An Efficient Certificateless Cloud Data Integrity Detection Scheme for Ecological Data

Yong Xie(✉), Muhammad Israr, Zhengliang Jiang, Pengfei Su, Ruoli Zhao, and Ruijiang Ma

Department of Computer Technology and Application, Qinghai University, Xining, China

Abstract. With the development of society, the ecological and environmental problems can not be ignored. To solve the ecological problems facing, a large amount of ecological data is indispensable. It is impractical to store a large amount of ecological data on a local server, but cloud storage can store it well. However, users are faced with the problems of untrustworthy cloud storage providers and vulnerable data. In order to solve related problems, this paper proposes an efficient certificateless cloud data integrity detection scheme for ecological data. The scheme adopts certificateless form to realize the audit, which can efficiently complete the user's audit requirements. The security analysis shows that this scheme can resist type I and type II adversary attacks. The computation and communication cost analysis results show that the efficiency advantage of this scheme is obvious.

Keywords: Integrity detection · Certificateless · Ecological data · Cloud data

1 Introduction

Eco-environmental issues refer to the global environmental pollution and ecological destruction caused by improper human behavior in the process of industrialization, which poses various realistic threats to human survival and development. For example, global climate change, biodiversity reduction, land desertification, ozone layer destruction and so on. The current ecological and environmental problems affecting the world affect the development of the world to a certain extent. Ecological and environmental issues cannot be ignored. With the development of Internet information technology, big data and other technologies have provided strong support for winning the battle against the ecological environment. But in this process, ecological data will increase massively. At this time,

The work was supported in part by the National Natural Science Foundation of China (61862052), and the Science and Technology Foundation of Qinghai Province (2019-ZJ-7065).

J. Xiong et al. (Eds.): MobiMedia 2021, LNICST 394, pp. 278–290, 2021.
https://doi.org/10.1007/978-3-030-89814-4_21

storing data on a local server is no longer an excellent method, and cloud storage can solve this problem well. Cloud storage technology allows users to outsource large amounts of data and store them in the cloud, thereby reducing user storage space management and computing costs. However, data security is also greatly threatened at this time. Untrusted cloud storage providers may tamper with or delete data for profit, or they may lose data due to hardware failure. In summary, it is imperative to ensure that the data stored in the cloud server is complete. An efficient certificate-free cloud data integrity detection scheme for ecological data is proposed in this paper, which can solve this problem. This paper uses a certificateless form, which solve key escrow issues and certificate management issues in the traditional public key cryptosystem and achieves high efficiency on this basis.

The rest of this paper is as follows: Section 2 introduces related work. Section 3 introduces the background. Section 4 introduces the system model. Section 5 introduces the plan. Section 6 proves the correctness of the scheme. Section 7 proves the security of the scheme. Section 8 analyzes the computation cost and communication cost of the scheme. Finally, Sect. 9 is the conclusion of the full text.

2 Related Work

It is not friendly to transmit data to the local server for verification. Ateniese et al. [1] defines provable data ownership (PDP), which promotes data integrity verification on cloud servers. In this scheme, a third-party auditor (TPA) was introduced for public review. Then Ateniese et al. [2] proposed a new scheme for the dynamically set scene, but it was unable to implement the insert operation. Many PDP schemes [3–7] were proposed later. However, schemes [2–7] are designed based on the traditional public key cryptosystem. In these schemes, the certificate authority generates the user's public-private key pair, and the generated certificates are managed by the certificate authority, which may cause certificate management problems.

Many years ago, Shamir et al. [8] proposed identity-based encryption technology. In this scenario, the public key is the identity of the owner, so the certificate is no longer needed. Taking advantage of identity-based encryption technology, many schemes [9–11] have been proposed. Work [9] proposed a new identity-based public audit scheme for aggregated signature, which solves the problem that storage users need to issue certificates before uploading data, which incurs huge costs. Work [10] adopts an effective certificate-based public key setting key management scheme, which combines identity based aggregate signature and public verification to construct a provable data integrity protocol, which reduces the audit time of a single TPA task. Work [11] proposes an identity-based audit scheme for multiple cloud environments. However, the above schemes have key escrow problem. The user's private key is completely generated by KGC. A malicious KGC may impersonate the user and may leak the user's private information, which is a security challenge.

The key escrow problem can be solved in a certificateless way. Work [12] proposed the concept of Certificateless Public Key Cryptography (CLPKC). In the scheme, the user's private key consists of two parts, one part is generated by the user, and the other part is generated by KGC. CLPKC solves the certificate management and key escrow problems in the traditional public key cryptosystem. Subsequently, the scheme [13] was also proposed, but this scheme is vulnerable to attacks by opponents, and some values can be used by opponents to replace the user's public key.

3 Background

3.1 Bilinear Pairings

G_1 is a cyclic group of prime q, where g_1 and g_2 are generators of G_1. There is a bilinear pairing operation $e : G_1 \times G_1 \rightarrow G_2$. The bilinear pair operation satisfies the following three properties.

Bilinear: If there are $a, b \in Z_q^*$ and $g_1, g_2 \in G_1$, then $e(g_1, g_2)^{ab} = e(g_1^a, g_2^b)$.

Non-degeneracy: If there is $P, Q \in G_1$, then $e(P, Q) \neq 1$.

Computability: For $P, Q \in G_1$, $e(P, Q)$ can be calculated by polynomial time algorithm.

In the security proof, we will use the following computation assumptions to construct.

Discrete Logarithm (DL) Problem: given $P, X \in G_1$, select an element $x \in Z_q^*$, the formula $X = xP$ exists.

Computational Diffie-Hellman (CDH) Problem: given $X = xp, Y = yP \in G_1$, select an element $R = xyP$, where $x \in Z_q^*$ and $y \in Z_q^*$ are unknown (Fig. 1).

4 System Model

The system model of this scheme consists of four parts, namely KGC, third-party auditors, cloud storage providers and group users.

The functions of each part are as follows:

Key generation center (KGC): KGC is responsible for generating the system master key, public parameters, and some public and private key pairs for group users and administrators. It is completely credible.

Third party auditor (TPA): Cloud data can be verified by TPA for completeness. And TPA cannot obtain any information in this process. It is completely credible.

Cloud storage provider (CSP): CSP is semi-trusted, and it may try to deceive data owners by forging integrity evidence. It provides sufficient storage space and retrieval functions.

Group users (Users): Users include a group manager and other users. Managers can register and track other users. Registered users can access and update data.

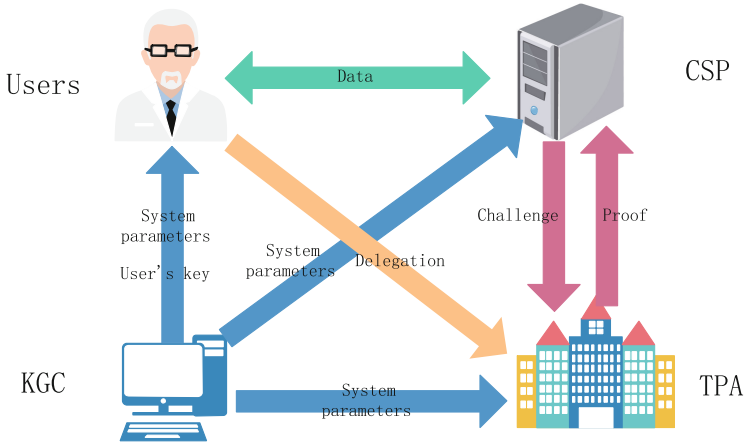

Fig. 1. System model

5 Scheme

The audit scenario for this paper consists of the following six phases.

5.1 Setup

In this section, KGC performs the following steps to generate system parameters.

Enter the security parameter η, and the system parameters are generated by KGC. G_1 is a cyclic group of prime q, where g_1 and g_2 are generators of G_1. There is a bilinear pairing operation $e : G_1 \times G_1 \rightarrow G_2$ and two hash functions $H_1 : \{0,1\}^* \rightarrow Z_q^*$, $H_2 : \{0,1\}^* \rightarrow G_1$ and $H_3 : \{0,1\}^* \rightarrow Z_q^*$. KGC chooses $x \in_R Z_q^*$ as the system master key and calculates the system public key $P_k = {g_1}^x$. The public parameter is $PP = \{G_1, G_2, g_1, g_2, q, H_1, H_2, P_k\}$. KGC discloses PP.

5.2 KeyGen

In this section, KGC and the user perform the following steps in order to generate the user's key.

The user identity information is id_o, and the user randomly selects $x_o \in_R Z_q^*$ and calculates $P_o = {g_1}^{x_o}$. The user sends $\{P_o, id_o\}$ to KGC through a secure channel. KGC randomly selects $t_o \in_R Z_q^*$ and then calculates $T_o = {g_1}^{t_o}$, $h_o = H_1(id_o, T_o, P_o)$ and $s_o = t_o + xh_o \ mod \ q$. KGC sends $\{s_o, T_o\}$ to users through a secure channel. The user uses $\{s_o, x_o\}$ as his private key and $\{P_o, T_o\}$ as his public key and publishes $\{P_o, T_o\}$. The user calculates $\alpha_o = H_3(id_o, P_o, T_o, P_k)$ and makes it public.

5.3 TagGen

This part is executed by the user. Assuming that the data block to be executed is $\{d_1, d_2, \cdots, d_n\}$, the user will generate a tag for each data block. The specific process is as follows

The user calculates $C_i = (H_2(i){g_2}^{d_i})^{s_o+\alpha_o x_o}$, where $i \in \{1, 2, \cdots, n\}$. The user uses $\{i, d_i, C_i\}$ as a tag for data block d_i, and then sends $\{i, d_i, C_i\}$ to CSP.

5.4 ChalGen

In this part, TPA sends a challenge message to the CSP after receiving the audit request sent by the user.

TPA randomly selects J blocks of data from $\{d_1, d_2, \cdots, d_n\}$, among which $J \in \{1, 2, \cdots, n\}$. TPA randomly selects $\tau_j \in_R Z_q^*$ for any $d_j \in \{d_1, d_2, \cdots, d_J\}$, and then sends $\{j, \tau_j\}_{j \in J}$ to CSP.

5.5 ProofGen

In this part, After receiving the challenge message sent by the TPA, the CSP calculates the evidence, and then sends the generated evidence to the TPA.

CSP calculates $C = \prod_{j=1}^{J} (C_j)^{\tau_j}$, where $d = \sum_{j=1}^{J} \tau_j d_j$. CAP sends $\{C, d\}$ to TPA.

5.6 ProofVerif

In this section, TPA will verify the evidence after receiving the evidence sent by the CSP.

The TPA verifies whether the formula $e(C, g_1) = e((\prod_{j=1}^{J} (H_2(j))^{\tau_j}) g_2^d, T_o P_k^{h_o} P_o^{\alpha_o})$ is established. If it is established, it means that the data is correctly saved by the CSP; otherwise, the data may be tampered with or lost.

6 Correctness Analysis

Theorem 1: Confirm that the TPA verifies that the integrity part of the data is correct.

Proof: If the formula $e(C, g_1) = e((\prod_{j=1}^{J} (H_2(i))^{\tau_j}) g_2^d, T_o P_k^{h_o} P_o^{\alpha_o})$ holds, it means that the TPA verification information is partly correct. The formula is derived

as follows.

$$\begin{aligned} e(C, g_1) &= e(\prod_{j=1}^{J}(C_j)^{\tau_j}, g_1) \\ &= e(\prod_{j=1}^{J}((H_2(j){g_2}^{d_j})^{s_o+\alpha_o x_o})^{\tau_j}, g_1) \\ &= e(\prod_{j=1}^{J}(H_2(j)^{\tau_j} g_2^{d_j\tau_j}), g_1^{s_o+\alpha_o x_o}) \\ &= e((\prod_{j=1}^{J}(H_2(j))^{\tau_j}) g_2^{\sum_{j=1}^{J} d_j\tau_j}, T_o P_k^{h_o} P_o^{\alpha_o}) \\ &= e((\prod_{j=1}^{J}(H_2(j))^{\tau_j}) g_2^{d}, T_o P_k^{h_o} P_o^{\alpha_o}) \end{aligned}$$

7 Security Proof

First of all, the security model used is based on work [14].

7.1 Security Mode

This model has two types of opponents, namely Type I opponent $\mathcal{A}_1$ and Type II opponent $\mathcal{A}_2$. $\mathcal{A}_1$ can replace any user public key but cannot obtain the master key. $\mathcal{A}_2$ can obtain the master key but cannot replace any user's public key. The game between $\mathcal{A} \in \{\mathcal{A}_1, \mathcal{A}_2\}$ and challenger $\mathcal{C}$ formally defines the security of the proposed scheme. After the game starts, $\mathcal{C}$ executes the Setup algorithm to generate system parameters and transmits them to $\mathcal{A}$. Then, $\mathcal{A}$ performs the following query on $\mathcal{C}$.

$\mathcal{C}$ runs related operations to generate private key and public key $\{P_u, T_u\}$. $\mathcal{C}$ then sends it to $\mathcal{A}$.

PartialPrivateKey(id_u): $\mathcal{C}$ sends the partial private key $\{s_u, T_u\}$ to $\mathcal{A}$.

PublicKeyReplacement(id_u, $\{P_u, T_u\}'$): $\mathcal{C}$ replaces $\{P_u, T_u\}$ with $\{P_u, T_u\}'$.

SecretValue(id_u): $\mathcal{C}$ sends x_u to $\mathcal{A}$, where x_u is the secret value.

TagGen(id_u, d_j): $\mathcal{C}$ gets tags through TagGen algorithm and gives them to $\mathcal{A}$.

$\mathcal{A}$ falsifies the proof $\{C^*, d^*\}$ relative to J^*. The conditions for $\mathcal{A}$ to win are as follows.

(1) $ProofVerif(\alpha_o{}^*, P_o{}^*, T_o{}^*, P_k{}^*, h_o{}^*) \rightarrow True$
(2) If $\mathcal{A}$ is an I-type opponent, there will be no query about PartialPrivate Key($id_o{}^*$). If $\mathcal{A}$ is a Type II opponent, there will be no query about SecretValue($id_o{}^*$) in the game.
(3) There is no query related to TagGen($id_o{}^*$, $d_{i_j^*}{}^*$) in the game, where $i_j^* \in J^*$

Define the probability of $\mathcal{A}$ winning in the game as $Succ_{CLPA}^{cma}(\mathcal{A})$. For any $\mathcal{A} \in \{\mathcal{A}_1, \mathcal{A}_2\}$ if $Succ_{CLPA}^{cma}(\mathcal{A})$ can be ignored, the proposed scheme is secure against two types of opponents.

7.2 Security Analysis

This part proves through several lemmas and theorems that the scheme in this article can prevent the two kinds of opponents proposed. In the security proof, H_1 and H_2 are regarded as random predictions [15].

Lemma 1. The premise that the scheme in this paper is secure for Type I opponents is that the CDH problem is difficult to solve.

Proof. Define $\mathcal{A}_1$ as a Type I opponent. In the game, its probability ϵ cannot be ignored. The concept of the security proof in this paper is that if $\mathcal{A}_1$ exists, then an attacker can be generated through $\mathcal{A}_1$, and the attacker can crack the CDH problem. Given $(P, Y = P^a, \Theta = P^b)$, $\mathcal{C}$ sets $Y \rightarrow P_k$, chooses id_o as the challenge identity, and then transmits system parameters to $\mathcal{A}_1$. Then, $\mathcal{A}_1$ can execute the following query on $\mathcal{C}$.

$H_1(id_u, T_u, P_u)$: $\mathcal{C}$ keeps an initially empty list L_{H_1}. $\mathcal{C}$ judges whether $\{id_u, T_u, P_u, h_u\}$ is in L_{H_1}. If it exists, $\mathcal{C}$ transmits h_u to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly selects $h_u \in Z_q^*$ and stores $\{id_u, T_u, P_u, h_u\}$ in L_{H_1}. $\mathcal{C}$ transmits h_u to $\mathcal{A}_1$.

$H_2(i, c_i, z_i, Z_i)$: $\mathcal{C}$ maintains an initially empty list L_{H_2}. $\mathcal{C}$ judges whether $\{i, c_i, z_i, Z_i\}$ is in L_{H_2}. If it exists, $\mathcal{C}$ transmits $H_2(i) = Z_i$ to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly selects $b_i \in \{0, 1\}$ and $Pr[b_i = 0] = \lambda$, where $\lambda \in (0, 1)$. $\mathcal{C}$ chooses $z_i \in Z_q^*$. If $b_i = 0$, then $\mathcal{C}$ calculates $Z_i = z_i \cdot P$. Otherwise, $\mathcal{C}$ calculates $Z_i = z_i \cdot \Theta$. $\mathcal{C}$ stores $\{i, c_i, z_i, Z_i\}$ in the list L_{H_2}, and gives $H_2(i) = Z_i$ to $\mathcal{A}_1$.

$H_3(id_o, P_o, T_o, P_k)$: $\mathcal{C}$ maintains an initially empty list L_{H_3}. $\mathcal{C}$ checks whether $\{id_o, P_o, T_o, P_k, \alpha\}$ is in L_{H_3}. If it exists, $\mathcal{C}$ sends α to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly selects a random value $\alpha \in Z_q^*$ and stores $\{id_o, P_o, T_o, P_k, \alpha\}$. $\mathcal{C}$ sends α to $\mathcal{A}_1$.

CreateUser(id_u): $\mathcal{C}$ keeps an initially empty list L_k. $\mathcal{C}$ judges whether $\{id_u, s_u, x_u, P_u, T_u\}$ is in L_k. If it is, then $\mathcal{C}$ gives $\{P_u, T_u\}$ to $\mathcal{A}_1$. If it is not and $id_o = id_u$, then $\mathcal{C}$ generates some random numbers $t_u, x_u \in Z_q^*$, sets $\perp \rightarrow s_u$, calculates $T_u = {g_1}^{t_u}$, $P_u = {g_1}^{x_u}$ and puts $\{id_u, s_u, x_u, P_u, T_u\}$ in L_k. If $\{id_u, s_u, x_u, P_u, T_u\}$ is not in L_k and $id_o! = id_u$, $\mathcal{C}$ generates some random numbers s_u, w_u, x_u, calculates $P_u = {g_1}^{x_u}$, $T_u = {g_1}^{s_u}/{g_1}^{w_u}$ and puts $\{id_u, s_u, x_u, P_u, T_u\}$ and $\{id_u, P_u, T_u, w_u\}$ in L_k and L_{H_1}. Then, $\mathcal{C}$ sends $\{P_u, T_u\}$ to $\mathcal{A}_1$.

PartialPrivateKey(id_u): $\mathcal{C}$ Check whether the formula $id_u = id_o$. If so, $\mathcal{C}$ interrupts the process. If not, $\mathcal{C}$ finds $\{id_u, s_u, x_u, P_u, T_u\}$ in L_k, and then sends $\{s_u, T_u\}$ to $\mathcal{A}_1$.

PublicKeyReplacement (id_u, $\{P_u, T_u\}'$): $\mathcal{C}$ Look for $\{id_u, s_u, x_u, P_u, T_u\}$ in L_k, and replace $\{P_u, T_u\}$ with $\{P_u, T_u\}'$.

SecretValue(id_u): $\mathcal{C}$ looks for $\{id_u, s_u, x_u, P_u, T_u\}$ in L_k, and sends x_u to $\mathcal{A}_1$.

TagGen(id_u, d_j): After $\mathcal{C}$ receives the query of id_u and data d_i, $\mathcal{C}$ judges whether id_u and id_o are the same.. If they are equal, $\mathcal{C}$ finds the list L_k of $\{id_u, s_u, x_u, P_u, T_u\}$ and gets τ_i. Otherwise, $\mathcal{C}$ aborts.

Finally, $\mathcal{A}_1$ uses id_o^* to forge proof that $\{C^*, d^*\}$, $\{C^*, d^*\}$ and J^* are related. If $id_o! = id_o^*$ or ${d_{i_j^*}}^* = 0 (i_j^* \in J^*)$, then $\mathcal{C}$ ends. Otherwise, $\mathcal{C}$ searches for $\{id_o, s_o, x_o, P_o, T_o\}$ and $\{i_j^*, c_{i_j^*}, z_{i_j^*}, Z_{i_j^*}\}$ in L_k and L_{H_2}, where $i_j^* \in J^*$. $\mathcal{C}$ gets

$$e(C^*, g_1) = e((\prod_{i_j^*=1}^{J^*} (H_2(i_j^*))^{\tau_{i_j^*}}) g_2^{d^*}, T_o P_k^{h_o} P_o^{\alpha_o})$$

According to the bifurcation lemma [15], $\{C^{*'}, d^*\}$ can be obtained by choosing different H_1 and H_3. Where $\{C^{*'}, d^*\}$ satisfies

$$e(C^{*'}, g_1) = e((\prod_{i_j^*=1}^{J^*} (H_2(i_j^*))^{\tau_{i_j^*}}) g_2^{d^*}, T_o P_k^{h_o'} P_o^{\alpha_o'})$$

From the above two equations. $\mathcal{C}$ gets

$$e((C^* - C^{*'}), g_1) = e(P_k^{h_o - h_o'} P_o^{\alpha_o - \alpha_o'} (\prod_{i_j^*=1}^{J^*} (H_2(i_j^*))^{\tau_{i_j^*}}) g_2^{d^* t_o}, g_1)$$

Then, $\mathcal{C}$ outputs Φ^{-1} and ω as answers to the CDH problem, where $\Phi^{-1} \in Z_q^*$ and $\Phi^{-1}.\Phi \equiv 1 \ mod \ q$. $\Phi = P_k^{h_o - h_o'} P_o^{\alpha_o - \alpha_o'} (\prod_{i_j^*=1}^{J^*} (H_2(i_j^*))^{\tau_{i_j^*}})$ and $\omega = (C^* - C^{*'})$.

We analyze the possibility of CDH problem being solved by $\mathcal{C}$. The following events need to be considered.

E_1: $\mathcal{C}$ is not interrupted.
E_2: $\mathcal{A}_1$ outputs the legal proof of J^* with ${d_{i_j^*}}^* = 1 (i_j^* \in J^*)$.
E_3: $id_o^* = id_o$ is established.

It is not easy to obtain $Pr[E_1] \geq (1 - 1/q_h)^{q_{tag}}$, $Pr[E_2|E_1] \geq (1 - \lambda)^{q_{tag}}$ and $Pr[E_3|E_2 \wedge E_1] \geq 1/q_h$, where q_h and q_tag represent the number of H_1 queries and TagGen queries. Let E_{suc} prove that the CDH problem can be cracked by $\mathcal{C}$.

Then we have $E_{suc} = E_1 \wedge E_2 \wedge E_3$ and

$$\begin{aligned} Pr[E_1 \wedge E_2 \wedge E_3] &= Pr[E_1] Pr[E_2|E_1] Pr[E_3|E_1 \wedge E_2] \\ &\geq (1 - 1/q_h)(1 - \lambda)^{q_{tag}} (1 - 1/q_h)^{q_{cert}} \epsilon \end{aligned}$$

Since ϵ is not negligible, there is no attacker like $\mathcal{C}$ who can crack the CDH problem. Therefore, if the CDH problem is difficult to solve, the proposed scheme is secure for Type I opponents.

Lemma 2. The premise that the scheme in this paper is secure for Type II opponents is that the CDH problem is difficult to solve.

Proof. Assume that $\mathcal{A}_2$ is a Type II opponent, and its probability ϵ in the game cannot be ignored. If there is such an adversary, an attacker can be defined, and the attacker can crack the CDH problem. Given $(P, Y = P^a, \Theta = P^b)$, $\mathcal{C}$ chooses a random value $s \in Z_q^*$, calculates $P_k = {g_1}^s$, and then selects id_o as the challenge identity. $\mathcal{C}$ gives the key s and PP to $\mathcal{A}_2$. $\mathcal{C}$ solves $\mathcal{A}_2$'s H_1 queries, H_2 queries, H_3 queries and TagGen queries. The process is the same as the proof of Lemma 1. The other query steps of $\mathcal{A}_2$ are as follows.

CreateUser(id_u): $\mathcal{C}$ maintains an initially empty list L_k. $\mathcal{C}$ judges whether $\{id_u, s_u, x_u, P_u, T_u\}$ is in L_k. If it is, then $\mathcal{C}$ gives $\{P_u, T_u\}$ to $\mathcal{A}_2$. If it is not and $id_o = id_u$, then $\mathcal{C}$ generates some random numbers $t_u \in Z_q^*$, sets $\perp \rightarrow x_u$ and $Y \rightarrow P_u$, calculates $T_u = {g_1}^{t_u}$, $h_u = H_1(id_u, T_u, P_u)$, $s_u = t_o + sh_u \ mod \ q$ and puts $\{id_u, s_u, x_u, P_u, T_u\}$ in L_k. If $\{id_u, s_u, x_u, P_u, T_u\}$ is not in L_k and $id_o! = id_u$, $\mathcal{C}$ generates some random numbers s_u, w_u, x_u, calculates $P_u = {g_1}^{x_u}$, $T_u = {g_1}^{t_u}$, $h_u = H_1(id_u, T_u, P_u)$, $s_u = t_o + sh_u \ mod \ q$ and puts $\{id_u, s_u, x_u, P_u, T_u\}$ and $\{id_u, P_u, T_u, w_u\}$ in L_k and L_{H_1}. Then, $\mathcal{C}$ transmits $\{P_u, T_u\}$ to $\mathcal{A}_2$.

PartialPrivateKey(id_u): $\mathcal{C}$ looks for $\{id_u, s_u, x_u, P_u, T_u\}$ in L_k, and then sends $\{s_u, T_u\}$ to $\mathcal{A}_2$.

SecretValue(id_u): $\mathcal{C}$ Check whether the formula $id_u = id_o$. If so, $\mathcal{C}$ interrupts the game. If not, $\mathcal{C}$ finds $\{id_u, s_u, x_u, P_u, T_u\}$ in L_k, and sends x_u to $\mathcal{A}_2$.

Finally, $\mathcal{A}_2$ uses id_o^* to forge proof that $\{C^*, d^*\}$, $\{C^*, d^*\}$ and J^* are related. If $id_o! = id_o^*$, then $\mathcal{C}$ is interrupted. Otherwise, $\mathcal{C}$ finds $\{id_o, s_o, x_o, P_o, T_o\}$ and $\{i_j^*, c_{i_j^*}, z_{i_j^*}, Z_{i_j^*}\}$ in L_k and L_{H_2}, where $i_j^* \in J^*$. $\mathcal{C}$ gets

$$e(C^*, g_1) = e(P_k^{h_o} P_o^{\alpha_o} (\prod_{i_j^*=1}^{J^*} (H_2(i_j^*))^{\tau_{i_j^*}}) Y^{d^*}, g_2)$$

Then, $\mathcal{C}$ outputs Φ^{-1} and ω as answers to the CDH problem, where $\Phi^{-1} \in Z_q^*$ and $\Phi^{-1}.\Phi \equiv 1 \ mod \ q$. $\Phi = P_k^{h_o} P_o^{\alpha_o} (\prod_{i_j^*=1}^{J^*} (H_2(i_j^*))^{\tau_{i_j^*}})$ and $\omega = (C^*)$.

We analyze the possibility of CDH problem being solved by $\mathcal{C}$. The following events need to be considered.

E_1: $\mathcal{C}$ is not interrupted.
E_2: $\mathcal{A}_1$ outputs the legal proof of J^* with ${d_{i_j^*}}^* = 1 (i_j^* \in J^*)$.
E_3: $id_o^* = id_o$ is established.

It is not easy to obtain $Pr[E_1] \geq (1 - 1/q_h)^{q_{tag}}$, $Pr[E_2|E_1] \geq (1 - \lambda)^{q_{tag}}$ and $Pr[E_3|E_2 \wedge E_1] \geq 1/q_h$, where q_h and q_tag represent the number of H_1 queries and TagGen queries. Let E_{suc} prove that the CDH problem can be cracked by $\mathcal{C}$.

Then we have $E_{suc} = E_1 \wedge E_2 \wedge E_3$ and

$$\begin{aligned} Pr[E_1 \wedge E_2 \wedge E_3] &= Pr[E_1] Pr[E_2|E_1] Pr[E_3|E_1 \wedge E_2] \\ &\geq (1 - 1/q_h)(1 - \lambda)^{q_{tag}} (1 - 1/q_h)^{q_{cert}} \epsilon \end{aligned}$$

Since ϵ is not negligible, there is no attacker like $\mathcal{C}$ who can crack the CDH problem. Therefore, if the CDH problem is difficult to solve, the proposed scheme is secure for Type II opponents.

Theorem 1 is derived from Lemma 1 and Lemma 2. Theorem 1 the premise that the scheme in this paper is secure for both Type I and Type II opponents is that the CDH problem is difficult to solve.

8 Performance

We analyzed the performance of the scheme, related experimental environment and data reference [16]. In order to better demonstrate the advantages of the scheme proposed in this paper, we compare the scheme of this paper, scheme [14] and scheme [17] (For the sake of comparison, we assume that $s = 2$ *and* $m = 3$ in the scheme [17].). Comparison method reference scheme [14]. The symbols used in this section are as follows Table 1.

Table 1. Run time of operations (millisecond)

Symbols	Definition	Time
T_H	The time spent in a hash operation mapped to the point	5.493
T_p	The time spent in a bilinear pairing operation	5.427
T_{mul}	The time it takes to perform a point multiplication operation	2.165
T_E	The time spent in an exponentiation operation	0.339
T_{add}	The time it takes to perform a point addition operation	0.013
T_h	The time spent in a regular hash operation	0.007
T_m	The time spent in a modular multiplication operation	0.001

8.1 Computation Cost

In order to compare the process more clearly. For the scheme, we only compare the three parts TagGen, ProofGen and ProofVerify. These three parts are the core of the scheme.

Regarding our scheme, part TagGen of it contains $2n$ exponentiation operations (EO), n regular hash operations (RHO) and n modular multiplication operations (MMO). The computation time of part TagGen is $2nT_E + nT_h + nT_m = 0.686n$ (ms). Part ProofGen contains J EO and $(J-1)$ MMO. The computation time of part ProofGen is $JT_E + (J-1)T_m = 0.340J - 0.001$ (ms). Part ProofVerify contains $(J+3)$ EO, J RHO and $(J+2)$ MMO. The computation time of part ProofVerify is $(J+3)T_E + (J)T_h + (J+2)T_m = 0.347J + 1.019$ (ms). The total time of our scheme is $(2n+2J+3)T_E + (n+J)T_h + (n+2J+1)T_m = 0.686n + 0.687J + 1.018$ (ms). The comparison results of other schemes are similar to this scheme, and the results are shown in Table 2.

Table 2. The comparison of communication cost

Schemes	Costs	Total
Work [14]	TagGen: $nT_H + 2nT_{mul} + nT_{add}$ ProofGen: $(J+1)T_{mul} + (J-1)T_{add}$ ProofVerify: $(J+1)T_H + (J+3)T_{mul} + (J+2)T_{add}$	$9.836n + 9.849J + 14.166$ (ms)
Work [17]	TagGen: $3nT_E + 4nT_m + nT_h$ ProofGen: $JT_E + (J-1)T_m$ ProofVerify: $(J+3)T_E + JT_h + (J+1)T_m$	$1.028n + 0.687J + 1.017$ (ms)
Our scheme	TagGen: $2nT_E + nT_h + nT_m$ ProofGen: $JT_E + (J-1)T_m$ ProofVerify:$(J+3)T_E + (J)T_h + (J+2)T_m$	$0.686n + 0.687J + 1.018$ (ms)

Tip: n represents the number of all data blocks.
s represents the number of data blocks that need to be verified ($s \leq n$).

It can be seen from Table 2 that s is less than or equal to n, so if you replace s with n for the data in the table and combined with practical applications, you can find $(1.373n+1.018)$ (ms) $\leq (1.715n+1.017)$ (ms) $\leq (19.685n+14.166)$ (ms). In summary, the advantages of the proposed scheme are obvious.

8.2 Communication Cost

The environmental reference of communication overhead [14], so the length of the elements in G_1 and Z_q^* are 1024 and 160 bits, respectively. Assume that n is 32 bits. n represents the amount of the data sub-block.

For scheme [14], in the communication process, what needs to be sent is the challenge $\{j, w_j\}_{j \in J}$ and the proof $\{R, S, \omega\}$, where $j \in \{1, 2, \cdots, n\}$, $\omega \in Z_q^*$, $R, S \in G_1$ and $d \in Z_q^*$. The result is $(32+160)J + 1024 + 1024 + 160 = 192J + 2208\ bits$.

For scheme [17], in the communication process, what needs to be sent is the challenge $\{auth, \{AID\}_{CEMR_{PK}}, (j, v_j)\}$ and the proof $\{\sigma, u\}$, where $j \in \{1, 2, \cdots, n\}$, $\{AID\}_{CEMR_{PK}}, auth, v_j \in Z_q^*$, $\sigma \in G_1$ and $u \in Z_q^*$. Therefore, the communication cost is $(32+160)J + 160 + 160 + 1024 + 160 = 192J + 1504\ bits$.

For our scheme, in the communication process, what needs to be sent is the challenge $\{j, \tau_j\}_{j \in J}$ and the proof $\{C, d\}$, where $j \in \{1, 2, \cdots, n\}$, $\tau_j \in Z_q^*$, $C \in G_1$ and $d \in Z_q^*$. Therefore, the communication cost is $(32+160)J + 1024 + 160 = 192J + 1184\ bits$.

In summary, formula $192J + 1184\ bits \leq 192J + 1504\ bits \leq 192J + 2208\ bits$ holds, so the communication consumption in this paper is lower in comparison. The communication efficiency of the scheme in this paper has advantages.

9 Conclusion

This paper proposes an efficient certificateless cloud data integrity detection scheme for ecological data, which solves the problems of cloud storage service providers' untrustworthiness and storage data loss faced by ecological data cloud storage. This scheme uses a certificate-free form to avoid certificate management and key escrow problems. The security analysis results show that this scheme is secure. The analysis of computational consumption and communication consumption shows that this scheme has obvious efficiency advantages. Future research considers to further improve efficiency and enhance functions on the basis of this scheme.

References

1. Ateniese, G.: Provable data possession at untrusted stores. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, pp. 598–609 (2007)
2. Ateniese, G., Di Pietro, R., Mancini, L.V., Tsudik, G.: Scalable and efficient provable data possession. In: Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks, pp. 1–10 (2008)
3. Ateniese, G., et al.: Remote data checking using provable data possession. ACM Trans. Inf. Syst. Secur. (TISSEC) **14**(1), 1–34 (2011)
4. Chris Erway, C., Küpçü, A., Papamanthou, C., Tamassia, R.: Dynamic provable data possession. ACM Trans. Inf. Syst. Secur. (TISSEC) **17**(4), 1–29 (2015)
5. Sebé, F., Domingo-Ferrer, J., Martinez-Balleste, A., Deswarte, Y., Quisquater, J.-J.: Efficient remote data possession checking in critical information infrastructures. IEEE Trans. Knowl. Data Eng. **20**(8), 1034–1038 (2008)
6. Wang, Q., Wang, C., Ren, K., Lou, W., Li, J.: Enabling public auditability and data dynamics for storage security in cloud computing. IEEE Trans. Parallel Distrib. Syst. **22**(5), 847–859 (2010)
7. Zhu, Y., Hongxin, H., Ahn, G.-J., Mengyang, Yu.: Cooperative provable data possession for integrity verification in multicloud storage. IEEE Trans. Parallel Distrib. Syst. **23**(12), 2231–2244 (2012)
8. Shamir, A.: Identity-Based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_5
9. Wang, H., Qianhong, W., Qin, B., Domingo-Ferrer, J.: Identity-based remote data possession checking in public clouds. IET Inf. Secur. **8**(2), 114–121 (2013)
10. Tan, S., Jia, Y.: NaEPASC: a novel and efficient public auditing scheme for cloud data. J. Zhejiang University SCIENCE C **15**(9), 794–804 (2014)
11. Wang, H.: Identity-based distributed provable data possession in multicloud storage. IEEE Trans. Serv. Comput. **8**(2), 328–340 (2014)
12. Al-Riyami, S.S., Paterson, K.G.: Certificateless public key cryptography. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 452–473. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-40061-5_29
13. Wang, B., Li, B., Li, H., Li, F.: Certificateless public auditing for data integrity in the cloud. In: 2013 IEEE Conference on Communications and Network Security (CNS), pp. 136–144. IEEE (2013)

14. He, D., Kumar, N., Wang, H., Wang, L., Choo, K.-K.R.: Privacy-preserving certificateless provable data possession scheme for big data storage on cloud. Appl. Math. Comput. **314**, 31–43 (2017)
15. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. J. Cryptol. **13**(3), 361–396 (2000)
16. Wu, L., Wang, J., Choo, K.-K.R., He, D.: Secure key agreement and key protection for mobile device user authentication. IEEE Trans. Inf. Forensics Secur. **14**(2), 319–330 (2018)
17. Zhou, L., Fu, A., Feng, J., Zhou, C.: An efficient and secure data integrity auditing scheme with traceability for cloud-based EMR. In: ICC 2020–2020 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2020)