRef]
5. Kumar, A.; Lee, B.G.; Lee, H.; Kumari, A. Secure storage and access of data in cloud computing. In Proceedings of the 2012 International Conference on ICT Convergence (ICTC), Jeju Island, Korea, 15–17 October 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 336–339.
6. Lu, Y.; Li, J. A pairing-free certificate-based proxy re-encryption scheme for secure data sharing in public clouds. *Future Gener Comput. Syst.* **2016**, *62*, 140–147. [CrossRef]
7. Li, H.; Dai, Y.; Tian, L.; Yang, H. Identity-based authentication for cloud computing. In Proceedings of the IEEE International Conference on Cloud Computing, Beijing, China, 1–4 December 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 157–166.
8. Tang, Q.; Hartel, P.; Jonker, W. Inter-domain Identity-Based Proxy Re-encryption. In Proceedings of the Information Security and Cryptology, Beijing, China, 14–17 December 2008; Springer: Berlin/Heidelberg, Germany, 2009; pp. 332–347.
9. Han, J.; Susilo, W.; Mu, Y. Identity-based data storage in cloud computing. *Future Gener. Comput. Syst.* **2013**, *29*, 673–681. [CrossRef]
10. Wang, X.A.; Xhafa, F.; Ma, J.; Zheng, Z. Controlled secure social cloud data sharing based on a novel identity based proxy re-encryption plus scheme. *J. Parallel Distrib. Comput.* **2019**, *130*, 153–165. [CrossRef]

11. Ruj, S.; Stojmenovic, M.; Nayak, A. Privacy preserving access control with authentication for securing data in clouds. In Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Ottawa, ON, Canada, 13–16 May 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 556–563.
12. Ruj, S.; Stojmenovic, M.; Nayak, A. Decentralized access control with anonymous authentication of data stored in clouds. *IEEE Trans. Parallel Distrib. Comput.* **2014**, *25*, 384–394. [CrossRef]
13. Maji, H.K.; Prabhakaran, M.; Rosulek, M. Attribute-Based Signatures: Achieving Attribute-Privacy and Collusion-Resistance. *IACR Cryptol. ePrint Arch.* **2008**, 328.
14. Wang, G.; Liu, Q.; Wu, J. Hierarchical Attribute-Based Encryption for Fine-Grained Access Control in Cloud Storage Services. In Proceedings of the 17th ACM conference on Computer and communications security, Chicago, IL, USA, 4–8 October 2010; ACM: New York, NY, USA, 2010; pp. 735–737.
15. Zhao, F.; Nishide, T.; Sakurai, K. Realizing Fine-Grained and Flexible Access Control to Outsourced Data with Attribute-Based Cryptosystems. In Proceedings of the International Conference on Information Security Practice and Experience, Guangzhou, China, 30 May 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 83–97.
16. Hur, J.; Noh, D.K. Attribute-Based Access Control with Efficient Revocation in Data Outsourcing Systems. *IEEE Trans. Parallel Distrib. Comput.* **2011**, *22*, 1214–1221. [CrossRef]
17. Wan, Z.; Liu, J.; Deng, R.H. HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 743–754. [CrossRef]
18. Wang, S.; Zhou, J.; Liu, J.K.; Yu, J.; Chen, J.; Xie, W. An efficient file hierarchy attribute-based encryption scheme in cloud computing. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 1265–1277. [CrossRef]
19. Huang, Q.; Yang, Y.; Fu, J. PRECISE: Identity-based private data sharing with conditional proxy re-encryption in online social networks. *Future Gener. Comput. Syst.* **2018**, *86*, 1523–1533. [CrossRef]
20. Seo, S.; Bertino, E. *Elliptic Curve Cryptography Based Certificateless Hybrid Signcryption Scheme without Pairing*; CERIAS TR 2013-10; CERIAS: West Lafayette, IN, USA, 2013.
21. Knudsen, L.R.; Lai, X.; Preneel, B. Attacks on fast double block length hash functions. *J. Cryptol.* **1998**, *18*, 59–72. [CrossRef]
22. Barbosa, M.; Farshim, P. Certificateless signcryption. In Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security, Tokyo, Japan, 18–20 March 2008; ACM: New York, NY, USA, 2008; pp. 369–372.
23. Won, J.; Seo, S.H.; Bertino, E. A secure communication protocol for drones and smart objects. In Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, Singapore, 2015; ACM: New York, NY, USA, 2015; pp. 249–260.
24. Yu, H.; Yang, B. Low-computation certificateless hybrid signcryption scheme. *Front. Inf. Technol. Electron. Eng.* **2017**, *18*, 928–940. [CrossRef]
25. Blaze, M.; Bleumer, G.; Strauss, M. Divertible protocols and atomic proxy cryptography. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Espoo, Finland, 31 May 1998; Springer: Berlin/Heidelberg, Germany, 1998; pp. 127–144.
26. Green, M.; Ateniese, G. Identity-based proxy re-encryption. In Proceedings of the International Conference on Applied cryptography and network security, Zhuhai, China, 5–8 June 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 288–306.
27. Liu, Q.; Wang, G.; Wu, J. Time-based proxy re-encryption scheme for secure data sharing in a cloud environment. *Inf. Sci.* **2014**, *258*, 355–370. [CrossRef]
28. Selvi, S.S.D.; Vivek, S.S.; Rangan, C.P. Certificateless KEM and hybrid signcryption schemes revisited. In Proceedings of the International Conference on Information Security Practice and Experience, Seoul, Korea, 12–13 May 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 294–307.
29. PBC Library, the Pairing-based Cryptography Library. Available online: http://crypto.stanford.edu/pbc/ (accessed on 13 July 2017).