A Certificateless Verifiable Bilinear Pair Free Conjunctive Keyword Search Encryption Scheme for IoMT

Weifeng Long *, Jiwen Zeng, Yaying Wu, Yan Gao, Hui Zhang

Posted Date: 31 January 2024

doi: 10.20944/preprints202401.2177.v1

Keywords: Internet of medical things (IoMT); certificateless encyption with keyword search; standard model; keyword guessing attack; file injection attack, security; privacy

Article

A Certificateless Verifiable Bilinear Pair Free Conjunctive Keyword Search Encryption Scheme for IoMT

Weifeng Long [1,2,*], Jiewen Zeng [1], Yaying Wu[2], Yan Gao[2] and Hui Zhang [2]

[1] School of Mathematical Sciences, Xiamen University, Xiamen 361005, China; longweifeng2009@126.com(W.F.L.);jwzeng@xmu.edu.cn(J.W.Z.)

[2] School of Mathematical Sciences, Guizhou Normal University, Gui'an New District, Guiyang 550025, China; 2430476644@qq.com(Y.Y.W.);727120985@qq.com(Y.G.);1467228300@qq.com(H.Z.)

* Correspondence: longweifeng2009@126.com(W.F.L.)

Abstract: The Internet of Medical Things (IoMT) has powerful cloud computing ability and efficient data collection ability, which can improve the accuracy and convenience of medical work. As most communications are over open networks, data should encrypted before uploading to ensure the confidentiality of sensitive user data. Searching for encrypted data has become a challenging problem. Public key Encryption with Keyword Search (PEKS) supports the search of encrypted data and provides data privacy protection. PEKS has become a trendy technology, but PEKS still has the following problems: 1. The cloud server, as a semi-honest but curious third party, is likely to return some wrong search results to save computing and bandwidth resources. 2. The single keyword search inevitably produces irrelevant results, wasting computing and bandwidth resources. 3. Most PEKS schemes use bilinear pairing, so the computational efficiency is relatively low. 4. PEKS schemes based on Public Key Infrastructure (PKI) or identity-based cryptography inevitably face problems with certificate management or key escrow. 5. Most PEKS schemes face serious security threats such as Keyword Guessing Attacks (KGA) and File Injection Attacks (FIA). To solve the above problems, we propose a new certificateless verifiable bilinear pair-free conjunctive keyword search encryption scheme (CLVPFCKS) for IoMT using multi-signature.

Keywords: Internet of medical things (IoMT); certificateless encyption with keyword search; standard model; keyword guessing attack; file injection attack; security; privacy

0. Introduction

Internet of Things is a network that connects any item to the Internet and uses information sensing devices such as radio frequency identification, infrared sensors, global positioning systems, and laser scanners to exchange and communicate information according to agreed protocols, achieving intelligent identification, positioning, tracking, monitoring, and management [1,2]. The Internet of Medical Things (IoMT) conveniently connects communication technology, medical staff, patients, various medical devices, and intelligent facilities, thus completing brilliant medical care and innovative item management. The Internet of Medical Things system can achieve real-time feedback on the health status of patients, improve medical response speed, provide 24-hour medical care, significantly reduce the work pressure of medical personnel, improve the accuracy and convenience of medical work, improve clinical medical quality, control costs, reduce efficiency, and save lives.

Electronic Medical Record (EMR) is crucial in IoMT systems. It electronically manages personal health status and healthcare information involving patient information collection, storage, transmission, processing, and utilization.The volume of medical data will inevitably surge as medical data is collected through various channels digitally. How to effectively store and manage the increasing number of electronic medical records has become an unprecedented challenge. In response to this problem, cloud computing can be a supplementary tool. With cloud computing, hospitals, and medical organizations outsource EMR to cloud servers for storage to save local data management and system maintenance

 doi:10.20944/preprints202401.2177.v1

costs while also achieving resource sharing with data recipients. Figure 1 shows a typical architecture of IoMT for medical IoT.

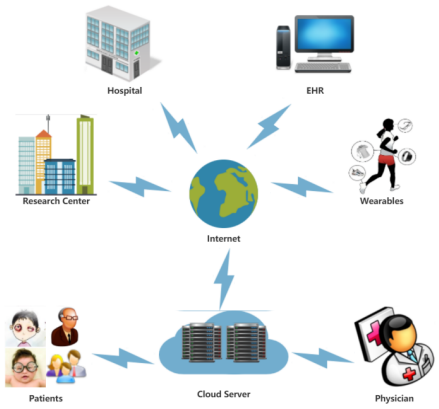

Figure 1. Architecture of IoMT.

EHRs involve patients' privacy, and are often encrypted to protect patients' privacy and security. However, this causes great inconvenience for users to search for EMRs containing specific keywords. A simple solution to this problem is for users to download all ciphertext data fully, decrypt it, and further search locally. However, this leads to high computational and communication costs, making them impractical. To solve this problem, researchers have proposed searchable encryption (SE) [7] technology. It is an encryption primitive that enables users to search for keywords on ciphertext in a privacy-protected manner. In 2004, Boneh et al. [8] proposed the concept of public key encryption with keyword search (PEKS) to solve the problem of searching data encrypted using public key cryptosystems.

1. Related work

The initial PEKS scheme had many areas for improvement regarding efficiency and safety. Firstly, review the efforts made by numerous researchers in improving the efficiency of PEKS. Baek [9] pointed out the inefficiency of the PEKS [8] scheme due to the use of secure channels. To eliminate the requirement for security channels, Baek et al. proposed the concept of secure channel-free PEKS (SCF-PEKS), also known as PEKS, with designated servers/testers (dPEKS) [10]. The efficiency of SCF-PEKS is improved. However, the PEKS (SCF-PEKS) schemes are built based on PKI or identity-based cryptosystem and encounter certificate management issues and key escrow issues in system deployment. To avoid the problems associated with certificate and key escrow, Peng et al. [11] proposed a certificateless keyword searchable encryption scheme (CLKS) without secure channels. Subsequently, much literature has studied certificateless searchable encryption schemes with keywords, such as [12–15]. Most PEKS schemes focus on searching for a single keyword, but single keyword search inevitably produces many irrelevant results, leading to bandwidth and computational resource waste. Golle et al. [16] proposed the first conjunctive keyword searchable encryption scheme to avoid resource waste. After this, many searchable encryption schemes, such as [17–19], support conjunctive keyword search.

Because the computational complexity of bilinear pairs is much higher than that of scalar multiplication on elliptic curve groups, designing searchable encryption schemes without bilinear pair operations can significantly improve the scheme's efficiency. However, currently, there are not many searchable encryption schemes without bilinear [11,15,20–27], and they are not perfect enough. In 2019, Lu et al. [28] proposed a pairing-free certificateless searchable public key encryption scheme, proving that this scheme achieves indistinguishability of keyword ciphertext against adaptive keyword selection attacks under the complexity assumption of computing Diffie-Hellman problems in a random

 doi:10.20944/preprints202401.2177.v1

oracle model. However, Ma et al. [30] found that the scheme proposed in [28] is not secure against user simulation attacks and offered a new cloud-based IIoT pairing-free dual server CLPEKS scheme. Recently, [25], [26], and [27] respectively proposed secure and effective pairing-free CLPEKS schemes, but they are all single keyword searches.

Cryptography researchers place a high value on security. Next, let's look at what researchers have done to improve the security of PEKS. The existing SCF-PEKS scheme is susceptible to keyword guessing attacks (KGA) [28] and file injection attacks (FIA) [29]. To avoid keyword guessing attacks and file injection attacks, Hwang et al. [30] embedded random keys in the keyword ciphertext of the PKSE scheme and claim that the scheme can resist the above attacks. However Yang [40] proved Hwang et al.'s SCF-PEKS scheme [30] is insecure under external online keyword attacks. The main reason for being vulnerable to external online keyword attacks is that opponents can generate legitimate ciphertext for the keyword. In addition to being attacked by external attackers, the PEKS scheme is also vulnerable to attacks from internal attackers (usually referring to malicious cloud servers). Jeong et al. [41] demonstrated that the PEKS framework is susceptible to offline keyword attacks (i.e., internal offline keyword attacks) by malicious data storage servers. Later, Shao and Yang [37] proposed a universal attack demonstrating the inability to construct an SCF-PEKS scheme to defend against malicious internal servers. They pointed out that because malicious servers can run keyword encryption and testing algorithms, SCF-PEKS is inherently vulnerable to offline keyword-guessing attacks from malicious insider servers.

In addition, a cloud server is a semi-honest but curious third party that may perform only a small portion of search operations and return a small amount of incorrect search results to save its computing and bandwidth resources. Therefore, the PEKS scheme should be equipped with a verification mechanism to ensure the correctness of search results without decrypting the ciphertext. Reference [31] proposed the first symmetric and verifiable encryption scheme for keyword search. Subsequently, many literature studies have investigated verifiable keyword searchable encryption schemes [32–35]. In reference [34], a verifiable conjunctive keyword search scheme (VCKSM) over mobile e-health cloud in shared multi-owner settings was proposed, and it was secure against keyword guessing attacks under the standard model; In reference [35], an identity-based certificateless verifiable conjunctive keyword searchable encryption scheme (VMKS) was constructed, which avoids certificate management or key escrow restrictions and achieves indistinguishability of ciphertext and unforgeability of signatures.

1.1. Our contribution

We construct a certificateless verifiable bilinear pair-free conjunctive keyword search encryption scheme (CLVPFCKS), and it has been proven under the standard model that it can resist keyword guessing attacks (KGA), file injection attacks (FIA), and choose keyword attacks (CKA). Specifically, the main contributions are as follows:

1. Search results validation: The scheme allows the signature to be attached to each file to verify the accuracy of the search results.
2. No bilinear pairing: The calculation of bilinear pairing needs more time, and the scheme's efficiency will significantly improve without bilinear pairing.
3. We prove that the new scheme can resist offline keyword guessing attacks, file injection attacks, and choose keyword attacks (CKA) under the standard model.

1.2. Organization

The following is the framework for the rest of this article. We summarize some related work in section 1. We discuss preparatory knowledge in section 2. We give the scheme's construction and security analysis (including the security model and proof) in section 3. We show the details of the CLVPFCKS scheme in section 4. We discuss the security of the CLVPFCKS scheme in section 5. We analyze the effectiveness of CLVPFCKS in section 6. Finally, we summarize this paper in section 7.

 doi:10.20944/preprints202401.2177.v1

2. Preliminaries

Let $q > 3$ be a large prime, F_q be a prime field, and the elliptic curve E over the field F_q must satisfy the equation

$$y^2 mod q = (x^3 + ax + b) mod q,$$

Which $a, b, x, y \in F_q$ and $(4a^3 + 27b^2) mod q \neq 0$. All points on E and the infinite point O form a cyclic group. ECC (elliptic curve cryptosystem) has the following difficulties:

1. Elliptic curve discrete logarithm problem (ECDLP): given $P, Q \in G_q$, where P is the generator of the group, and Q is the element in G_q. It is difficult to calculate the integer k such that $Q = kP$, where $k \in Z_{q^*}$.
2. Elliptic curve Computational Diffie-Hellman problem (ECDCHP): for any given a, b in Z_{q^*}, it is difficult to calculate abP, where $(P, aP, bP) \in G_q$.
3. Elliptic curve Decisional Diffie-Hellman problem (ECDDDH): Given $aP, bP \in G$ where a, b unknown. The DDH (Decisional Diffie-Hellman) problem is to decide whether X equals abP or a random element in G.

3. The System Model and Attack Model of CLVPFCKS

3.1. System model

There exist five entities in the proposed CLVPFCKS scheme for KGC: multiple data owners (patient and his doctors), Cloud Service Provider (CSP), data user (other authorized doctors or healthcare center), and Private Audit Server (PAS), as shown in Figure 2.

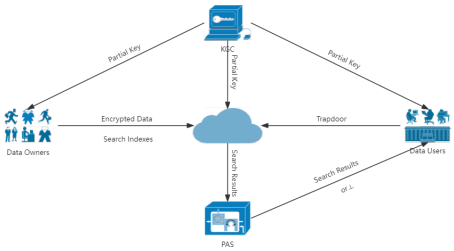

Figure 2. System model of CLVPFCKS

KGC: It is a trusted third party, and it is responsible for generating system parameters and producing data owners, Cloud Service Providers (CSP), and data user's partial private keys.

Multiple data owners: In reality, the cloud search system supports more shared scenarios of multiple data owners, especially in the electronic medical system. For example, one patient and several hospital staff (i.e., surgeons, physicians, etc.) share an electronic medical record jointly, and each staff member is responsible for the content of a specific part (i.e., block). If each block is an independent record, it will inevitably encounter multiple indexes, thus greatly expanding the computational and spatial overhead. Otherwise, the many data owners must have the same random elements, which is not feasible in practice. On the contrary, the scheme designed for shared multi-owner settings only requires a single index of the entire record, saving considerable time and space.

For each EHR jointly owned by the patient and its doctor, each data owner generates a signature on this record. All data users entrust user O_j to establish an index based on the keywords of each EHR, and then the index and signature are sent to CSP. Note that the detailed algorithm for encrypting each EHR is beyond the discussion, and any public key encryption algorithm can be applied.

CSP: It is a semi-trusted entity. It has professional knowledge and can provide data storage and retrieval services for authorized cloud clients. After receiving a trapdoor from the data user, it returns the corresponding documents. But CSP is a semi-honest but curious third party, which many only

 doi:10.20944/preprints202401.2177.v1

perform a small part of the search operation and provide a small portion of the wrong search results to save its computing and bandwidth resources.

Data users: Data users obtain a partial private key from KGC generate the trapdoor of a keyword they wish to search, and then send it to the cloud server.

PAS: This fully trusted server is responsible for verifying the correctness of search results.

The CSP is assumed to be semi-honest but curious. It performs a fraction of search operations honestly but is interested in spying out the sensitive information valuable for the CSP. Furthermore, it may return false search results to save computing resources. On the other hand, the PAS is fully trusted and can guarantee the correctness of search results. Besides, the authorized data user can issue search queries without leaking valuable information to CSP.

3.2. Design Goals

We plan to achieve the following goals:

Multiple keyword search :The proposed scheme allows specific data users to send multiple keyword searches without increasing the trapdoor size and ciphertext search size, which improves the user search experience.

Search results validation: The scheme verifies the accuracy of search results by attaching a signature to each file.

Certificateless: The scheme is based on the certificateless cryptosystem to eliminate the limitations of certificate management and key escrow in existing searchable encryption schemes.

Pairing free: It takes more time to compute bilinear pairing. The efficiency will improve if it does not use bilinear pairing operations.

Free secure channel: It is necessary to eliminate the security channel in practice to reduce the costs of the system.

Supporting safe goals: We have demonstrated under the standard model that our scheme can resist keyword guessing attacks (KGA), file injection attacks (FIA), and chosen keyword attacks (CKA).

3.3. Solution Framework

Set integer k as security level, (F, W) as EHR file and keyword set. Definition1 (Our proposed CLVPFCKS scheme) Our scheme is a tuple of six algorithms, as follows:

(1)SetUp(1^k):Given the security parameter k, KGC outputs the public parameters Ω and the public/secret key pair (PK, SK) for the traditional public key algorithm.

(2)KeyGen (Ω, O, U, C):For the data owner set O , data user set U and cloud server, KGC generates the public/secret key pairs $\{PK_i, SK_i\}$ $(1 \leq i \leq d)$, $\{PK_u, SK_u\}$ and $\{PK_C, SK_C\}$, respectively .

a)Set-Secret-Value:After inputting the public parameters Ω, this probabilistic algorithm outputs Data owners, Data users, and CSP's Secret-Value.

b)Partial-Private-Key-Extract:The KGC executes this algorithm, which accepts the identity of the data owner, data user, and CSP, then uses them in combination with the master key to generate a partial private key for the data owner, data user, and CSP.

c)Set-private-Key:Set the full secret keys of the data owner, data user, and CSP.

d)Set-Public-Key:Set the full public keys of the data owner, data user, and CSP.

(3)$Enc(\Omega, F, W, \{ID_i\}, ID, \{SK_i\}, \{PK_i\})$:Data owners first conduct this probabilistic algorithm to generate the ciphertext set C for the set F. Then data owners generate multiple signatures Sig and index set I for ciphertext C. Then he sends the tuple (C, I, Sig) to CSP.

(4)$TraGen(\Omega, SK_u, W')$:Given the keyword W' , the data user runs this algorithm to output trapdoor $T_{W'}$.

(5)$Test(\Omega, T_{W'}, I)$:Using the trapdoor $T_{W'}$ as an input, the CSP matches it with the index set I, then returns the relevant ciphertext $C' \subseteq C$ and signature Sig' set to PAS.

 doi:10.20944/preprints202401.2177.v1

(6)$Verify(\Omega, Sig, C_{W\prime}, PK_{O_i})$:PAS runs this algorithm by initiating interaction with CSP to check the correctness of the search result $C_{W\prime}$. If $C_{W\prime}$ passes the result validation, PAS will return it to the data user. Otherwise, it will abort the algorithm.

3.4. Security Model

To protect the security and privacy of the scheme, we must satisfy the following requirements:

(1)Keyword ciphertext indistinguishability: When encrypted data is stored in CSP, it will attach the corresponding keywords $\{w_{i1}, w_{i2}, \cdots, w_{im}\}$. Even if keyword ciphertext is captured during transmission, no adversary can obtain keywords embedded in the ciphertext.

(2)Indistinguishability of trapdoor:Any adversary shall not obtain any information from the trapdoor.

Our scheme proves that trapdoors are indistinguishable and the keywords ciphertext are indistinguishable in the standard model. They are defined as follows:

ciphertext indistinguishability against chosen keyword ciphertext attack: We will define the definition of ciphertext indistinguishability against chosen keyword and ciphertext attack (CKCA-CIND). There are two types of adversaries: external adversaries (receivers) and internal adversaries (servers). A_1 and A_2 represent these two adversaries, and their attack methods are as follows:

A_1:A_1 doesn't know the master key, but A_1 can replace any user's public key.

A_2:A_2 knows the master key, but A_2 cannot replace any user's public key.

Call them the adversary of type-1 and the adversary of type-2. There are two games to discuss the security of CKCA-CIND.

Game I. A_1 simulates malicious users and B is the challenger. B and A_1 play this game together.

Setup:B running SetUp(1^k) program to get public parameters Ω and the public/secret key pair (PK, SK).B sends the public key PK to A_1 and keeps the master private key SK. Then B sets the key pair of O_i $(i \in \{1, 2, \cdots, d\})$ and CSP, i.e., (PK_{O_i}, SK_{O_i}) $(i \in \{1, 2, \cdots, d\})$ and (PK_c, SK_c) .B sends the public key PK_{O_i} $(i \in \{1, 2, \cdots, d\})$ and PK_c to A_1 , while SK_{O_i} and SK_c are unknown to A_1.

Phase 1. A_1 executes the User-Public-Key query using date user's identity ID_u first and then executes other queries using the identity ID_u. Set up lists to store the above queries and answers. All lists are initially empty. A_1 makes the queries to the challenger B as following:

(1)User-Public-key query:When A_1 inputs the identity ID_u, B outputs the user's public key PK_u .

(2)Replace - Public - Key query: A_1 inputs (ID_j, PK_j') ,B replaces PK_j with PK_j' .

(3)Secret-Value query:When A_1 inputs the identity ID_j , B returns the secret value corresponding to the ID_j. If PK_j is replaced, B refuses to answer.

(4)Partial-Private-Key-Extract query:When A_1 enters the ID_j, if $ID_j = ID_u^{\diamondsuit}$($ID_u^{\diamondsuit}$ is the challenge identity), B fails and stops. Otherwise, B returns the corresponding Partial Private Key.

(5)Keyword Ciphertext Query: A_1 asks B for the ciphertext of any keyword W it cares about. B runs the $Enc(\Omega, F, W, \{ID_i\}, ID, \{SK_i\}, \{PK_i\})$ algorithm to answer W's ciphertext C_W.

(6)Keyword Trapdoor Query: A_1 sends a keyword W' to B. B runs the $TrapGen(\Omega, SK_u, W')$ algorithm to answer W''s trapdoor $T_{W'}$.

(7)Test Query: A_1 selects and sends the ciphertext C_W and trapdoor $T_{W'}$ to B. B executes the $Test(\Omega, T_{W'}, I)$ algorithm to return the test result of whether the ciphertext and the trapdoor match.

Challenge: A_1 submits a tuple $(W_0, W_1, ID_U{}^*, PK_U{}^*)$ to B, where W_0 and W_1 are challenging keywords not asked in the previous trapdoor and ciphertext query. If $ID_u^* \neq ID_u^{\diamondsuit}$, B aborts. Otherwise, B picks $\xi \in \{0, 1\}$ randomly computes keyword trapdoor C_{W_ξ}, and returns the challenge ciphertexts C_{W_ξ} to A_1.

Phase 2. A_1 can perform many queries like Phase 1, but A_1 cannot query the ciphertext and trapdoor of W_0 and W_1.

Guess: A_1 outputs $\xi' \in \{0, 1\}$. A_1 wins if $\xi' = \xi$. Otherwise, it fails.

 doi:10.20944/preprints202401.2177.v1

Game 2. A_2 simulates the malicious server, and B is a challenger. B and A_2 play this game together.

Setup:It differs from Setup of Game 1 only in the following steps. B send spublic key $PK_{O_i}(i \in \{1,2,\cdots,d\})$ and public/secret key pair (PK, SK) to A_2, and SK_{O_i} are unknown to A_2.

Phase 1.The steps are the same as phase 1 of Game 1, except for Secret Value and Partial Key query. The changes in them are as follows:

Secret-Value query :When A_2 enters the ID_j, if $ID_j = ID_u^{\diamondsuit}$ ($ID_u^{\diamondsuit}$ is the challenge identity), B fails and stops. Otherwise, B returns the secret value corresponding to ID_j.

Partial-Private-Key-Extract query: When A_2 inputs the identity ID_j , B returns the partial private key corresponding to the ID_j .If PK_j is replaced, B refuses to answer.

Phase 2 :Same as Phase 2 of Game 1.

Definition 1(Security of CKCA-CIND): If the probability that any adversary will win the above two games in polynomial time is negligible, then we state that the CLVPFCKS scheme is CKCA-CIND safe.

Indistinguishability of trapdoor (IND-KGA):No adversary can obtain valuable information from trapdoors. A sufficient condition for ensuring security against offline key-guessing attacks is that the trapdoors are indistinguishable. In the following, we define the concept of indiscernibility of CLVPFCKS against keyword guessing attacks. Specifically, IND-KGA ensures that adversaries (servers or the receivers) cannot observe the connection between the trapdoors and any keywords.

Game 3. A_1 simulates malicious users and B is the challenger. B and A_1 play this game together.

Setup:A_1 B running SetUp(1^k) program to get public parameters Ω and the public/secret key pair (PK, SK). B sends the public key PK to A_1 and keeps the master private key SK. Then B sets the key pair of date user and CSP, i.e., (PK_u, SK_u) and (PK_c, SK_c) .The challenger B sends the public key PK_u and PK_c to A_1 , while SK_u and SK_c are unknown to A_1.

Phase 1. A_1 executes the User-Public-Key query using data owner's identity ID_i first and then executes other queries using the identity ID_i. Set up lists to store the above queries and answers. All lists are initially empty. A_1 makes the queries to the challenger B as following:

(1)User-Public-key query:When A_1 inputs the identity ID_j, B outputs the user's public key PK_j .

(2)Replace-Public-Key query: same as that in Game 1.

(3)Secret-Value query:same as that in Game 1.

(4)Partial-Private-Key-Extract query:When A_1 enters the ID_j, if $ID_j = ID_i^{\diamondsuit}$($ID_i^{\diamondsuit}$ is the challenge identity), B fails and stops. Otherwise, B returns the corresponding Partial Private Key.

(5)Keyword Ciphertext Query:same as that in Game 1.

(6)Keyword Trapdoor Query:same as that in Game 1.

(7)Test Query: same as that in Game 1.

Challenge :A_1 submits a tuple $(W_0, W_1, ID_i^*, PK_i^*)$ to B, where W_0 and W_1 are challenging keywords not asked in the previous trapdoor and ciphertext query. If $ID_i^{\diamondsuit} \notin \{ID_i^*\}(i \in \{1,2,\cdots d\})$, B aborts. Otherwise, B picks $\xi \in \{0,1\}$ randomly computes keyword trapdoor T_{W_ξ}, and returns the challenge ciphertexts T_{W_ξ} to the adversary A_1.

Phase 2. A_1 can perform many queries like Phase 1, but A_1 cannot query the ciphertext and trapdoor of W_0 and W_1.

Guess: A_1 outputs $\xi' \in \{0,1\}$. Adversary A_1 wins if $\xi' = \xi$. Otherwise, it fails.

Game 4. A_2 simulates the malicious server, and B is a challenger. B and A_2 play this game together.

Setup:It differs from Setup of Game 3 only in the following steps. B send spublic key PK_u and public/secret key pair (PK, SK) to A_2, and SK_u are unknown to A_2.

Phase 1. The steps are the same as phase 1 of Game 3, except for Secret Value and Partial Key query. The changes in them are as follows:

Secret-Value query: When A_2 enters the ID_j, if $ID_j = ID_i^{\diamondsuit}$ ($ID_i^{\diamondsuit}$ is the challenge identity), B fails and stops. Otherwise, B returns the secret value corresponding to ID_j.

 doi:10.20944/preprints202401.2177.v1

Partial-Private-Key-Extract query: same as that in Game 2.

Phase 2. Same as Phase 2 of Game 3.

Definition 2 (Security of IND-KGA) :If the probability that any adversary will win the above two games in polynomial time is negligible, then we state that the CLVPFCKS scheme is IND-KGA safe.

4. the proposed CLVPFCKS

To better understand about our proposed scheme, we explain the pre-defined notations used throughout this paper in Table 1.

Table 1. Notation descriptions

Notations	DESCRIPTIONS
x	Master key
P_{pub}	system public key
$O = \{O_1, O_2, \cdots, O_d\}$	Data Owner Collection
$ID_i \in \{0,1\}^*(1 \leq i \leq d)$	identity set for data owner O_i
ID_C	identity set for CSP
ID_u	identity set for data user
(PK_C, SK_C)	Public/secret key pair for CSP
(PK_{O_i}, SK_{O_i})	Public/secret key pair for data owner O_i
(PK_u, SK_u)	Public/secret key pair for data user
$F = \{f_1, f_2, \cdots, f_n\}$	File set F
$C = \{c_1, c_2, \cdots, c_n\}$	ciphertext set
$ID = \{id_1, id_2, \cdots, id_n\}$	identity set for F
$W_i = \{w_{i1}, w_{i2}, \cdots, w_{im}\}$	Collection of keywords
$sig_{i,t}$	Data owner O_i' signature for c_t
sig_t	Data owners' multi-signatures for c_t
$Sig = \{sig_1, sig_2, \cdots, sig_n\}$	F's multi-signature
I_i	Index of f_i
$I = \{I_1, I_2, \cdots, I_n\}$	Index set for F
$W' = \{w_1', w_2', \cdots, w_l'\}$	Search keyword set
$T_{W'}$	The trapdoor of W'
$C' = \{c_{k_1}, c_{k_2}, \cdots, c_{k_s}\}$	Search results
$ID' = \{id_{k_1}, id_{k_2}, \cdots, id_{k_s}\}$	Identity set for C'

4.1. *Specific construction of CLVPFCKS*

This system uses traditional public key encryption algorithm to encrypt files, which we will not discuss. Therefore, the following algorithms focus on indexing and signature.

SetUp(1^k) :Given a security parameter k, this deterministic algorithm outputs the global public parameters Ω and PKG's master private key (MSK). Given k, PKG performs in the following way:

(1)Chooses a k bits prime number q and determine the tuple $\{F_q, E/F_q, G_q, P\}$, where the point P is the generator of G_q .

(2) Chooses the master key $x \in_R Z_q^*$ and computes the system public key $P_{pub} = xP$. Let $(PK, SK) = (P_{pub}, x)$.

(3) Selects five hash functions H_0,H_1,H_2 :$\{0,1\}^* \times G \times G \to Z_q^*$, $h_1 : \{0,1\}^* \to Z_q^*$,$h_2 : G_q \to Z_q^*$.

(4) Let $\Omega = \{F_q, E/F_q, G_q, P, H_0, H_1, H_2, h_1, h_2, P_{pub}\}$.

KeyGen(Ω, O, U, CSP):Each EHR has a fixed number of data owners $O = \{O_1, O_2, \cdots, O_d\}$, then PKG generates the public/secret key pairs for CSP, data owner set O and data user U, respectively.

(1)Set-Secret-Value: The Participant with $ID_i (i = 1, 2, \cdots, d, C, u)$ selects an element $x_i \in_R Z_q^* (i = 1, 2, \cdots, d, C, u)$ and generates the corresponding public key $P_i = x_i P (i = 1, 2, \cdots, d, C, u)$.

(2))Extract-Partial-Private-Key: To get the partial private key, the user ID_i sends (ID_i, P_i) to the PKG and then the PKG executes the following steps.

 doi:10.20944/preprints202401.2177.v1

(a)Taking the participant's $ID_i(i = 1,2,\cdots,d,C,u)$ as input, KGC selects a random number $r_i \in Z_q^*(i = 1,2,\cdots,d,C,u)$ and calculates $R_i = r_iP(i = 1,2,\cdots,d,C,u)$.

(b)PKG computes $e_i = r_i + xH_0(ID_i, R_i, P_i)modq(i = 1,2,\cdots,d,C,u)$. The partial private key of the participant with $ID_i(i = 1,2,\cdots,d,C,u)$ is e_i. The participant with $ID_i(i = 1,2,\cdots,d,C,u)$ can verify his partial private key by checking whether the equation $Q_i = e_iP = R_i + l_iP_{pub}$ holds, where $l_i = H_0(ID_i, R_i, P_i)$. If the above equation is true, then the private key ID_i will accepted.

(3)Set-Private-Key: The partial private key of the participant with $ID_i(i = 1,2,\cdots,d,C,u)$ takes the pair $SK_i = (x_i, e_i)$ as his full private key.

(4)Set-Public-Key: The participant with $ID_i(i = 1,2,\cdots,d,C,u)$ takes $PK_i = (P_i, R_i)$ as his full public key.

Enc$(\Omega, F, W, \{ID_i\}, ID, \{SK_i\}, \{PK_i\})$:

Step 1: Given the EHR set $F = \{f_1, f_2, \cdots, f_n\}$ with corresponding identities $ID = \{id_1, id_2, \cdots, id_n\}$, it will be encrypted as the ciphertext set $C = \{c_1, c_2, \cdots, c_n\}$ through the traditional public key encryption algorithm. To generate the multi-signature on the encrypted file $c_t \in C(1 \le t \le n)$, each signer $O_i(1 \le i \le d)$ does the following:

(1) O_i chooses a number $y_{i,t} \in_R Z_q^*$ and computes $Y_{i,t} = y_{i,t}P$.

(2)O_i broadcasts $Y_{i,t}$ to other members $O_k(1 \le k \le d, k \ne i)$ of the group.

(3)computes $Y_{0,t} = \sum_{i=1}^{d} Y_{i,t}$, $P_0 = \sum_{i=1}^{d} P_i$, $Q_0 = \sum_{i=1}^{d} Q_i$.

(4)computes $h_t = H_1(c_t, id_t, Y_{0,t}, P_0)$ and $h_t' = H_1(c_t, id_t, Y_{0,t}, Q_0)$.

(5)O_i computes $V_{i,t} = y_{i,t} + h_tx_i + h_t'e_i$, generates a signature $sig_{i,t} = (Y_{i,t}, V_{i,t})$ for c_t, and then sends $V_{i,t}$ to the designated clerk O_z.

(6)Upon receiving $V_{i,t}$, O_z computes $V_{0,t} = \sum_{i=1}^{d} V_{i,t}$ and outputs the signature $sig_t = \{Y_{0,t}, V_{0,t}\}$. Let $sig = \{sig_1, \cdots, sig_n\}$, where $V_{0,t}P = Y_{0,t} + h_tP_0 + h_t'Q_0$.

Step 2: All users specify a user to encrypt files, for example, O_j. O_j runs this algorithm to generate the index of file set F. Given the keyword set W, O_j builds an index for each file $f_i \in F$. The index for each f_i is generated based on the keyword field $W = \{w_{i1}, w_{i2}, \cdots, w_{im}\}$, where m is a fixed integer. O_j randomly selects $\xi \in Z_q^*$ and calculates $\xi + x_j$ to O_1 through public key encryption. O_1 computes $\xi + x_j + x_1$ and sends it to O_2 through public key encryption. O_2 computes $\xi + x_j + x_1 + x_2$ and sends it to O_3 through public key encryption. $\cdots$, O_d computes $\xi + x_j + x_1 + \cdots x_{j-1} + x_{j+1} + \cdots + x_d$ and sends it to O_j through public key encryption. O_j calculates $x_0 = \xi + x_j + x_1 + \cdots x_{j-1} + x_{j+1} + \cdots + x_d - \xi = x_1 + x_2 + \cdots + x_d$. Calculates $e_0 = e_1 + e_2 + \cdots + e_d$ in the same way as O_j.

Let $R_0 = \sum_{t=1}^{d} R_t$, $l_0 = \sum_{t=1}^{d} l_t$, Construct an m-degree polynomial by the following equation: $F(x) = b_{i,m}x^m + b_{i,m-1}x^{m-1} + \cdots + b_{i,1}x + b_{i,0}$. $h_2(t)h_1(w_{i,1}), \cdots, h_2(t)h_1(w_{i,m})$ is the root of equation $F(x) = 1$, where $t = (x_0 + e_0)P_u + x_0R_u + x_0l_uP_{pub}$.

Then O_j selects $\lambda_i, \mu_i \in_R Z_q{}^*$ and computes $M_i = \lambda_iQ_c$, set $I_{i,1} = M_i - \mu_iP$, $I_{i,2} = \lambda_iP$, $V_{i,j} = \mu_ib_{i,j}(0 \le j \le m)$, and the index set is $I = \{I_1, \cdots, I_n\}$, where $I_i = \{I_{i,1}, I_{i,2}, V_{i,0}, V_{i,1}, \cdots, V_{i,m}\}$. Finally O_j send I to CSP.

TrapGen(Ω, SK_u, W') :

The Data user calculates the value of P_0, R_0, l_0 as follows $P_0 = \sum_{i=1}^{d} P_i$, $R_0 = \sum_{i=1}^{d} R_i$, $l_0 = \sum_{i=1}^{d} l_i$. Given the queried keywords set $W' = \{w_1', w_2', \cdots, w_l'\}$, the data user U first selects an elements $\eta \in_R Z_q^*$ and sets $T_{W'_{m+1}} = \eta P$,$T_{W'_j} = l^{-1}h_2(t)^j \sum_{r=1}^{l} h_1(w'_r)^j P + \eta P_C$, where $0 \le j \le m$, $t = (x_u + e_u)P_0 + x_uR_0 + x_ul_0P_{pub}$. Finally, he sent $T_{w'} = \{T_{w'_0}, T_{w'_1}, \cdots, T_{w'_m}, T_{w'_{m+1}}\}$ to CSP.

 doi:10.20944/preprints202401.2177.v1

test$(\Omega, T_{W'}, I, C)$: After gaining the search token $T_{W'}$, the CSP first computes $M_i = \lambda_i Q_C$, and verifies whether Eq.(1) holds.

$$I_{i,1} + \sum_{j=0}^{m} V_{i,j}(T_{w'_j} - x_C T_{w'_{m+1}}) = M_i \tag{1}$$

If Eq (1) holds, the CSP returns the relevant ciphertext set $C' = \{c_{k_1}, c_{k_2}, \cdots, c_{k_s}\}$ and its corresponding identity set $ID' = \{id_{k_1}, id_{k_2}, \cdots, id_{k_s}\}$ to PAS. Otherwise, it returns $\perp$. The specific test process is shown in the Algorithm 1.

Algorithm 1 Search over encrypted data

Input: Trapdoor $T_{W'}$, indexes I, ciphertexts C, secret key SK_C and public parameters Ω .

Output: Search results C' and corresponding identity set ID' or $\perp$.

(1) $T_{w'} = \{T_{w'_0}, T_{w'_1}, \cdots, T_{w'_m}, T_{w'_{m+1}}\}$

(2)$I = \{I_1, I_2, \cdots, I_n\}$,$I_i = \{I_{i,1}, I_{i,2}, V_{i,0}, V_{i,1}, \cdots, V_{i,m}\}$.

(3)$M_i = e_C I_{i,2}$

(4)$for\ 0 \le i \le n\ do$

$$I_{i,1} + \sum_{j=0}^{m} V_{i,j}(T_{w'_j} - x_C T_{w'_{m+1}}) = M_i$$

(5) If Eq.(1) holds, CSP returns the ciphertext c_{k_t} ; otherwise, it returns $\perp$;

(6)end for

(7)CSP returns the relevant results C' and corresponding identity set ID' or $\perp$ to PAS.

Verify(Ω, C', ID', sig):After receiving the search results C' , CSP computes the proof information (ϕ_1, ϕ_2) through Eq.(2) and sends it to PAS. Finally, PAS verifies whether Eq.(3) holds.

Algorithm 2 results verification

Input: Search results C' with corresponding identity set ID', public keys $\{PK_i\}$, signature $sig = \{sig_1, \cdots, sig_n\}$ and public parameters Ω, where $sig_t = \{Y_{0,t}, V_{0,t}\}$.

Output: "Accept" or "Reject"

()1$C' = \{c_{k_1}, c_{k_2}, \cdots, c_{k_s}\}$,$ID' = \{id_{k_1}, id_{k_2}, \cdots, id_{k_s}\}$;

(2)$\{PK_1, PK_2, \cdots, PK_d\}$;

(3)$sig = \{sig_1, \cdots, sig_n\}$, $sig_t = \{Y_{0,t}, V_{0,t}\}$;

(4)compute $P_0 = \sum_{t=1}^{d} P_t$, $Q_0 = \sum_{t=1}^{d} Q_t$;

(5)compute $\phi_1 = \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, P_0)$,$\phi_2 = \sum_{\tau=1}^{s} H_2(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, Q_0)$,

$$\sigma = \sum_{\tau=1}^{s} V_{0,k_\tau}; \tag{2}$$

(6)Check

$$\sigma P = \sum_{\tau=1}^{s} Y_{o,k_\tau} + \phi_1 P_0 + \phi_2 Q_0 \tag{3}$$

(7)If Eq.(3) holds, output "Accept" and send C' to data user; otherwise, output "Reject".

5. Security of scheme

In this section, we will analyze the security and correctness of this scheme.

5.1. Correctness

Theorem 1. Our CLVPFCKS scheme is computationally consistent.

 doi:10.20944/preprints202401.2177.v1

Proof: For the correctness of Our CLVPFCKS scheme, we do two things. First, We show that CSP can correctly test whether the index of ciphertext matches the trapdoor when the keyword set $W' \subseteq W$, where W' is a set of keywords searched by a specific user and W is the keyword set of ciphertext. Secondly, we will show that if the search results pass the result verification mechanism, data users can guarantee the correctness of the search results.

In the test phase, CSP obtains the index $I = \{I_1, I_2, \cdots, I_n\}$ and trapdoor $T_{W'} = \{T_{W'_0} T_{W'_1}, \cdots, T_{W'_m}, T_{W'_{m+1}}\}$. The CSP first computes

$e_C I_{i,2} = (r_C + x l_C)\lambda_i P = \lambda_i \, Q_C = M_i$.

$$
\begin{aligned}
& I_{i,1} + \sum_{j=0}^{m} V_{i,j}(T_{w'_j} - x_C T_{w'_{m+1}}) \\
&= M_i - \mu_i P + \sum_{j=0}^{m} \mu_i b_{ij} [l^{-1} h_2(t)^j \sum_{r=1}^{l} h_1(w'_r)^j P + \eta P_C - x_C \eta P] \\
&= M_i - \mu_i P + \sum_{j=0}^{m} \mu_i b_{ij} (l^{-1} h_2(t)^j \sum_{r=1}^{l} h_1(w'_r) P) \\
&= M_i - \mu_i P + l^{-1} \mu_i \sum_{j=0}^{m} b_{ij} h_2(t)^j \sum_{r=1}^{l} h_1(w'_r)^j P \\
&= M_i - \mu_i P + l^{-1} \mu_i [\sum_{j=0}^{m} b_{ij} h_2(t)^j h_1(w'_1)^j + \cdots + \sum_{j=0}^{m} b_{ij} h_2(t)^j h_1(w'_l)^j] P
\end{aligned}
$$

If $W' \subseteq W$, then $h_2(t)h_1(w_1), \cdots, h_2(t)h_1(w_l)$ are the root of the equation $F(x) = 1$, where $F(x) = b_{i,m}x^m + b_{i,m-1}x^{m-1} + \cdots + b_{i,1}x + b_{i,0}$. Thus

$$
\begin{aligned}
& I_{i,1} + \sum_{j=0}^{m} V_{i,j}(T_{w'_j} - x_C T_{w'_{m+1}}) \\
&= M_i - \mu_i P + l^{-1} \mu_i [\sum_{j=0}^{m} b_{ij} h_2(t)^j h_1(w'_1)^j + \cdots + \sum_{j=0}^{m} b_{ij} h_2(t)^j h_1(w'_l)^j] P \\
&= M_i - \mu_i P + l^{-1} \mu_i (1 + 1 + \cdots + 1) P \\
&= M_i - \mu_i P + \mu_i P \\
&= M_i
\end{aligned}
$$

Equation (1) holds so CSP can correctly test whether the index of ciphertext matches the trapdoor.

In the test phase, PAS obtains signature $sig = \{sig_1, sig_2, \cdots, sig_n\}$ and ciphertext $C' = \{c_{k_1}, c_{k_2}, \cdots, c_{k_s}\}$, computing

$$\phi_1 = \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, P_0),$$

$$\phi_2 = \sum_{\tau=1}^{s} H_2(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, Q_0)$$

getting the proof information (ϕ_1, ϕ_2), and then continuing to calculate

$$
\begin{aligned}
\sigma P &= \sum_{\tau=1}^{s} V_{0,k_\tau} P = \sum_{\tau=1}^{s} (Y_{0,k_\tau} + h_{k_\tau} P_0 + h'_{k_\tau} Q_0) \\
&= \sum_{\tau=1}^{s} Y_{0,k_\tau} + \sum_{\tau=1}^{s} h_{k_\tau} P_0 + \sum_{\tau=1}^{s} h'_{k_\tau} Q_0 \\
&= \sum_{\tau=1}^{s} Y_{0,k_\tau} + \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, P_0) P_0 + \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, Q_0) Q_0.
\end{aligned}
$$

If $C' \subseteq C$, then

$$\phi_1 = \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, P_0) = \sum_{\tau=1}^{s} H_1(c_{\rho(\tau)}, id_{\rho(\tau)}, Y_{0,\rho(\tau)}, P_0)$$

$$\phi_2 = \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, Q_0) = \sum_{\tau=1}^{s} H_1(c_{\rho(\tau)}, id_{\rho(\tau)}, Y_{0,\rho(\tau)}, Q_0)$$

Where $\rho(\tau) \in [1, n]$. So we have $\sigma P = \sum_{\tau=1}^{s} Y_{o,k_\tau} + \phi_1 P_0 + \phi_2 Q_0$. Eq. (3) in program 2 holds. We can also make sure that the ciphertext can not modified.

 doi:10.20944/preprints202401.2177.v1

5.2. Security

Lemma 1 . Assuming the adversary A_1 can win Game 1, then an algorithm B can be constructed to solve the ECDDDH problem.

Proof: Suppose that the tuple (P, aP, bP, X) is an example of ECDDDH problem. To determine whether $X = abP$, B will play the part of the challenger.

Set up : B runs the setup(1^k) program to get public parameters $\Omega = \{F_q, E/F_q, G_q, P, G, H_0, H_1, H_2, P_{pub}\}$, where master private key $SK = x$ and the public key $PK = P_{pub} = xP$. Then B sends parameter Ω to A_1 , and the master private key SK is kept secret. B selects $x_i \in Z_q^*(i \in \{2, \cdots, d, C\})$, $r_i \in Z_q^*(i \in \{1, 2, \cdots, d, C\})$ randomly and set

$P_i = x_i P \ (i \in \{2, \cdots, d, C\})$,

$P_1 = aP, R_i = r_i P \ (i \in \{1, 2, \cdots, d, C\})$,

$e_i = r_i + xH_0(ID_i, R_i, P_i) \bmod q(i \in \{1, 2, \cdots, d, C\})$.

B sends the public key PK_{O_i} $(i \in \{1, 2, \cdots, d\})$ and PK_C to A_1, but SK_{O_i} $(i \in \{1, 2, \cdots, d\})$ and SK_C are unknown to A_1.

Phase 1: Executed the user's public key query before other queries using the identity ID_u. Set up multiple lists to store the queries and answers. All lists are initially empty.

User public key query: B keeps a list L_u of the tuple $(ID_u, x_u P, r_u P, r_u)$ and upon receiving an identity ID_u, performs the following steps.

Case1. $ID_u = ID_u^{\diamondsuit}$. B picks at randomly $x_u^{\diamondsuit} \in Z_q^*$, setting $PK_u^{\diamondsuit} = (x_u^{\diamondsuit} P, bP)$, and adds the tuple $(ID_u^{\diamondsuit}, x_u^{\diamondsuit} P, x_u^{\diamondsuit}, bP, \diamondsuit)$ to the list L_u, Where $\diamondsuit$ represents a null value.

Case2. $ID_u \neq ID_u^{\diamondsuit}$. B picks at randomly $x_u, r_u \in Z_q^*$, setting $PK_u = (x_u P, r_u P)$, and adds the tuple $(ID_u, x_u P, r_u P, r_u)$ to the list L_u.

Replace-Public-Key query: B maintains a list L_R of tuple $(ID_j, PK_j, PK_j{}')$. When A_1 inputs $(ID_j, PK_j{}')$, B replaces PK_j with $PK_j{}'$, and adds $(ID_j, PK_j, PK_j{}')$ to the list L_R.

Secret-Value query: When A_1 asks the secret value for ID_j, B finds $(ID_j, x_j P, r_j P, r_j)$ in the list L_u and returns x_j.If PK_j is replaced, B refuses to answer.

Partial-Private-Key query: B establishes a list L_e of tuple (ID_j, e_j) when A_1 asks the partial private key of ID_j. If $ID_j = ID_u^{\diamondsuit}$, B fails and stops. Otherwise, B finds $(ID_j, x_j P, r_j P, r_j)$ in the list L_u , running the Extract-Partial-Private-Key algorithm generating e_j. B output e_j and adds (ID_j, e_j) to the list L_e.

Keyword Ciphertext Query: When A_1 asks $W = \{w_{i,1}, w_{i,2}, \cdots, w_{i,m}\}$ for the ciphertext, B operates the $Enc(\Omega, F, W, \{ID_i\}, ID, \{SK_i\}, \{PK_i\})$ algorithm to generate ciphertext $C_W = \{I_{i,1}, I_{i,2}, V_{i,0}, V_{i,1}, \cdots, V_{i,m}\}$.

Keyword Trapdoor Query: When A_1 asks $W' = \{w_1', w_2', \cdots, w_l'\}$ for the trapdoor, B operates the $TrapGen\{\Omega, SK_u, W'\}$ algorithm to generate trapdoor $T_{w'} = \{T_{w'_0}, T_{w'_1}, \cdots, T_{w'_m}, T_{w'_{m+1}}\}$.

Test Query: A_1 gives keyword ciphertext $C_{Wi} = \{I_{i,1}, I_{i,2}, V_{i,0}, V_{i,1}, \cdots, V_{i,m}\}$ and keyword trapdoor $T_{w'} = \{T_{w'_0}, T_{w'_1}, \cdots, T_{w'_m}, T_{w'_{m+1}}\}$, and B compares them using Algorithm 1.

Challenge: A_1 submits a tuple $(W_0, W_1, ID_u{}^*, PK_u^*)$, where $W_0 = \{w_{0,1}, w_{0,2}, \cdots, w_{0,m}\}$ and $W_1 = \{w_{1,1}, w_{1,2}, \cdots, w_{1,m}\}$ are challenging keywords not requested in the previous trapdoor and ciphertext query. If $ID_u^* \neq ID_u^{\diamondsuit}$, B aborts. Otherwise $ID_u^* = ID_u^{\diamondsuit}$, B calculates $l_u{}^{\diamondsuit} = H_0(ID_u^{\diamondsuit}, R_u^{\diamondsuit}, P_u^{\diamondsuit})$ and picks $\xi \in \{0, 1\}$ randomly. B computes

$$t = (\sum_{k=2}^{d} x_k + \sum_{k=1}^{d} e_k) P_u^{\diamondsuit} + \sum_{k=2}^{d} x_k R_u^{\diamondsuit} + \sum_{k=2}^{d} x_k l_u^{\diamondsuit} P_{pub} + x_u^{\diamondsuit} aP + l_u^{\diamondsuit} xaP + X.$$

Let $F(x) = (x - h_2(t)h_1(w_{\xi,1}))(x - h_2(t)h_1(w_{\xi,2})) \cdots (x - h_2(t)h_1(w_{\xi,m})) - 1$, which can get $F(x) = b_{\xi,m}x^m + b_{\xi,m-1}x^{m-1} + \cdots + b_{\xi,1}x + b_{\xi,0}$ by combining similar terms. Then B selects $\lambda_\xi, \mu_\xi \in_R Z_q{}^*$ and computes $M_\xi = \lambda_\xi Q_c$. Set $I_{\xi,1} = M_\xi - \mu_\xi P, I_{\xi,2} = \lambda_\xi P$, $V_{\xi,j} = \mu_\xi b_{\xi,j} (0 \leq j \leq m)$. Thus, the corresponding ciphertext of $W_\xi = \{w_{\xi,1}, w_{\xi,2}, \cdots, w_{\xi,m}\}$ is $C_{W\xi} = \{I_{\xi,1}, I_{\xi,2}, V_{\xi,0}, V_{\xi,1}, \cdots, V_{\xi,m}\}$. B returns the challenge ciphertexts C_{W_ξ} to the adversary A_1.

 doi:10.20944/preprints202401.2177.v1

Phase 2: A_1 can continue to execute various queries, but there is a limitation that A_1 is not allowed to query the keyword ciphertext or trapdoor of W_0 or W_1.

Guess: A_1 returns ξ'.

Solve CDH problem: If $\xi' = \xi$, B returns 1, otherwise 0. If $X = abP$, then

$$t = (\sum_{k=2}^{d} x_k + \sum_{k=1}^{d} e_k)P_u^{\diamondsuit} + \sum_{k=2}^{d} x_k R_u^{\diamondsuit} + \sum_{k=2}^{d} x_k l_u^{\diamondsuit} P_{pub} + x_u^{\diamondsuit} aP + l_u^{\diamondsuit} xaP + X$$
$$= (x_0 + e_0)P_u^{\diamondsuit} + \sum_{k=2}^{d} x_k R_u^{\diamondsuit} + x_0 l_u^{\diamondsuit} P_{pub} + abP$$
$$= (x_0 + e_0)P_u^{\diamondsuit} + x_0 R_u^{\diamondsuit} + x_0 l_u^{\diamondsuit} P_{pub}$$

Therefore, C_{W_ξ} is a valid ciphertext. Suppose that the advantage of A_1 wins in the above game is ε. So

$$\Pr[\xi' = \xi \,|X = abP] = \tfrac{1}{2} + \varepsilon.$$

If $X \neq abP$, then C_{W_ξ} is an invalid ciphertext. A_1 has no advantage in distinguishing $\xi = 0$ from $\xi = 1$. Hence

$$\Pr[\xi' = \xi \,|X = abP] = \tfrac{1}{2}.$$

Probability: Let q_u, q_r and q_e be the number of the User public key query, Replace-Public-Key query, and Partial-Private-Key query, respectively. The two events are as follows:

π_1 : A_1 did not replace of $ID_u^{\diamondsuit}$'s public key $R_u^{\diamondsuit}$ and query the partial-private-key for $ID_u^{\diamondsuit}$.

π_2 : $ID_u^* = ID_u^{\diamondsuit}$.

It's not hard to get the following results.

$\Pr[\pi_1] = \frac{q_u - q_r - q_e}{q_u}$,

$\Pr[\pi_2 \,|\pi_1] = \frac{1}{q_u - q_r - q_e}$,

$\Pr[B \; success] = \Pr[\pi_1 \wedge \pi_2] = \frac{1}{q_u}$.

If A_1 win Game 1 with an advantage of ε, then B has a probability greater than $\frac{\varepsilon}{q_u}$ to determine whether $X = abP$.

Lemma 2 . Assuming the adversary A_2 can win Game 2, an algorithm B can constructed to solve the ECDDDH problem by exploiting the adversary's ability.

Proof: Suppose that the tuple (P, aP, bP, X) is an example of an ECDDDH problem. To determine whether $X = abP$, B will play the part of the challenger.

Set up: B runs the setup(1^k) program to get public parameters $\Omega = \{F_q, E/F_q, G_q, P, G, H_0, H_1, H_2, P_{pub}\}$, where master private key $SK = x$ and the public key $PK = P_{pub} = xP$. B selects $x_i \in Z_q^*(i \in \{1, 2, \cdots, d, C\})$, $r_i \in Z_q^*(i \in \{2, \cdots, d, C\})$ randomly and set

$P_i = x_i P \; (i \in \{1, 2, \cdots, d, C\})$,

$R_1 = aP$,$R_i = r_i P \; (i \in \{2, \cdots, d, C\})$,

$e_1 = a + xH_0(ID_1, R_1, P_1) \bmod q$, $e_i = r_i + xH_0(ID_i, R_i, P_i) \bmod q (i \in \{2, \cdots, d, C\})$

B sends the public parameters Ω, the public key $PK_{O_i}(i \in \{1, 2, \cdots, d\})$ and the public/secret key pair (PK, SK) to A_2, while $SK_{O_i}(i \in \{1, 2, \cdots, d\})$ are unknown to A_2.

Phase 1: Executed the user's public key query before other queries using the identity ID_u. Set up multiple lists to store the queries and answers. All lists are initially empty.

User public key query: B maintains a list L_u containing the tuple $(ID_u, x_u P, r_u P, r_u)$ and takes the following actions when receiving an identity ID_u:

Case1. $ID_u = ID_u^{\diamondsuit}$. B chooses a number $r_u^{\diamondsuit} \in Z_q^*$ at random, sets $PK_u^{\diamondsuit} = (bP, r_u^{\diamondsuit} P)$, and adds the tuple $(ID_u^{\diamondsuit}, bP, \diamondsuit, r_u^{\diamondsuit} P, r_u^{\diamondsuit})$ to the list L_u, Where $\diamondsuit$ represents a null value.

Case2. $ID_u \neq ID_u^{\diamondsuit}$. B chooses $x_u, r_u \in Z_q^*$ at random, sets $PK_u = (x_u P, r_u P)$, and adds the tuple $(ID_u, x_u P, r_u P, r_u)$ to the list L_u.

Replace-Public-Key query: same as that in Lemma 1

Secret-Value query: B established a list L_s of tuple (ID_j, x_j). When A_2 asks the secret value for ID_j. If $ID_j = ID_u^{\diamondsuit}$, B fails and stops. Otherwise, B finds $(ID_j, x_j P, r_j P, r_j)$ in list L_u, returns x_j.

 doi:10.20944/preprints202401.2177.v1

Partial-Private-Key query: When A_2 asks the partial private key of ID_j, B finds (ID_j, x_jP, r_jP, r_j) in list L_u, running the Extract-Partial-Private-Key algorithm and returning e_j.If PK_j is replaced, B refuses to answer.

Keyword Ciphertext Query: same as that in Lemma 1.

Keyword Trapdoor Query: same as that in Lemma 1.

Test Query: same as that in Lemma 1.

Challenge: A_2 submits a tuple $(W_0, W_1, ID_u^*, PK_u^*)$ that meets the requirements of Game 2, where $W_0 = \{w_{0,1}, w_{0,2}, \cdots, w_{0,m}\}$ and $W_1 = \{w_{1,1}, w_{1,2}, \cdots, w_{1,m}\}$ are challenging keywords not asked in the previous trapdoor query and ciphertext query. If $ID_u \neq ID_u^{\diamondsuit}$, B aborts. Otherwise $ID_u^* = ID_u^{\diamondsuit}$, B computes $l_u^{\diamondsuit} = H_0(ID_u^{\diamondsuit}, R_u^{\diamondsuit}, P_u^{\diamondsuit})$, $l_1 = H_0(ID_1, R_1, P_1)$ and picks $\xi \in \{0,1\}$ randomly. B computes

$$t = (\sum_{k=1}^{d} x_k + xl_1 + \sum_{k=2}^{d} e_k)P_u^{\diamondsuit} + \sum_{k=1}^{d} x_k R_u^{\diamondsuit} + \sum_{k=1}^{d} x_k l_u^{\diamondsuit} P_{pub} + X.$$

Let $F(x) = (x - h_2(t)h_1(w_{\xi,1}))(x - h_2(t)h_1(w_{\xi,2})) \cdots (x - h_2(t)h_1(w_{\xi,m})) - 1$, which can get $F(x) = b_{\xi,m}x^m + b_{\xi,m-1}x^{m-1} + \cdots + b_{\xi,1}x + b_{\xi,0}$ by combining similar terms. Then select $\lambda_\xi, \mu_\xi \in_R Z_q^*$ at random and compute $M_\xi = \lambda_\xi Q_c$. Set $I_{\xi,1} = M_\xi - \mu_\xi P$, $I_{\xi,2} = \lambda_\xi P$, $V_{\xi,j} = \mu_\xi b_{\xi,j} (0 \leq j \leq m)$, and thus $W_\xi = \{w_{\xi,1}, w_{\xi,2}, \cdots, w_{\xi,m}\}$'s ciphertext is $C_{W_\xi} = \{I_{\xi,1}, I_{\xi,2}, V_{\xi,0}, V_{\xi,1}, \cdots, V_{\xi,m}\}$. B returns the challenge ciphertexts C_{W_ξ} to the adversary A_2.

Phase 2:The attacker A_2 can continue to execute various queries, but there is a limitation that the attacker A_2 is not allowed to query the keyword ciphertext or trap of W_0 or W_1.

Guess: A_2 returns ξ'.

Solve the ECDDDH problem. If $\xi' = \xi$, B returns 1. Otherwise 0. If $X = abP$, then

$$t = (\sum_{k=2}^{d} x_k + \sum_{k=1}^{d} e_k)P_u^{\diamondsuit} + \sum_{k=2}^{d} x_k R_u^{\diamondsuit} + \sum_{k=2}^{d} x_k l_u^{\diamondsuit} P_{pub} + r_u^{\diamondsuit} aP + l_u^{\diamondsuit} xaP + X$$
$$= (\sum_{k=2}^{d} x_k + e_0)P_u^{\diamondsuit} + x_0 R_u^{\diamondsuit} + x_0 l_u^{\diamondsuit} P_{pub} + abP$$
$$= (x_0 + e_0)P_u^{\diamondsuit} + x_0 R_u^{\diamondsuit} + x_0 l_u^{\diamondsuit} P_{pub}$$

Therefore, C_{W_ξ} is a valid ciphertext. Suppose that the advantage of A_2 wins in the above game is ε, so

$$\Pr[\xi' = \xi \,|X = abP] = \tfrac{1}{2} + \varepsilon.$$

If $X \neq abP$, then C_{W_ξ} is an invalid ciphertext. A_2 has no advantage in distinguishing $\xi = 0$ from $\xi = 1$. Hence

$$\Pr[\xi' = \xi \,|X = abP] = \tfrac{1}{2}.$$

Probability. Let q_u, q_r, q_s be the number of User public key query, Replace-Public-Key query, and Secret-Value query, respectively. The two events are as follows:

π_1 : A_2 did not replace of $ID_u^{\diamondsuit}$'s public key $P_u^{\diamondsuit}$ nor perform the Secret-Value query for $ID_u^{\diamondsuit}$.

π_2 : $ID_u^* = ID_u^{\diamondsuit}$.

It's not hard to get the following results.

$\Pr[\pi_1] = \frac{q_u - q_r - q_s}{q_u}$,

$\Pr[\pi_2 \,|\pi_1] = \frac{1}{q_u - q_r - q_s}$,

$\Pr[B \; success] = \Pr[\pi_1 \wedge \pi_2] = \frac{1}{q_u}$.

If A_2 has an ε advantage to win Game, then B has a probability greater than $\frac{\varepsilon}{q_u}$ to determine whether $X = abP$.

Theorem 2. Our CLVPFCKS scheme is CKCA-CIND secure in standard model if the ECDDDH problem is hard.

Proof: Theorem 2 holds from Lemma 1 and Lemma 2.

Lemma 3. Assuming the adversary A_1 can win Game 3, then an algorithm B can be constructed to solve the ECDDDH problem.

Proof: Suppose that the tuple (P, aP, bP, X) is an example of ECDDDH problem. To determine whether $X = abP$, B will play the part of the challenger.

 doi:10.20944/preprints202401.2177.v1

Set up: B runs the setup (1^k) program to obtain the public parameters $\Omega = \{F_q, E/F_q, G_q, P, G, H_0, H_1, H_2, P_{pub}\}$, where master private key $SK = x$,$PK = P_{pub} = xP$, then randomly selects $r_u, x_C, r_C \in Z_q^*$,and set

$$P_u = aP, R_u = r_uP \; P_C = x_CP, R_C = r_CP,$$
$$e_C = r_C + xH_0(ID_C, R_C, P_C) \bmod q,$$
$$e_u = r_u + xH_0(ID_u, R_u, P_u) \bmod q \; .$$

B sends the public key PK_u and PK_C to A_1, but SK_u and SK_C are unknown to A_1.

Phase 1: Executed the user's public key query before other queries using the identity ID_i. Set up multiple lists to store the queries and answers. All lists are initially empty.

User public key query: B keeps a list L_o of the tuple(ID_i, x_iP, r_iP, r_i) and upon receiving an identity ID_i, performs the following steps.

Case1. $ID_i = ID_i^{\diamondsuit}$, B picks at randomly $x_i^{\diamondsuit} \in Z_q^*$, setting $PK_i^{\diamondsuit} = (x_i^{\diamondsuit}P, bP)$, and adds the tuple $(ID_i^{\diamondsuit}, x_i^{\diamondsuit}P, x_i^{\diamondsuit}, bP, \diamondsuit)$ to the list L_o,Where $\diamondsuit$ represents a null value.

Case2. $ID_i \neq ID_i^{\diamondsuit}$, B picks at randomly $x_i, r_i \in Z_q^*$, setting $PK_i = (x_iP, r_iP)$, and adds the tuple (ID_i, x_iP, r_iP, r_i) to the list L_o.

Replace-Public-Key query: same as that in Lemma 1.

Secret-Value query:same as that in Lemma 1.

Partial-Private-Key query: B establishes a list L_e of tuple (ID_j, e_j) when A_1 asks the partial private key of ID_j. If $ID_j = ID_i^{\diamondsuit}$, B fails and stops. Otherwise, B finds (ID_j, x_jP, r_jP, r_j) in the list L_o, running the Extract-Partial-Private-Key algorithm generating e_j. B output e_j and adds (ID_j, e_j) to the list L_e.

Keyword Ciphertext Query: same as that in Lemma 1.

Keyword Trapdoor Query: same as that in Lemma 1.

Test Query: same as that in Lemma 1.

Challenge: A_1 submits a tuple $(W_0', W_1', \{ID_1{}^*, \cdots, ID_d{}^*\}, \{PK_1{}^*, \cdots, PK_d{}^*\})$, where $W_0' = \{w_{0,1}', w_{0,2}', \cdots, w_{0,l}'\}$ and $W_1' = \{w_{1,1}', w_{1,2}', \cdots, w_{1,l}'\}$ are challenging keywords not requested in the previous trapdoor and ciphertext query. If $ID_i^{\diamondsuit} \notin \{ID_1{}^*, ID_2{}^*, \cdots, ID_d{}^*\}$, B aborts. Otherwise,without losing generality, it is better to set ID_1^* as $ID_i^{\diamondsuit}$. B calculates $l_0{}^* = \sum_{i=1}^{d} H_0(ID_i^*, R_i^*, P_i^*)$. B picks $\xi \in \{0,1\}$ randomly, and computes

$$t = e_u \sum_{i=1}^{d} P_i{}^* + \sum_{i=2}^{d} x_i{}^* aP + \sum_{i=1}^{d} r_i{}^* aP + l_0{}^* xaP + X$$

B selects an elements $\eta_\xi \in_R Z_q^*$ and sets $T_{W'_{\xi,m+1}} = \eta_\xi P$,

$T_{W'_{\xi,j}} = l^{-1}h_2(t)^j \sum_{r=1}^{l} h_1(w'_{\xi,j})^j P + \eta_\xi P_C$, where $0 \leq j \leq m$. Finally, B sent $T_{W_\xi'} = \{T_{w'_\xi,0}$, $T_{w'_\xi,1}, \cdots, T_{w'_\xi,m}, T_{w'_{\xi,m+1}}\}$ to the adversary A_1.

Phase 2: The attacker A_1 can continue to execute various queries, but there is a limitation that the attacker A_1 is not allowed to query the keyword ciphertext or trapdoor of W_0 or W_1.

Guess: A_1 returns ξ'.

Solve CDH problem: If $\xi' = \xi$, B returns 1, otherwise 0. If $X = abP$, then

$$t = e_u \sum_{i=1}^{d} P_i{}^* + \sum_{i=1}^{d} x_i{}^* aP + \sum_{i=2}^{d} r_i{}^* aP + l_0{}^* xaP + X$$
$$= (x_u + e_u)P_0^* + \sum_{i=2}^{d} r_i{}^* aP + x_u l_0^* P_{pub} + abP$$
$$= (x_u + e_u)P_0^* + x_u R_0^* + x_u l_0^* P_{pub}.$$

Therefore, $T_{W_\xi'}$ is a valid ciphertext. Suppose that the advantage of A_1 wins in the above game is ε. So

$$\Pr[\xi' = \xi \,|X = abP] = \tfrac{1}{2} + \varepsilon.$$

If $X \neq abP$, then$T_{W_\xi'}$ is an invalid ciphertext. A_1 has no advantage in distinguishing $\xi = 0$ from $\xi = 1$. Hence

$$\Pr[\xi' = \xi \,|X = abP] = \tfrac{1}{2}.$$

 doi:10.20944/preprints202401.2177.v1

Probability: Let q_o, q_r and q_e be the number of the User public key query, Replace-Public-Key query, and Partial-Private-Key query, respectively. The two events are as follows:

$\pi_1 : A_1$ did not replace of $ID_i^{\diamondsuit}$'s public key $R_i^{\diamondsuit}$ and query the partial-private-key for $ID_i^{\diamondsuit}$.

$\pi_2 : ID_1^* = ID_i^{\diamondsuit}$.

It's not hard to get the following results.

$\Pr[\pi_1] = \frac{q_o - q_r - q_e}{q_o}$,

$\Pr[\pi_2 | \pi_1] = \frac{1}{q_o - q_r - q_e}$,

$\Pr[B\ success] = \Pr[\pi_1 \wedge \pi_2] = \frac{1}{q_u}$.

If A_1 win Game 1 with an advantage of ε, then B has a probability greater than $\frac{\varepsilon}{q_o}$ to determine whether $X = abP$.

Lemma 4 . Assuming the adversary A_2 can win Game 4, then an algorithm B can be constructed to solve the ECDDDH problem.

Proof: Suppose that the tuple (P, aP, bP, X) is an example of ECDDDH problem. To determine whether $X = abP$, B will play the part of the challenger.

Set up : B runs the setup (1^k) program to obtain the public parameters $\Omega = \{F_q, E/F_q, G_q, P, G, H_0, H_1, H_2, P_{pub}\}$, where master private key $SK = x$,$PK = P_{pub} = xP$, then randomly selects $x_u, x_C, r_C \in Z_q^*$, and set

$$P_u = x_u P,\ R_u = aP\ P_C = x_C P,\ R_C = r_C P,$$
$$e_C = r_C + xH_0(ID_C, R_C, P_C) \bmod q,$$
$$e_u = a + xH_0(ID_u, R_u, P_u) \bmod q$$

B sends the public parameters Ω, the public key PK_u, and the public/secret key pair (PK, SK) to A_2, while SK_u are unknown to A_2.

Phase 1: Executed the user's public key query before other queries using the identity ID_i. Set up multiple lists to store the queries and answers. All lists are initially empty.

User public key query: user public key query: B keeps a list L_o of the tuple$(ID_i, x_i P, r_i P, r_i)$ and upon receiving an identity ID_i, performs the following steps.

Case1. $ID_i = ID_i^{\diamondsuit}$, B picks at randomly $r_i^{\diamondsuit} \in Z_q^*$, setting $PK_i^{\diamondsuit} = (bP, r_i^{\diamondsuit} P)$, and adds the tuple $(ID_i^{\diamondsuit}, bP, \diamondsuit, r_i^{\diamondsuit} P, r_i^{\diamondsuit})$ to the list L_o, Where $\diamondsuit$ represents a null value. .

Case2. $ID_i \neq ID_i^{\diamondsuit}$, B picks at randomly $x_i, r_i \in Z_q^*$, setting $PK_i = (x_i P, r_i P)$, and adds the tuple $(ID_i, x_i P, r_i P, r_i)$ to the list L_o.

Replace-Public-Key query: same as that in Lemma 1.

Secret-Value query:B established a list L_s of tuple (ID_j, x_j). When A_2 asks the secret value for ID_j. If $ID_j = ID_i^{\diamondsuit}$, B fails and stops. Otherwise, B finds $(ID_j, x_j P, r_j P, r_j)$ in list L_o, returns x_j.

Partial-Private-Key query: same as that in Lemma 2.

Keyword Ciphertext Query: same as that in Lemma 1.

Keyword Trapdoor Query: same as that in Lemma 1.

Test Query: same as that in Lemma 1.

Challenge: A_2 submits a tuple $(W_0', W_1', \{ID_1{}^*, \cdots, ID_d{}^*\}, \{PK_1{}^*, \cdots, PK_d{}^*\})$, where $W_0' = \{w_{0,1}', w_{0,2}', \cdots, w_{0,l}'\}$ and $W_1' = \{w_{1,1}', w_{1,2}', \cdots, w_{1,l}'\}$ are challenging keywords not requested in the previous trapdoor and ciphertext query. If $ID_i^{\diamondsuit} \notin \{ID_1{}^*, ID_2{}^*, \cdots, ID_d{}^*\}$, B aborts. Otherwise, without losing generality, it is better to set ID_1^* as $ID_i^{\diamondsuit}$. B calculates $l_0{}^* = \sum_{i=1}^{d} H_0(ID_i^*, R_i^*, P_i^*)$. B picks $\xi \in \{0, 1\}$ randomly, and computes

$$t = x_u \sum_{i=1}^{d} P_i{}^* + \sum_{i=2}^{d} x_i{}^* aP + l_u x \sum_{i=1}^{d} P_i{}^* + \sum_{i=1}^{d} r_i{}^* ap + x_u l_0{}^* P_{pub} + X$$

B selects an elements $\eta_\xi \in_R Z_q^*$ and sets $T_{W'_{\xi,m+1}} = \eta_\xi P$,

$T_{W'_{\xi,j}} = l^{-1} h_2(t)^j \sum_{r=1}^{l} h_1(w'_{\xi,j})^j P + \eta_\xi P_C$, where $0 \leq j \leq m$. Finally, B sent $T_{W'_\xi} = \{T_{w'_\xi,0}, T_{w'_\xi,1}, \cdots, T_{w'_\xi,m}, T_{w'_{\xi,m+1}}\}$ to the adversary A_2.

 doi:10.20944/preprints202401.2177.v1

Phase 2: The attacker A_2 can continue to execute various queries, but there is a limitation that the attacker A_2 is not allowed to query the keyword ciphertext or trapdoor of W_0 or W_1.

Guess: A_2 returns ξ'.

Solve CDH problem: If $\xi' = \xi$, B returns 1, otherwise 0. If $X = abP$, then

$$t = x_u \sum_{i=1}^{d} P_i^* + \sum_{i=2}^{d} x_i^* aP + l_u x \sum_{i=1}^{d} P_i^* + \sum_{i=1}^{d} r_i^* aP + x_u l_0^* P_{pub} + X$$

$$= x_u P_0^* + \sum_{i=2}^{d} x_i^* aP + l_u x \sum_{i=1}^{d} P_i^* + x_u R_0^* + x_u l_0^* P_{pub} + abP$$

$$= x_u P_0^* + \sum_{i=1}^{d} x_i^* aP + l_u x \sum_{i=1}^{d} P_i^* + x_u R_0^* + x_u l_0^* P_{pub}$$

$$= x_u P_0^* + (a + l_u x) \sum_{i=1}^{d} P_i^* + x_u R_0^* + x_u l_0^* P_{pub}$$

$$= (x_u + e_u) P_0^* + x_u R_0^* + x_u l_0^* P_{pub}.$$

Therefore, $T_{W'_\xi}$ is a valid ciphertext. Suppose that the advantage of A_2 wins in the above game is ε. So

$$\Pr[\xi' = \xi \,|X = abP] = \tfrac{1}{2} + \varepsilon.$$

If $X \neq abP$, then $T_{W'_\xi}$ is an invalid ciphertext. A_2 has no advantage in distinguishing $\xi = 0$ from $\xi = 1$. Hence

$$\Pr[\xi' = \xi \,|X = abP] = \tfrac{1}{2}.$$

Probability: Let q_o , q_r and q_s be the number of User public key query, Replace-Public-Key query, and Secret-Value query, respectively. The two events are as follows:

π_1 : A_2 did not replace of $ID_i^{\diamondsuit}$'s public key $P_i^{\diamondsuit}$ and query the secret value for $ID_i^{\diamondsuit}$.

π_2 : $ID_1^* = ID_i^{\diamondsuit}$.

It's not hard to get the following results.

$\Pr[\pi_1] = \frac{q_o - q_r - q_e}{q_u}$,

$\Pr[\pi_2 \,|\pi_1] = \frac{1}{q_o - q_r - q_e}$,

$\Pr[B\ success] = \Pr[\pi_1 \wedge \pi_2] = \frac{1}{q_u}$.

If A_2 win Game 4 with an advantage of ε, then B has a probability greater than $\frac{\varepsilon}{q_o}$ to determine whether $X = abP$.

Theorem 3. Our CLVPFCKS scheme is IND-KGA safe in the standard model if the ECDDDH problem is hard. Proof: Theorem 3 holds from Lemma 3 and Lemma 4.

Theorem 4. Under the ECDLP assumption, it is not computationally feasible for the CSP to forge valid proof information through the result verification mechanism.

Proof: The malicious CSP can't forge a valid multi-signature on each returned record and pass the verification. Since it does not have the key of multiple data owners, it is computationally infeasible to forge a valid multi-signature. Therefore, the malicious CSP can only win the next security game by directly generating valid proof information according to the wrong search result C^* instead of wining the next security game by forging multiple signatures. But after the following analysis, this is also impossible.

Assume that the correct ciphertext and its identity is $C = \{c_1, c_2, \cdots, c_n\}$ and $sig = \{sig_1, \cdots, sig_n\}$, where $sig_t = \{Y_{0,t}, V_{0,t}\}$. The malicious CSP may forge wrong proof information (ϕ_1^*, ϕ_2^*) on false search results $C^* = \{c_{k_1}^*, c_{k_2}^*, \cdots, c_{k_s}^*\}$, where

$$\phi_1{}^* = \sum_{\tau=1}^{s} H_1(c_{k_\tau}^*, id_{k_\tau}^*, Y_{0,k_\tau}, P_0),$$

$$\phi_2{}^* = \sum_{\tau=1}^{s} H_1(c_{k_\tau}^*, id_{k_\tau}^*, Y_{0,k_\tau}, Q_0).$$

 doi:10.20944/preprints202401.2177.v1

If the forged proof information (ϕ_1^*, ϕ_2^*) can successfully pass the result verification mechanism, the malicious CSP will win the security game; Otherwise, it will fail. Suppose a malicious CSP wins the game. We then know that

$$\phi_1{}^* = \sum_{\tau=1}^{s} H_1(c_{k_\tau}^*, id_{k_\tau}^*, Y_{0,k_\tau}, P_0),$$

$$\phi_2{}^* = \sum_{\tau=1}^{s} H_1(c_{k_\tau}^*, id_{k_\tau}^*, Y_{0,k_\tau}, Q_0) \tag{4}$$

where $\sigma = \sum_{\tau=1}^{s} V_{0,k_\tau}$. The proof information of correct ciphertext C is (ϕ_1, ϕ_2), where

$$\phi_1 = \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, P_0),$$

$$\phi_2 = \sum_{\tau=1}^{s} H_2(c_{k_\tau}, id_{k_\tau}, Y_{0,k_\tau}, Q_0).$$

The signature of the correct ciphertext can pass the verification mechanism, so we have

$$\sigma P = \sum_{\tau=1}^{s} Y_{o,k_\tau} + \phi_1 P_0 + \phi_2 Q_0 \tag{5}$$

Subtract Formula (4) from Formula (5) to get

$$(\phi_1 - \phi_1{}^*)P_0 = (\phi_2{}^* - \phi_2)Q_0 \tag{6}$$

Because (ϕ_1, ϕ_2) is not equal to (ϕ_1^*, ϕ_2^*), so $\phi_1 \neq \phi_1^*$ or $\phi_2 \neq \phi_2^*$. Set $\Delta\phi_1 = \phi_1 - \phi_1{}^*$, $\Delta\phi_2 = \phi_2 - \phi_2{}^*$, then $\Delta\phi_1 \neq 0$ or $\Delta\phi_2 \neq 0$. Suppose $\Delta\phi_1$ is not zero, then $P_0 = \frac{\Delta\phi_2}{\Delta\phi_1} Q_0$. If the probability of $\Delta\phi_1 = 0$ is $\frac{1}{q}$, then the probability that we can break the ECDLP problem is $1 - \frac{1}{q}$, where q is the length of G_q. This means that if the malicious CSP can pass the verification, we can break the ECDLP problem.

6. Performance analysis

In this section, we compare our scheme with other certificateless or verifiable search schemes in terms of computational complexity, storage overhead and security.

First, let's talk about security. Table 2 compares the security of the current scheme with other schemes. The CKCA-CIND, the IND-KGA, the insider KGA, the FIA, the SCF, and the VER are used to measure the scheme's security. The CKCA-CIND means that the scheme is in differentiability of ciphertext for selected keyword ciphertext attacks, the IND-KGA means the indistinguishability of indiscernibility of trapdoors, the insider-KGA denotes that the scheme is secure against keyword-guessing attacks launched by malicious server, the FIA denotes that the scheme is resistant to file injection attack, the SCF means that the scheme has no secure channel, and the VER indicates that the scheme can prevent malicious CSP from returning wrong search results.

 doi:10.20944/preprints202401.2177.v1

Table 2. COMPARISON OF SAFETY

scheme	CKCA-CIND	IND-KGA	insider-KGA	FIA	SCF	VER
VCKSM[18]	No proof	yes	no	no	yes	yes
VMKS[46]	yes	yes	No proof	No proof	yes	yes
VMKDO[33]	yes	yes	no	no	yes	yes
VCSE[46]	yes	No proof	no	no	yes	yes
CLPAEKS[12]	No proof	No proof	yes	yes	yes	No proof
CL-dPAEKS[11]	No proof	No proof	yes	yes	yes	No proof
our	yes	yes	yes	yes	yes	yes

It is worth noting that no solution can simultaneously achieve result validation, conjunctive keyword search, and certificateless and bilinear pairings, as shown in Table 2. Our proposed solution can simultaneously perform four functions and is proven secure under the standard model.

Next, we compare the computational complexity. To compare the computational complexity, we use the operation time of He et al.'s scheme[47] as the benchmark. He et al. tested the time required for the relevant operations in the experimental environment of Samsung Galaxy S5 based on the Android 4.4.2 operating system, quad-core 2.45G processor, and 2G byte memory. Table 4 shows the exact running time and symbols of the various operations. The mapping $e : G_1 \times G_1 \rightarrow G_2$ is a bilinear pair where G_1 is an additive group of singular elliptic curves of order p defined on a finite field F_q. The lengths of p and q are 512 bits and 160 bits, respectively. G is an additive group of non-sigular elliptic curve of order q defined on the prime finite field F_q. The length of p and q is 160.

Table 3. Symbol definition

symbols	Definition
T_{bp}	Running time required for a bilinear pairing operation, $T_{bp} \approx 32.713ms$
T_{htp}	Running time required for a hash-to-operation,$T_{htp} \approx 33.582ms$
T_{sm}	Running time required for a scalar multiplication operation in G_1, $T_{sm} \approx 13.405ms$
$T_{\exp}$	Running time required for an exponentiation operation in G_2, $T_{\exp} \approx 2.249ms$
$T'sm$	Running time required for a scalar multiplication operation in G, $T_{sm} \approx 3.335ms$
u	Number of data users
d	Number of data owners
m	Number of keywords in ciphertext
n	Number of ciphertext
s	Number of search keywords

Table 4 shows the comparison results of the running time of the verifiable conjunctive keyword search encryption scheme. To be more intuitive and specific, we taking n=1000, m=100, d=100, u=50, s=50. Table 5 shows the calculation time of the three schemes. Except for the search algorithm, our scheme takes less time than the other two schemes, especially for the verification algorithm. Therefore, our scheme is the most efficient.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 January 2024 doi:10.20944/preprints202401.2177.v1

Table 4. Comparison of running time of various schemes

scheme	keyGen	Enc	Trap	search	verify
VCKSM[18]	$(u+d+1)T_{sm}$	$[2nd+n(m+2)]T_{sm}$ $+2nT_{\exp}+2nT_{bp}$ $+nT_{htp}$	$(m+3)T_{sm}$	$(n+3)T_{sm}$ $+[n(m+1)+1]T_{bp}$	$(2s+1)T_{sm}$ $+sT_{htp}+2T_{bp}$
VMKS[46]	$(4u+4d)T_{sm}$	$nT_{htp}+$ $(6n+mn)T_{sm}$	$(l+5)T_{sm}$	$4T_{bp}$	$(s+2)T_{sm}$ $+sT_{htp}+3T_{bp}$
our	$(2d+5)T'$sm	$[nd+n(m+4)]T'sm$	$(m+6)T'sm$	$2+n(m+1)]T'sm$	$3T'sm$

Table 5. Comparison of running time of various schemes

scheme	keyGen(ms)	Enc(ms)	Trap(ms)	search(ms)	verify(ms)	Total
VCKSM[18]	2024.155	4151817	1380.715	3305426.428	3098.481	7463746.779
VMKS[46]	8043	1382126	737.275	130.852	2474.349	1393511.476
our	683.675	680340	353.51	336838.335	6.67	1018222.19

Let $|G_1|$, $|G_2|$, $|G|$ and $\left|Z^*_q\right|$ denote the size of an element in G_1, G_2, G and Z^*_q, respectively. For more intuitive comparison of communication costs. Table 6 uses n=100, m=10 and s=10. As shown in Figure 3, our solution has the lowest storage cost.

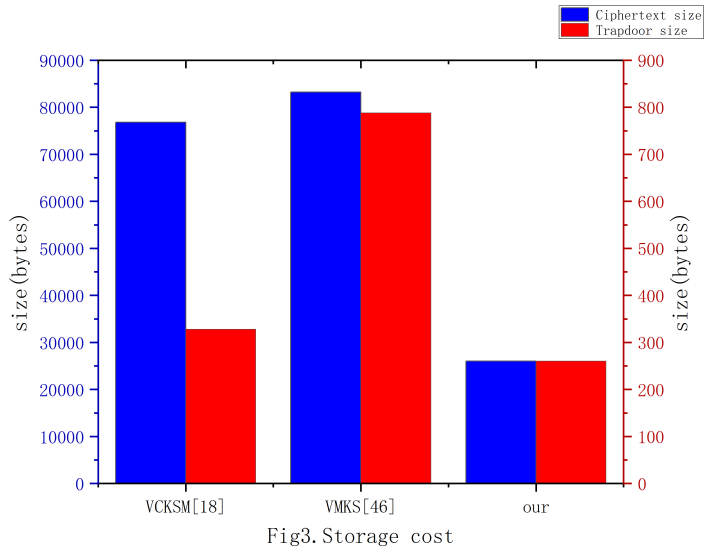

Figure 3. System model of CLVPFCKS

Table 6. Communication Comparison Of Various Schemes

scheme	Ciphertext size(bytes)	Trapdoor size(bytes)
VCKSM[18]	$n\,(m+2)\,\lvert G_2\rvert = 1200\times 512/8 = 76800$	$\lvert G_1\rvert+\lvert G_2\rvert+s\left\lvert Z^*_q\right\rvert = (512+512+10\times 160)/8 = 328$
VMKS[46]	$n\,(\lvert G_2\rvert+(m+2)\,\lvert G_1\rvert)$ $= 100\times(512+12\times 512)/8$ $= 83200$	$(m+2)\,\lvert G_1\rvert+\left\lvert Z^*_q\right\rvert = (12\times 512+160)/8 = 788$
our	$n\,(m+3)\,\lvert G\rvert = (1300\times 160)/8 = 26000$	$(m+2)\,\lvert G\rvert = 12\times 160/8 = 240$

As stated in the summary, our scheme is more efficient than others in computing and communication costs. And it is also the best in security.

7. Conclusion

Searchable encryption is an essential technology for medical Internet of Things. We have constructed a certificateless verifiable bilinear pair-free conjunctive keyword search encryption scheme (CLVPFCKS) for the Internet of Medical. The performance analysis shows that the CLVPFCKS scheme proposed in this paper performs better than the verifiable conjunctive keyword searchable encryption scheme using bilinear pairing. We prove the new system can resist keyword guessing attacks, choose keyword attacks (CKA), and file injection attacks under the standard model.

References

1. L. AtzoriA. Iera and G. Morabito, "The Internet of Things: A survey," Comput. Netw., vol. 54, no. 15, pp. 2787-2805, 2010, doi: 10.1016/j.comnet.2010.05.010.
2. A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, "Internet of Things for Smart Cities," Ieee Internet of Things Journal, vol. 1, no. 1, pp. 22-32, 2014, doi: 10.1109/JIOT.2014.2306328.
3. P. Bellavista, G. Cardone, A. Corradi and L. Foschini, "Convergence of MANET and WSN in IoT Urban Scenarios," Ieee Sens. J., vol. 13, no. 10, pp. 3558-3567, 2013, doi: 10.1109/JSEN.2013.2272099.
4. V. Jagadeeswari, V. Subramaniyaswamy, R. Logesh and V. Vijayakumar, "A study on medical Internet of Things and Big Data in personalized healthcare system," Health Information Science and Systems, vol. 6, no. 1, 2018, doi: 10.1007/s13755-018-0049-x.
5. D.J. He, R. Ye, S. Chan, M. Guizani and Y.P. Xu, "Privacy in the Internet of Things for Smart Healthcare," Ieee Commun. Mag., vol. 56, no. 4, pp. 38-44, 2018, doi: 10.1109/MCOM.2018.1700809.
6. Y.Y. ChenJ.C. Lu and J.K. Jan, "A Secure EHR System Based on Hybrid Clouds," J. Med. Syst., vol. 36, no. 5, pp. 3375-3384, 2012, doi: 10.1007/s10916-012-9830-6.
7. D.X. SongD. Wagner and A. Perrig, Practical Techniques for Searches on Encrypted Data, Proc. IEEE Symposium on Security & Privacy, 2002, pp. 44-55.
8. D. Boneh, G.D. Crescenzo, R. Ostrovsky and G. Persiano, "Public Key Encryption with Keyword Search," Eurocrypt 2004, pp. 506-522, 2004, doi: 10.1007/978-3-540-24676-3_30.
9. J. BaekR. Safiavinaini and W. Susilo, Public Key Encryption with Keyword Search Revisited, Proc. International Conference on Computational Science and Its Applications (ICCSA 2008), 2008, pp. 1249-1259.
10. H.S. Rhee, J.H. Park, W. Susilo and D.H. Lee, Improved searchable public key encryption with designated tester, Proc. International Symposium on Information, 2009, pp. 376.
11. L. Wu, Y. Zhang, M. Ma, N. Kumar and D. He, "Certificateless searchable public key authenticated encryption with designated tester for cloud-assisted medical Internet of Things," Ann. Telecommun., vol. 74, no. 7-8, pp. 423-434, 2019, doi: 10.1007/s12243-018-00701-7.
12. D. He, M. Ma, S. Zeadall, N. Kumar and K. Liang, "Certificateless Public Key Authenticated Encryption with Keyword Search for Industrial Internet of Things," Ieee T. Ind. Inform., pp. 1, 2017, doi: 10.1109/TII.2017.2771382.
13. M.M. A, D.H.B. C, M.K.K. D and J.C. A, "Certificateless searchable public key encryption scheme for mobile healthcare system," Comput. Electr. Eng., vol. 65, pp. 413-424, 2018, doi: 10.1016/j.compeleceng.2017.05.014.
14. M. Ma, D. He, S. Fan and D. Feng, "Certificateless searchable public key encryption scheme secure against keyword guessing attacks for smart healthcare," Journal of Information Security and Applications, vol. 50, 2020, doi: 10.1016/j.jisa.2019.102429.
15. Y. Lu and J. Li, "Constructing pairing-free certificateless public key encryption with keyword search," Frontiers of Information Technology & Electronic Engineering, vol. 20, no. 8, pp. 1049-1060, 2019, doi: DOI10.1631/FITEE.1700534.
16. P. GolleJ. Stadon and B. Waters, Secure conjunctive keyword search over encrypted data,3089, M. Jakobsson, et al., eds., 2004, pp. 31-45.
17. Y.H. Hwang and P.J. Lee, Public key encryption with conjunctive keyword search and its extension to a multi-user system, Book Public key encryption with conjunctive keyword search and its extension to a multi-user system, Series Public key encryption with conjunctive keyword search and its extension to a multi-user system 4575,ed., Editor ed., 2007, pp. 2.

18. Y.B. Miao, et al., "VCKSM: Verifiable conjunctive keyword search over mobile e-health cloud in shared multi-owner settings," Pervasive Mob. Comput., vol. 40, pp. 205-219, 2017, doi: 10.1016/j.pmcj.2017.06.016.
19. Y. Yang and M.D. Ma, "Conjunctive Keyword Search With Designated Tester and Timing Enabled Proxy Re-Encryption Function for E-Health Clouds," Ieee T. Inf. Foren. Sec., vol. 11, no. 4, pp. 746-759, 2016, doi: 10.1109/TIFS.2015.2509912.
20. S.H. Heng and K. Kurosawa, k-resilient identity-based encryption in the standard model,2964, T. Okamoto, ed., 2004, pp. 67-80.
21. D. Khader, Public key encryption with keyword search based on K-resilient IBE,3982, M. Gavrilova, et al., eds., 2006, pp. 298-308.
22. H.M. YangC.X. Xu and H.T. Zhao, An Efficient Public Key Encryption with Keyword Scheme Not Using Pairing, Proc. First International Conference on Instrumentation, 2011.
23. T.F. Vallent and H. Kim, A Pairing-Free Public Key Encryption with Keyword Searching for Cloud Storage Services, Book A Pairing-Free Public Key Encryption with Keyword Searching for Cloud Storage Services, Series A Pairing-Free Public Key Encryption with Keyword Searching for Cloud Storage Services 135,ed., Editor ed., 2014, pp. 70.
24. N.B. YangS.M. Xu and Z. Quan, "An Efficient Public Key Searchable Encryption Scheme for Mobile Smart Terminal," Ieee Access, vol. 8, pp. 77940-77950, 2020, doi: 10.1109/ACCESS.2020.2989628.
25. M. Ma, M. Luo, S. Fan and D. Feng, "An Efficient Pairing-Free Certificateless Searchable Public Key Encryption for Cloud-Based IIoT," Wirel. Commun. Mob. Com., vol. 2020, 2020, doi: 10.1155/2020/8850520.
26. M.R. Senouci, I. Benkhaddra, A. Senouci and F.G. Li, "A provably secure free-pairing certificateless searchable encryption scheme," Telecommun. Syst., vol. 80, no. 3, pp. 383-395, 2022, doi: 10.1007/s11235-022-00912-3.
27. Z.Y. Hu, L.Z. Deng, Y.Y. Wu, H.Y. Shi and Y. Gao, "Secure and Efficient Certificateless Searchable Authenticated Encryption Scheme Without Random Oracle for Industrial Internet of Things," Ieee Syst. J., vol. 17, no. 1, pp. 1304-1315, 2023, doi: 10.1109/JSYST.2022.3197174.
28. J.W. Byun, H.S. Rhee, H.A. Park and D.H. Lee, Off-line keyword guessing attacks on recent keyword search schemes over encrypted data,4165, W. Jonker and M. Petkovic, eds., 2006, pp. 75-83.
29. Y. LuG. Wang and J.G. Li, "Keyword guessing attacks on a public key encryption with keyword search scheme without random oracle and its improvement," Inform. Sciences, vol. 479, pp. 270-276, 2019, doi: 10.1016/j.ins.2018.12.004.
30. M.S. HwangS.T. Hsu and C.C. Lee, "A New Public Key Encryption with Conjunctive Field Keyword Search Scheme," Inf. Technol. Control, vol. 43, no. 3, pp. 277-288, 2014, doi: 10.5755/j01.itc.43.3.6429.
31. Q. Chai and G. Gong, Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers, Proc. IEEE International Conference on Communications, 2012.
32. W. Sun, X. Liu, W. Lou, Y.T. Hou and H. Li, Catch You If You Lie to Me: Efficient Verifiable Conjunctive Keyword Search over Large Dynamic Encrypted Cloud Data, Proc. 34th IEEE Conference on Computer Communications (INFOCOM), 2015.
33. Y. Miao, J. Ma, X. Liu, Z. Liu and F. Wei, "VMKDO: Verifiable multi-keyword search over encrypted cloud data for dynamic data-owner," Peer Peer Netw. Appl., vol. 11, no. 2, pp. 287-297, 2018, doi: 10.1007/s12083-016-0487-7.
34. Y. Miao, J. Weng, X. Liu, K. Raymond Choo, Z. Liu and H. Li, "Enabling verifiable multiple keywords search over encrypted cloud data," Inform. Sciences, vol. 465, pp. 21-37, 2018, doi: 10.1016/j.ins.2018.06.066.
35. L. Fang, W. Susilo, C. Ge and J. Wang, "Public key encryption with keyword search secure against keyword guessing attacks without random oracle," Inform. Sciences, vol. 238, pp. 221-241, 2013, doi: 10.1016/j.ins.2013.03.008.
36. Z.Y. Shao and B. Yang, "On security against the server in designated tester public key encryption with keyword search," Inform. Process. Lett., vol. 115, no. 12, pp. 957-961, 2015, doi: 10.1016/j.ipl.2015.07.006.
37. C. Hu and P. Liu, "An Enhanced Searchable Public Key Encryption Scheme with a Designated Tester and Its Extensions," Journal of Computers, vol. 7, no. 3, 2012, doi: 10.4304/jcp.7.3.716-723.
38. H.S. Rhee, J.H. Park, W. Susilo and D.H. Lee, "Trapdoor security in a searchable public-key encryption scheme with a designated tester," J. Syst. Software, vol. 83, no. 5, pp. 763-771, 2010, doi: 10.1016/j.jss.2009.11.726.
39. Y. LuG. Wang and J. Li, "On Security of a Secure Channel Free Public Key Encryption with Conjunctive Field Keyword Search Scheme," Inf. Technol. Control, vol. 47, no. 1, pp. 56-62, 2018, doi: 10.5755/j01.itc.47.1.16137.

40. I.R. Jeong, J.O. Kwon, D. Hong and D.H. Lee, "Constructing PEKS schemes secure against keyword guessing attacks is possible?" Computer Communications, vol. 32, no. 2, pp. 394-396, 2009, doi: 0.1016/j.comcom.2008.11.018.
41. P. Xu, H. Jin, Q. Wu and W. Wang, "Public-Key Encryption with Fuzzy Keyword Search: A Provably Secure Scheme under Keyword Guessing Attack," Ieee T. Comput., vol. 62, no. 11, pp. 2266-2277, 2013, doi: 10.1109/TC.2012.215.
42. X.J. Lin, L. Sun, H.P. Qu and D.X. Liu, "On the Security of Secure Server-Designation Public Key Encryption with Keyword Search," Comput. J., vol. 61, no. 12, pp. 1791-1793, 2018, doi: 10.1093/comjnl/bxy073.
43. Q. Huang and H.B. Li, "An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," Inform. Sciences, vol. 403, pp. 1-14, 2017, doi: 10.1016/j.ins.2017.03.038.
44. L. Wu, B. Chen, S. Zeadally and D. He, "An efficient and secure searchable public key encryption scheme with privacy protection for cloud storage," Soft Comput., vol. 22, no. 23, pp. 7685-7696, 2018, doi: 10.1007/s00500-018-3224-8.
45. S.S. Al-Riyami and K.G. Paterson, Certificateless public key cryptography,2894, C. S. Laih, ed., 2003, pp. 452-473.
46. Y.M.J.W. Miao, "VCSE: Verifiable conjunctive keywords search over encrypted data without secure-channel," Peer-to-Peer Netw, pp. 995-1007, 2017.
47. D.B. He, H.Q. Wang, L.N. Wang, J. Shen and X.Z. Yang, "Efficient certificateless anonymous multi-receiver encryption scheme for mobile devices," Soft Comput., vol. 21, no. 22, pp. 6801-6810, 2017, doi: 10.1007/s00500-016-2231-x.