Singapore Management University

Institutional Knowledge at Singapore Management University

Research Collection School Of Computing and Information Systems | School of Computing and Information Systems

5-2024

Attribute-hiding fuzzy encryption for privacy-preserving data evaluation

Zhenhua CHEN
Xi'an University of Electronic Science and Technology

Luqi HUANG
University of Wollongong

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Willy SUSILO
University of Wollongong

Xingbing FU
Hangzhou Dianzi University

See next page for additional authors

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

 Part of the Information Security Commons

Citation

CHEN, Zhenhua; HUANG, Luqi; YANG, Guomin; SUSILO, Willy; FU, Xingbing; and JIA, Xingxing. Attribute-hiding fuzzy encryption for privacy-preserving data evaluation. (2024). *IEEE Transactions on Services Computing*. 17, (3), 789-803.
Available at: https://ink.library.smu.edu.sg/sis_research/8694

This Journal Article is brought to you for free and open access by the School of Computing and Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Computing and Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email cherylds@smu.edu.sg.

Author

Zhenhua CHEN, Luqi HUANG, Guomin YANG, Willy SUSILO, Xingbing FU, and Xingxing JIA

This journal article is available at Institutional Knowledge at Singapore Management University: https://ink.library.smu.edu.sg/sis_research/8694

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

Attribute-hiding fuzzy encryption for privacy-preserving data evaluation

Zhenhua Chen, Luqi Huang, Guomin Yang, Willy Susilo, Xingbing Fu, Xingxing Jia

***Abstract*—Privacy-preserving data evaluation is one of the prominent research topics in the big data era. In many data evaluation applications that involve sensitive information, such as the medical records of patients in a medical system, protecting data privacy during the data evaluation process has become an essential requirement. Aiming at solving this problem, numerous fuzzy encryption systems for different similarity metrics have been proposed in literature. Unfortunately, the existing fuzzy encryption systems either fail to achieve attribute-hiding or achieve it, but are impractical. In this paper, we propose a new fuzzy encryption scheme for privacy-preserving data evaluation based on overlap distance, which can work in an integer domain while achieving attribute-hiding. In particular, we develop a novel approach to enable an accurate overlap distance to be fast calculated. This technique makes the number of pairing operations during decryption stage negative correlation with the size of the threshold, which is pretty practical for some applications especially with a large threshold. Additionally, we provide a formal security analysis of the proposed scheme, followed by a comprehensive experimental. Also we show that our scheme can be well applied to some scenarios, such as fuzzy keyword searchable encryption and attribute-hiding closest substring encryption.**

***Index Terms*—Fuzzy encryption, predicate encryption, attribute-hiding, data evaluation, overlap distance**

I. Introduction

The rapid development of information and communications technology has greatly affected many aspects of our daily life. A huge amount of data are being generated/collected daily by end users and devices (e.g., IoT sensors) and these data form a valuable asset in the information age. Data analysis techniques have been employed by both public and private sectors to exploit the potential and valuable information from these massive data. To make the data analysis more accurate and efficient, data evaluation refers to as a process of determining the merit, worth and significance of data by using certain evaluation criteria governed by a set of measurements. It can assist the data analyzer to select available data for analysis. However, some data are sensitive in nature (e.g., medical data), and therefore it is important to ensure that these sensitive data, often including their attributes (or metadata), are well-protected during the evaluation process. Hence, privacy-preserving data evaluation has become a prominent research topic in recent years.

One typical application of privacy-preserving data evaluation is privacy-preserving medical data analysis and clinical diagnosis in an electronic health record system (see Fig. 1), where patients (or elderly people in Aged Care Center) with chronic diseases or severe illnesses are equipped with implanted or on-body medical sensors to monitor various kinds of physiology symptoms and collect the corresponding data, e.g., smart heart rate monitor can detect patients' heart beats per minute to avoid heart rate overboard, and blood sugar-sensing wearable device can track the patients' glucose levels to ensure that the right amount of insulin is released at the right time, etc. The collected data will provide valuable information to doctors for analyzing diseases and performing diagnosis. However, medical data, including both patients' records and symptoms, are very sensitive personal information. It is crucial to prevent leakage or abuse of patients' personal data when the data are stored in a medical database and further used in a data evaluation process. Therefore, it has become an important issue how to ensure patients' privacy in an electronic health record system. For protecting the privacy of their data, patients often encrypt their data (health records and symptoms) prior to sending them to a storage server in hospital. At the later point, the doctor will check whether a patient's symptoms satisfy a certain criteria of the disease based on an appropriate similarity metric without learning anything else about patients' health records and symptoms, and if and only if the criteria are met, the doctor is able to conceal the patients' health records and perform more investigations.

This work was supported in part by the National Natural Science Foundation of China under Grant 61872289 and Grant 62172266, in part by the Guangxi Key Laboratory of Trusted Software under Grant KX202308, and in part by the Henan Key Laboratory of Network Cryptography Technology LNCT2020-A07. *Corresponding author:Luqi Huang*

Z.Chen is with the College of Computer Science and Technology, Xi'an University of Science and Technology, Xi'an 710054, China. Z.Chen is also with the Guangxi Key Laboratory of Cryptography and Information Security, Guilin University of Electronic Technology, Guilin 541004, China.
W.Susilo and L.Huang are with the Institute of Cybersecurity and Cryptology, School of Computing and Information Technology, University of Wollongong, Wollongong, Australia.
Guomin Yang is with the School of computing and information systems, Singapore Management University, Singapore.
X.Fu is with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, PR China.
X.Jia is with the School of Mathematics and Statistics, Lanzhou University, Lanzhou 730000, China.

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

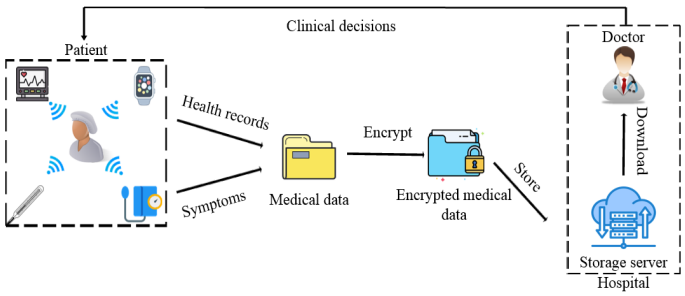


Fig. 1. Privacy-preserving clinical data analysis

Another similar example of privacy-preserving data evaluation is about privacy-preserving food health rating (see Fig. 2). A food industry develops a new product and makes a request on its health rate (e.g., 5/4/... stars) to a Food Inspection Agency. The Food Inspection Agency can determine its health rate based on the evaluation on the ingredients of new product. However, the recipe of new product is often a commercial secret of the food industry, who would not want its product formula to be known by anyone else except the target evaluator of Food Inspection Agency. That is, only and if only a certain similarity metric is met between the recipe of the new product and a criteria provided by the agency. The agency is able to conceal the recipe of a new product and returns complete decisions on its health rate to the food industry.

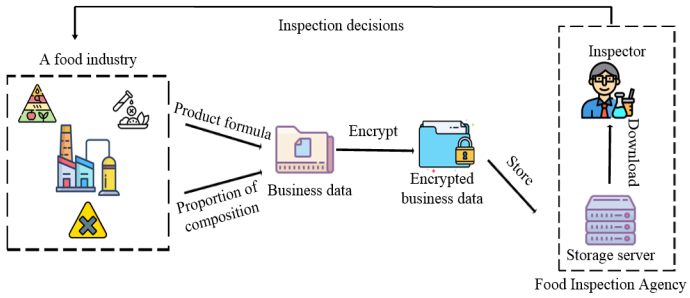


Fig. 2. Privacy-preserving food inspection

From the above application scenarios, we observe that to manage the privacy-preserving data evaluation, the "overlap distance" is often taken as an important role, to measure the similarity between the data attributes and an evaluation criteria but in a secure manner rather than in a clear text. To apply privacy-preserving data evaluation based on an overlap distance in the above applications, the crux of a privacy-preserving data evaluation system lies in three properties: *fuzziness*, *efficiency*, and *attribute-hiding*. Nevertheless, it is a challenging task to achieve the above properties simultaneously in one system. Fuzzy encryption schemes have been introduced in the literature for the purpose of fuzziness based on different similarity metrics such as Hamming distance, Edit distance, Overlap distance, Mahalanobis distance and Set difference. Unfortunately, among all the existing schemes, most of them just provide message privacy but no protection on attributes' privacy [1], [2], [3], [4], [5]. On the other hand, some fuzzy schemes with attribute-hiding can only work in a binary domain [6], [7]. Therefore, in this paper, we attempt to design a new fuzzy encryption scheme under the public-key setting which works in an integer domain and meanwhile achieves attribute-hiding. Particularly, this new scheme is able to measure the overlap distance between a target integer string and an attribute integer string associated with a message in a high-level secure manner - attribute-hiding but with a higher efficiency.

A. Our contributions

To apply privacy-preserving data evaluation based on an overlap distance, in this paper we present a new construction of attribute-hiding fuzzy encryption (AHFE) for an overlap distance in an integer domain. The crux of a privacy-preserving data evaluation system lies in three properties: *fuzziness*, *attribute-hiding* and *efficiency*. In order to achieve these goals, we first show how to conduct fuzziness based on the overlap distance between two strings in an integer domain via some ingenious transformation, and then present a concrete construction of fuzzy encryption with an attribute-hiding property by leveraging a modified version of inner product encryption (IPE) [8] along with a novel calculation mechanism to improve efficiency. Specifically, the main contributions of our work are summarized as follows:

- We develop some new techniques for constructing attribute-hiding fuzzy encryption in an integer domain rather than a binary domain. Although the existing fuzzy identity based encryption (IBE) scheme [7], a version closest to our system has already achieved attribute-hiding in a binary domain and a naive approach is to realize our system by leveraging their idea, the main obstacle is that if we transform an integer string into a binary one and leverage the scheme in [7] to realize fuzziness straightforwardly, it will raise a misjudgment described in related work later. As a result, we develop a new encoding to conduct the overlap distance between two strings in an integer domain. Further, we borrow the technique of the dual paring vector space in IPE [8] after slightly modified to realize attribute-hiding property. Similarly, a same obstacle will be arise if we employ the scheme in [6] to cope with our issue since they can only conduct a binary domain instead of an integer one. All these will be detailed later.
- For a higher efficiency, we modify the technique of privacy-preserving mapping in [9] and derive a new calculation mechanism for a fast decryption. This technique enables our construction to fast decrypt the ciphertext without a loop to try all possible decryption keys one by one and thus enjoys the benefit of the number of pairing operations during decryption stage negative correlation with the size of the threshold. Accordingly, it is pretty practical for some applications especially with a large threshold. As opposed to ours, the existing fuzzy encryption scheme [7] is impractical since the computational

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

complexity of pairing operations during decryption stage is exponential with threshold.
- We prove the security of our AHFE scheme under a well-defined security model. We also implement our AHFE scheme to demonstrate its efficiency in real experiments. In addition, we will discuss some interesting applications of our scheme, including fuzzy keyword searchable encryption and attribute-hiding closest substring encryption.

B. Related work

In 2005, Sahai and Waters [10] first addressed the notion of fuzziness in the field of identity-based cryptography, a new encryption mechanism initiated by Shamir [11] in 1984, where a private key of identity can decrypt a ciphertext encrypted with a slightly different identity. Later, Sahai and Waters also introduced the notion of attribute-based encryption (ABE) in literature [10], an advanced IBE version with more expresiveness. In an IBE system, private keys were issued by a fully trusted key generation center that was introduced to verify all identities when issuing users' private keys.

Therefore in the IBE system, the key generation center is capable of generating private keys for all users and it has the ability to access the messages of each user. The problem is named as key escrow. How to solve the key escrow problem existed in IBE system is certificateless public-key encryption (CL-PKE) [12], later than IBE invented by Al-Riyami and Paterson. Roughly speaking, their idea was to combine the functionality of public key encryption with that of identity based encryption. Also, Lewko and Waters [13] provided an alternative solution, named decentralized ABE. However, our effort has focused on the IBE system, which was inherent with a fully trusted third party.

In 2015, Guo et al. [1] introduced distance-based encryption equipped with Mahalanobis distance in measurement, if and only if the distance between two identity vectors is no greater than a threshold the ciphertext will be decrypted successfully. Similarly, Guo et al. [2] in 2016 also provided the closest substring encryption by measuring the overlap distance between substrings from two main strings, respectively. In the same year, Phuong et al. [3] came up with an edit-distance based encryption which is to measure the similarity between two alphabet strings. Recently, [4], [5] devoted themselves to develop a new public key encryption with fuzzy matching functionality which can test whether two encrypted messages is equal or not based on a certain similarity metric. Specifically, the former is based on whether the edit distance between two encrypted messages is less than a threshold, and the latter is based on Vi 'ete's formula for testing two messages of ciphertexts equal or not without the care of some designated wildcard bits. However, the above fuzzy encryption schemes all can only achieve payload-hiding, a weaker security than attribute-hiding in the sense that it can only guarantee the traditional message privacy without protection on attribute privacy.

For the existing fuzzy encryption schemes with attribute-hiding [14], [6], [7], Boneh and Waters [14] constructed a hidden vector encryption which can support a certain fuzziness in the sense that the encryption system only asks two components in two vectors are identical in the rest positions without the care of the wildcard positions. Unlike ours, they fail to support the fuzziness based on a threshold. Dodis et al. [6] proposed a fuzzy extractor that allows exact recovery of a secret string when a given another string is close enough to the secret string until the distance between them is less than a threshold. However, in their mechanism the two strings are only restricted to working in a binary domain instead of an integer domain. Zhang et al. [7] put forward an anonymous fuzzy IBE for similarity search by testing whether the Hamming distance between two identity vectors in a binary domain is no greater than a threshold. However, it is noticed that this scheme will raise a misjudgment when we leverage it to cope with an integer domain like ours. The main obstacle is that if we transform an integer string into a binary string straightforwardly, the hamming distance between two binary vectors which is greater than a threshold will also be searched successfully but not really. Also, the same obstacle will arise if we employ the scheme [6] to cope with our issue.

As mentioned above, the previous works have pursued the following directions falling into either payload-hiding fuzzy encryption [1], [2], [3], [4], [5], or fuzzy schemes with attribute-hiding but failing to support threshold [14] or restricted in a binary domain [6], [7]. Regarding how to design a fuzzy encryption under IBE mechanism which can work in an integer domain but with a higher security – attribute-hiding, little attention has been paid to it. To address this issue, in this work we put forward a new attribute-hiding fuzzy encryption scheme in an integer domain.

C. Organization

The paper starts with the problem formulation in Section 2. Section 3 presents the system and security definitions. Section 4 presents preliminaries and Section 5 provides a concrete construction. Section 6 proves the security of the scheme. The implementation of AHFE scheme and comparison analysis are illustrated in Section 7. Section 8 offers two application examples of our schemes to fuzzy keyword search encryption and attribute-hiding closest substring encryption. Finally, we conclude our work in Section 9.

II. Problem formulation

A. System components

From Fig. 3, our evaluation system involves three parties: a key generation center (or system administrator) who is responsible for the system setup and key extraction for an evaluator; a user (i.e., the patient in Fig. 1 and the food industry in Fig. 2) who encrypt their individual data and would like to collaborate a privacy-preserving evaluation on their encrypted data by an authorized evaluator; an evaluator (i.e.,

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

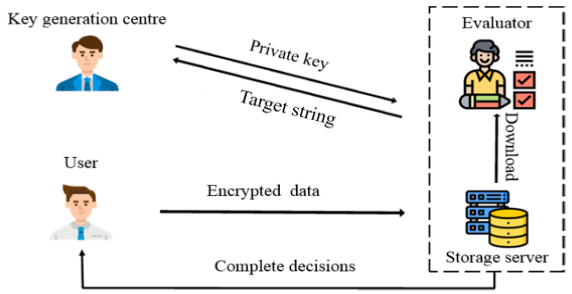


Fig. 3. System model

the doctor in Fig. 1 and the inspector in Fig. 2) who is allowed to perform an evaluation based on these received data and further make a complete decision for users.

1. Key generation center: A key generation center is responsible for generating system parameters and master public/secret key pair. Additionally, it computes a private key based on a received target string along with a threshold and sends the private key to an evaluator.
2. User: A user is responsible for transmitting encrypted data based on a message along with its attribute string to an evaluator.
3. Evaluator: An evaluator is allowed to investigate data transferred from a user if and only if the overlap distance between the target string and the attribute string associated with data is no less than the threshold. If so, it is able to download the user's encrypted data from a storage server and learn the message to make a further corresponding evaluation. Finally, the evaluator returns a complete decision to the user based on the evaluation result.

B. Threat model

In this paper, the key generation center is fully trusted, which will not collude with any entity to peep data contents. The user is assumed to be honest. In other words, it honestly generates ciphertexts forward to an evaluator. The evaluator is supposed to be honest-but-curious. It honestly responds to the request and executes the evaluation as it is defined and returns the actual results to the user but may be "curious" about the user's sensitive data and try to learn more information motivated by financial interests or economic incentives.

C. System requirements

A secure evaluation system should satisfy the following features and security requirements.

1. Availability: The authorized evaluator is allowed to learn the message from the user when the overlap distance between a user's attribute string and a target string is no less than a threshold.
2. Efficiency: The system is able to process data evaluation efficiently.
3. Attribute-hiding (Anonymous): The evaluator cannot acquire the user's attributes during the evaluation process.
4. Message privacy: The evaluator is not be able to learn the user's message if the overlap distance is less than the threshold.

III. System and Security Definitions

A. Definition of AHFE

Let $U = \{1, \ldots, l\}$ be an attribute universe and l be a positive integer. S and S' are two different strings of length n and each element of both is from universe U. Our AHFE scheme consists of four algorithms as follows:

- **Setup**$(1^\lambda, n, l)$: The key generation center taking as input a security parameter λ, the length of strings n and the size of universe l, the setup algorithm outputs a master key pair (mpk, msk).
- **KeyGen**(mpk, msk, S', t): The key generation center taking as input the master public key mpk, the master secret key msk, a string S' and a threshold t, the keygen algorithm outputs a private key $sk_{S'}$.
- **Enc**(mpk, m, S): A user taking as input the master public key mpk, a message m and a string S, the encryption algorithm outputs a ciphertext ct_S.
- **Dec**$(mpk, sk_{S'}, ct_S, n, t)$: An evaluator taking as input the master public key mpk, the ciphertext ct_S, the private key $sk_{S'}$, the length of strings n, and the threshold t, the decryption algorithm outputs the message m if the overlap distance between S and S' is no less than the threshold t.

Correctness: For all S' and $S \in \{1, \ldots, l\}^n$, all $m \in \mathbb{G}_T$, let $(mpk, msk) \xleftarrow{R}$ **Setup**$(1^\lambda, n, l)$, $sk_{S'} \xleftarrow{R}$ **KeyGen**(mpk, msk, S', t) and $ct_S \xleftarrow{R}$ **Enc**(mpk, m, S). If we have $OverlapDist(S', S) \geq t$, $m \xleftarrow{R}$ **Dec**$(mpk, sk_{S'}, ct_S, n, t)$, otherwise $\Pr[\perp \longleftarrow$ **Dec**$(mpk, sk_{S'}, ct_S, n, t)] > 1 - \epsilon(\lambda)$ where $\epsilon(\lambda)$ is a negligible function.

B. Security model

Okamoto et al. [8] proposed an efficient IPE on dual pairing vector spaces (DPVS) with selectively fully attribute-hiding against chosen plaintext attacks. Similarly, our security is defined with indistinguishability under selective-string, chosen-plaintext attacks (IND-sS-CPA), which also achieves selectively fully attribute-hiding after slightly modifying Okamoto et al's definition [8] in matching and non-matching queries. Specifically, the matching queries are changed from an inner product of two vectors being zero to an overlap distance of two strings greater than or equal to a threshold. Meanwhile,

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

non-matching queries is changed to an overlap distance of two strings is less than a threshold. Moreover, there is an extra restriction on matching queries in our security model in that, we require $OverlapDist(S', S^{(0)}) = OverlapDist(S', S^{(1)})$ where $S^{(0)}$ and $S^{(1)}$ are two challenge strings chosen by the adversary ahead of attack, and S' is any string queried by the adversary. The security of AHFE is captured by the following game interaction between an attacker $\mathcal{A}$ and a challenger C.

1. $\mathcal{A}$ outputs two challenge attribute strings $S^{(0)}, S^{(1)}$.
2. **Setup** is run to generate keys mpk and msk, and mpk is given to $\mathcal{A}$.
3. $\mathcal{A}$ may adaptively make a polynomial number of key queries for predicate strings S', under the following condition

   $$OverlapDist(S', S^{(0)}) = OverlapDist(S', S^{(1)}).$$

   In response, $\mathcal{A}$ is given the corresponding key $sk_{S'} \xleftarrow{R} \mathbf{KeyGen}(mpk, msk, S', t)$.
4. $\mathcal{A}$ outputs two challenge plaintexts $m^{(0)}, m^{(1)}$. If there exists a key query S' satisfying

   $$OverlapDist(S', S^{(0)}) = OverlapDist(S', S^{(1)}) = d \geq t$$

   two challenge plaintexts are equal, i.e., $m^{(0)} = m^{(1)}$.
5. A random bit b is chosen. $\mathcal{A}$ is given $ct_{S^{(b)}} \xleftarrow{R} \mathbf{Enc}(mpk, m^{(b)}, S^{(b)})$.
6. The adversary may continue to issue key queries for more predicate strings S', subject to the restriction given in steps 3 and 4. $\mathcal{A}$ is given the corresponding key $sk_{S'} \xleftarrow{R} \mathbf{KeyGen}(mpk, msk, S', t)$.
7. $\mathcal{A}$ outputs a bit b', and wins if $b' = b$.

The advantage of $\mathcal{A}$ in the above game is defined as $Adv_{\mathcal{A}}^{AHFE}(\lambda) = \Pr[\mathcal{A}\ wins] - 1/2$ for any security parameter λ.

Definition 1. *We say that a AHFE scheme is selectively secure against chosen plaintext attacks if for any polynomial-time adversaries $\mathcal{A}$, the advantage $Adv_{\mathcal{A}}^{AHFE}$ is negligible.*

IV. Preliminaries

A. Dual pairing vector spaces

In our construction, an overlap distance between two strings will be transformed into a calculation on inner product of two vectors. Thus, we can leverage IPE technique to deal with our issue. IPE first introduced by Katz et al. [15] aims to solve the disjunction of equality tests, which is based on a composite-order group. Soon after, Park et al. [16] present an improved IPE under a prime-order group. However, both of them can decrypt successfully restricted to two conditions that not only the inner product between two attribute vectors is zero but also the additional random elements satisfying some equalities, which are functioned as to achieve the attribute-hiding property and prevent the order of components in ciphertexts or secret keys from being changed, should be canceled out later in the decryption phase. Unlike them, in 2009, Okamoto et al. [8] provided a new IPE based on DPVS to ease construction, whose decryption only requires one condition – the inner product is zero. That is the main reason why we chose the IPE based on DPVS for our construction. Furthermore, in our construction we need to compute different inner products, $\left\langle \vec{X}, \vec{Y}_j \right\rangle = 0$ for $j = 1, \ldots, n - t + 1$. From a practical point of view, however, the performance of the scheme [16] is not so satisfying. Therefore, we choose an efficient IPE based on DPVS provided by Okamoto [8]. For interested readers, please refer to the literature [8]. Below, we introduce DPVS briefly.

Let N – dimensional vector space $\mathbb{V}$ be $\overbrace{\mathbb{G} \times \cdots \times \mathbb{G}}^{N}$ over $\mathbb{F}_q$, canonical basis $\mathbb{A}$ be $(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N)$ of $\mathbb{V}$, where $\boldsymbol{a}_i = (\overbrace{0, \ldots, 0}^{i-1}, G, \overbrace{0, \ldots, 0}^{N-i})$ and pairing $e : \mathbb{V} \times \mathbb{V} \to \mathbb{G}_T$. Additionally, $param_{\mathbb{V}}$ is defined as $param_{\mathbb{V}}(q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e) \xleftarrow{R} \mathcal{G}_{ob}^{DPVS}(1^\lambda, N)$, $X = (\chi_{i,j}) \xleftarrow{U} GL(N, \mathbb{F}_q)$, $(\upsilon_{i,j}) = (X^T)^{-1}$, $\boldsymbol{b}_i = \sum_{j=1}^{N} \chi_{i,j} \boldsymbol{a}_j$, $\mathbb{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_N)$, $\boldsymbol{b}_i^* = \sum_{j=1}^{N} \upsilon_{i,j} \boldsymbol{a}_j$, $\mathbb{B}^* = (\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_N^*)$.

B. Privacy preserving mapping supporting comparison

A privacy preserving mapping (PPM) supporting comparison enables a user with his/her private key to map data items into images such that, with a set of images, any entity can determine the $<$, $=$, $>$ relationship among the corresponding data items. For interested readers, please refer to the literature [9] for the details. For a higher efficiency, in our construction we will leverage the idea of PPM to locate the correct private key to decrypt a ciphertext as fast as possible.

Inspired by the idea in [9], we develop a brand-new PPM as a building block plugged into the DPVS IPE for fast decryption in our construction. The main modification is that we replace x and y of PPM in [9] with an inner product of two vectors and a threshold, respectively. Accordingly, the comparison between x and y is changed to such one between an inner product and a threshold.

C. Decisional Linear Assumption (DLIN)

The DLIN problem [17] is to guess $\beta \in \{0, 1\}$, given $(param_{\mathbb{G}}, G, G^{\xi}, G^{\kappa}, G^{\delta\xi}, G^{\sigma\kappa}, Y_\beta) \xleftarrow{R} \mathcal{G}_\beta^{DLIN}(1^\lambda)$, where $\mathcal{G}_\beta^{DLIN}(1^\lambda)$: $param_{\mathbb{G}} := (q, \mathbb{G}, \mathbb{G}_T, G, e) \xleftarrow{R} \mathcal{G}_{bgp}(1^\lambda)$, $\kappa, \delta, \xi, \sigma \xleftarrow{U} \mathbb{F}_q$, $Y_0 := G^{\delta+\sigma}$, $Y_1 \xleftarrow{U} \mathbb{G}$, return $(param_{\mathbb{G}}, G, G^{\xi}, G^{\kappa}, G^{\delta\xi}, G^{\sigma\kappa}, Y_\beta)$, for $\beta \xleftarrow{U} \{0, 1\}$.

For a probabilistic machine $\mathcal{D}$, we define the advantage of $\mathcal{D}$ for the DLIN problem as: $Adv_{\mathcal{D}}^{DLIN} := |\Pr[\mathcal{D}(1^\lambda, \varrho) \longrightarrow 1 | \varrho \xleftarrow{R} \mathcal{G}_0^{DLIN}(1^\lambda)] - \Pr[\mathcal{D}(1^\lambda, \varrho) \longrightarrow 1 | \varrho \xleftarrow{R} \mathcal{G}_1^{DLIN}(1^\lambda)]|$. The DLIN assumption [17] is: For any probabilitic polynomial-time adversary $\mathcal{D}$, the advantage $Adv_{\mathcal{D}}^{DLIN}(\lambda)$ is negligible in λ.

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

D. A key lemma

Definition 2: Problem 1 is to guess $\beta \in \{0,1\}$, given $(param_{\mathbb{V}}, \hat{\mathbb{B}}, \hat{\mathbb{B}}^*, \boldsymbol{h}^*_{\beta,1}, \{\boldsymbol{h}^*_i\}_{i=2,\ldots,n\cdot l}, \boldsymbol{g}^*_{\beta,1}, \boldsymbol{e}_\beta, g^\delta) \xleftarrow{R} \mathcal{G}^{P1}_\beta(1^\lambda, n, l)$, where

$\mathcal{G}^{P1}_\beta(1^\lambda, n, l)$: $(param_{\mathbb{V}}, \mathbb{B}, \mathbb{B}^*) \xleftarrow{R} \mathcal{G}^{AHFE}_{ob}(1^\lambda, n\cdot l+5)$,
$\omega, \tau, \delta, \rho, \gamma, \varphi, \vartheta \xleftarrow{U} \mathbb{F}_q$,
$\hat{\mathbb{B}} = (\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{n\cdot l+1}, \boldsymbol{b}_{n\cdot l+4})$,
$\hat{\mathbb{B}}^* = (\boldsymbol{b}^*_0, \ldots, \boldsymbol{b}^*_{n\cdot l+1}, \boldsymbol{b}^*_{n\cdot l+3})$,
$\boldsymbol{h}^*_{0,1} = \delta\boldsymbol{b}^*_1 + \gamma\boldsymbol{b}^*_{n\cdot l+3}$,
$\boldsymbol{h}^*_{1,1} = \delta\boldsymbol{b}^*_1 + \tau\boldsymbol{b}^*_{n\cdot l+2} + \gamma\boldsymbol{b}^*_{n\cdot l+3}$,
$\boldsymbol{h}^*_i = \delta\boldsymbol{b}^*_i$ for $i = 2, \ldots, n\cdot l$,
$\boldsymbol{g}^*_{0,1} = \delta\boldsymbol{b}^*_{n\cdot l+1} + \vartheta\boldsymbol{b}^*_{n\cdot l+3}$,
$\boldsymbol{g}^*_{1,1} = \delta\boldsymbol{b}^*_{n\cdot l+1} + \tau\boldsymbol{b}^*_{n\cdot l+2} + \vartheta\boldsymbol{b}^*_{n\cdot l+3}$,
$\boldsymbol{e}_0 = \omega\boldsymbol{b}_1 - \omega\boldsymbol{b}_{n\cdot l+1} + \varphi\boldsymbol{b}_{n\cdot l+4}$,
$\boldsymbol{e}_1 = \omega\boldsymbol{b}_1 - \omega\boldsymbol{b}_{n\cdot l+1} + \rho\boldsymbol{b}_{n\cdot l+2} + \varphi\boldsymbol{b}_{n\cdot l+4}$,
return $(param_{\mathbb{V}}, \hat{\mathbb{B}}, \hat{\mathbb{B}}^*, \boldsymbol{h}^*_{\beta,1}, \{\boldsymbol{h}^*_i\}_{i=2,\ldots,n\cdot l}, \boldsymbol{g}^*_{\beta,1}, \boldsymbol{e}_\beta, g^\delta)$ for $\beta \xleftarrow{U} \{0,1\}$.

For a probabilistic machine $\mathcal{B}$, we define the advantage of $\mathcal{B}$ for Problem 1 as: $Adv^{P1}_{\mathcal{B}}(\lambda) = |\Pr[\mathcal{B}(1^\lambda, \varrho) \to 1|\varrho \xleftarrow{R} \mathcal{G}^{P1}_0(1^\lambda, n, l)] - \Pr[\mathcal{B}(1^\lambda, \varrho) \to 1|\varrho \xleftarrow{R} \mathcal{G}^{P1}_1(1^\lambda, n, l)]|$.

Lemma 1: For any adversary $\mathcal{B}$, there is probabilistic machines $\mathcal{D}_1$ and $\mathcal{D}_2$, whose running times are essentially the same as that of $\mathcal{B}$, such that for any security parameter λ, $Adv^{P1}_{\mathcal{B}}(\lambda) \le Adv^{DLIN}_{\mathcal{D}_1}(\lambda) + Adv^{DLIN}_{\mathcal{D}_2}(\lambda) + 10/q$.

Sketch proof: Problem 1 is slightly modified from the Problem 1 in [8] which is first presented by Okamoto. For our security proof, we add $\boldsymbol{g}^*_{0,1}$ and $\boldsymbol{g}^*_{1,1}$ whose structures are similar to $\boldsymbol{h}^*_{0,1}$ and $\boldsymbol{h}^*_{1,1}$ in [8]. Since $\boldsymbol{h}^*_{0,1}$ and $\boldsymbol{h}^*_{1,1}$ in [8] are indistinguishable, $\boldsymbol{g}^*_{0,1}$ and $\boldsymbol{g}^*_{1,1}$ are also indistinguishable here. Moreover, we add one term, $-\omega\boldsymbol{b}_{n\cdot l+1}$ both in $\boldsymbol{e}_0$ and $\boldsymbol{e}_1$. Since $\boldsymbol{e}_0$, $\boldsymbol{e}_1$ in [8] are indistinguishable, and $-\omega\boldsymbol{b}_{n\cdot l+1}$ will not benefit the guess of $\beta \in \{0,1\}$ for $\boldsymbol{e}_0$, $\boldsymbol{e}_1$. Therefore $\boldsymbol{e}_0$ and $\boldsymbol{e}_1$ here are also indistinguishable and the extra term g^δ is helpless for the adversary to win the game. Except these modifications, the rest remains unchanged. Therefore the above Lemma 1 employed in our proof still follows the framework of security proof in [8].

V. Our construction

A. Overview of Technique

In this section, we show how to construct an attribute-hiding fuzzy encryption in an integer domain but with a higher efficiency. We cannot combine the techniques of IPE in [8] and the idea in [9] straightforwardly to achieve the capabilities required in our scheme. Thus, it is a challenge to achieve the *fuzziness*, *efficiency* and *attribute-hiding* in one encryption system simultaneously.

- **Fuzziness.** In public key fuzzy encryption with attribute-hiding, [7] employed the Hamming distance plugged into IPE to introduce the concept of fuzziness. Nevertheless, it is noticed that this scheme will raise a misjudgment when we leverage it to cope with an integer domain as in ours. In order to conduct the fuzziness in an integer domain without a misjudgment, we develop two new encoding techniques as follows.

  String encoding and threshold encoding. Let $U = \{1, \ldots, l\}$ be an attribute universe and l be a positive integer. According to the following rule I, two strings $S = \{s_1 s_2 \ldots s_n\}$ and $S' = \{s'_1 s'_2 \ldots s'_n\}$ can be encoded two $(n \cdot l)$ – dimension vectors $\vec{U}$ and $\vec{V}$, respectively, where $j = 1, \ldots, n$.

$$I: \quad \vec{U}_{\{s_1 s_2 \ldots s_n\}} = \begin{cases} u_i = 1, & if \quad i = s_j + (j-1)\cdot l \\ u_i = 0, & \text{otherwise} \end{cases}$$
$$\vec{V}_{\{s'_1 s'_2 \ldots s'_n\}} = \begin{cases} v_i = 1, & if \quad i = s'_j + (j-1)\cdot l \\ v_i = 0, & \text{otherwise} \end{cases}$$

  In order to embed a threshold t into the above vector $\vec{V}$, we should add $n-t+1$ bits, $t, \ldots, n$ at the end of it for all possible decryption cases, which eventually consists of a vector $\vec{Y} = (\vec{V}, t, \ldots, n)$. On the other hand, we simply need to add one bit "-1" at the end of vector $\vec{U}$, i.e., $\vec{X} = \{\vec{U}, -1\}$. Consequently, we have the resulted two vectors $\vec{X} = (\vec{U}, -1) = (x_1, \ldots, x_{n\cdot l}, -1)$ and $\vec{Y} = (\vec{V}, t, \ldots, n) = (y_1, \ldots, y_{n\cdot l}, y_{n\cdot l+1}, \ldots, y_{n\cdot(l+1)-t+1})$. Besides that, $n-t+1$ separate vectors $\vec{Y}_j$ can be deprived from vector $\vec{Y}$, where $\vec{Y}_j = (\vec{V}, y_{n\cdot l+j})$, $j = 1, \ldots, n-t+1$.
- **Efficiency.** The existing attribute-hidng fuzzy encryption [7] demands a loop to try all possible decryption keys one by one and unavoidably blows up the computational complexity of decryption exponential with threshold. To improve the efficiency, we borrow the idea of privacy-preserving mapping in [9] and come up with a modified version suitable to our construction as follows. First, during decryption process, we build a new list $\mathcal{L}_3$. It is generated from part of ciphertexts and the list $\mathcal{L}_1$ from private keys to index all possible overlap distances and the corresponding decryption keys for the successful decryption. Second, we use part of ciphertexts and private keys to compute the result $\mathcal{J}$, which is functioned as to test the overlap distance. If the overlap distance between two strings is no less than the threshold, one item in the list of $\mathcal{L}_3/\mathcal{J}$ can be found its trace in the list $\mathcal{L}_2$ produced from ciphertexts. Then the correct decryption key will be picked up immediately according to the index of this item in the list $\mathcal{L}_2$. Otherwise, no any correct decryption key can be found. This technique enables our scheme to fast decrypt the ciphertext without the need to iteratively try all possible decryption keys one by one and the computational complexity of decryption is negative correlation with threshold. Therefore, it is practical for some real-life applications, especially those with a large threshold.
- **Attribute-hiding.** To achieve the attribute-hiding security, the technique of DPVS is introduced in IPE [8], we also leverage this technique but with a slight modification to reach the attribute-hiding in our construction. We

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

modify it as follows: 1) Transform two attribute vectors to multiple vectors in order to embed all possible values of the threshold for fuzziness; 2) Separate one private key in literature [8] into multiple private keys in order to test the overlap distance and locate the correct decryption key for fast decryption.

B. Scheme

In this section, we present a concrete scheme of attribute-hiding fuzzy encryption (AHFE) which integrates our new encoding and PPM mechanisms as well as the DPVS technique [8] to achieve three functionalities: *fuzziness*, *efficiency*, and *attribute-hiding*. Set U, S, S', n, l as before, with the restriction $q \gg 2n$ [1]. Now we describe our construction as follows:

- **Setup**$(1^\lambda, n, l)$: $(param_{\mathbb{V}}, \mathbb{B} = (\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{n\cdot l+4}), \mathbb{B}^* = (\boldsymbol{b}_0^*, \ldots, \boldsymbol{b}_{n\cdot l+4}^*)) \xleftarrow{R} \mathcal{G}_{ob}^{AHFE}(1^\lambda, n\cdot l+5), \alpha \xleftarrow{U} \mathbb{F}_q, \hat{\mathbb{B}} = (\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{n\cdot l+1}, \boldsymbol{b}_{n\cdot l+4}), \hat{\mathbb{B}}^* = (\boldsymbol{b}_0^*, \ldots, \boldsymbol{b}_{n\cdot l+1}^*, \boldsymbol{b}_{n\cdot l+3}^*)$, return $mpk = (1^\lambda, param_{\mathbb{V}}, \hat{\mathbb{B}}, \{g_T^{\alpha^i}\}, i \in \{0, \ldots, n\})$, $msk = (\hat{\mathbb{B}}^*, \alpha)$.
- **KeyGen**(mpk, msk, S', t): $\sigma, \eta, r_0, \ldots, r_n, \varsigma_0, \ldots, \varsigma_{n-t} \xleftarrow{U} \mathbb{F}_q$, the key generation algorithm first transforms string S' to vector $\vec{Y} = (y_1, \ldots, y_{(n\cdot l)+(n-t))}) = (\vec{V}, t, \ldots, n)$. It then computes:

$$\boldsymbol{k}_1^* = (0, \overbrace{\quad \sigma\vec{V} \quad}^{n\cdot l}, \sigma t, 0, \eta, 0)_{\mathbb{B}^*},$$

$$\boldsymbol{t}_0^* = (1, \overbrace{\quad r_0\vec{V} \quad}^{n\cdot l}, r_0 t, 0, \varsigma_0, 0)_{\mathbb{B}^*},$$

$$\boldsymbol{t}_1^* = (1, \overbrace{\quad r_1\vec{V} \quad}^{n\cdot l}, r_1(t+1), 0, \varsigma_1, 0)_{\mathbb{B}^*},$$

$$\vdots$$

$$\boldsymbol{t}_{n-t}^* = (1, \overbrace{\quad r_{n-t}\vec{V}}^{n\cdot l}, r_{n-t} n, 0, \varsigma_{n-t}, 0)_{\mathbb{B}^*}$$

and generates a list $\mathcal{L}_1 = \{g^{(\alpha^i + i\sigma)}\}$, $i \in \{0, \ldots, n-t\}$. The keygen algorithm returns $sk_{S'} = (\boldsymbol{k}_1^*, \boldsymbol{t}_0^*, \boldsymbol{t}_1^*, \ldots, \boldsymbol{t}_{n-t}^*, \mathcal{L}_1)$.

- **Enc**(mpk, m, S): $\omega, \varphi, \zeta \xleftarrow{U} \mathbb{F}_q$, the encryption algorithm first transforms string S to vector $\vec{X} = (x_1, \ldots, x_{(n\cdot l)+1}) = (\vec{U}, -1)$. Then, it computes:

$$c_1 = (\zeta, \overbrace{\quad \omega\vec{U} \quad}^{n\cdot l}, -\omega, 0, 0, \varphi)_{\mathbb{B}}, \; c_2 = g_T^\zeta m, c_3 = g^\omega,$$

and generates a list $\mathcal{L}_2 = \{g_T^{\omega\alpha^i}\}, i \in \{0, \ldots, n\}$. The Enc algorithm returns $ct_S = (c_1, c_2, c_3, \mathcal{L}_2)$.
- **Dec**$(mpk, sk_{S'}, ct_S, n, t)$:
  $Step$ 1. Compute $e(c_3, g^{(\alpha^i + i\sigma)}) = g_T^{\omega(\alpha^i + i\sigma)}$ for each item in the list $\mathcal{L}_1$ and generate a new list $\mathcal{L}_3 = \{g_T^{\omega(\alpha^i + i\sigma)}\}$ with an index list $\mathcal{I} = \{0, \ldots, n-t\}$.
  $Step$ 2. Compute $e(c_1, \boldsymbol{k}_1^*) = g_T^{\omega\sigma(\vec{U}\cdot\vec{V} - t)}$, set it as $\mathcal{J}$. Next we compute $g_T^{\omega(\alpha^i + i\sigma)} / \mathcal{J} = g_T^{\omega\alpha^i + \omega\sigma i - \omega\sigma(\vec{U}\cdot\vec{V} - t)}$, $i \in \{0, \ldots, n-t\}$ and these results consisit of the items in the list $\mathcal{T}$. If the overlap distance between S and S' is greater than or equal to the threshold t and then, there exists one item in $\mathcal{T}$ being the format of $g_T^{\omega\alpha^i}$, which must fall into the list $\mathcal{L}_2$ with the index d, $\left\langle \vec{U}\cdot\vec{V} \right\rangle - t = d \geq 0$.
  $Step$ 3. Compute $m = c_2 / e(c_1, \boldsymbol{t}_d^*)$, return m.
  Correctness If the overlap distance between S and S' is $d+t$, then $\left\langle \vec{U}\cdot\vec{V} \right\rangle = d+t$. Therefore, we are able to locate the correct decryption key $t_d^* = (1, \overbrace{\quad r_d\vec{V} \quad}^{n\cdot l}, r_d(d+t), 0, \varsigma_d)_{\mathbb{B}^*}$, which is able to decrypt ciphertext succcessfully. Consequently, $e(c_1, \boldsymbol{t}_d^*) = g_T^{\zeta + \omega r_d(\vec{U}\cdot\vec{V} - (d+t))} = g_T^\zeta$. Conversely, for those overlap distances between S and S' less than the threshold, i.e., $d < t$, there are no any corret decryption keys for them.

[1] To avoid incorrect decryption, namely, the decryption keys corresponding to indexes greater than $q/2$ will not appear in the list we build during decryption phase.

VI. Security

A. Security discussion

- **Multiple evaluation attack.** A multiple evaluation attack occurs when an adversary decrypts the same ciphertext multiple times aiming at revealing the attribute set encapsulated in the ciphertext. To resist this attack, we introduce items $g^{\alpha^i}, i \in 0, \ldots, n-t$ in the list $\mathcal{L}_1$ during private key generation. As a result, the same items $g_T^{\alpha^i}, i \in 0, \ldots, n-t$ will be regenerated in the first step of the decryption algorithm, which will consist of a new list $\mathcal{L}_3$. All these items cannot be modified by anyone including the adversary, such as through adding some new items $g_T^{\alpha^i}$, such that $i < 0$ into $\mathcal{L}_3$. This technique will prevent the adversary from decrypting the ciphertext with an illegitimate decryption key. That is, the adversary will fail to decrypt the ciphertext with such a key that the overlap distance between two strings less than a threshold because the illegitimate key $g^{\alpha^i}, i < 0$ will not arise in the List $\mathcal{L}_3$. However, the multiple evaluation attack will not prevent the adversary from learning the overlap distance once decryption is successful if he holds a legitimate decryption key. Even so, the adversary still learns nothing about the knowledge of attribute and the security of attribute-hiding is still remained.
- **Private key computable attack.** Without accessing the key generation center, the adversary tries to figure out a valid decryption key with a specific evaluation string from some known decryption keys associated with different evaluation strings through linear operation on them since pieces of decryption keys seem linear with each other. However, this effort will be unsuccessful because different randoms are embedded into different pieces of decryption keys to resist this attack and these randoms are chosen by the key generation center, which is unknown

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

to anyone. Therefore, the adversary will learn nothing about the knowledge of attributes and messages if no a valid decrypton key issued by the key generation center.

B. Semi-functional algorithms

Semi-functional keys:

$$\boldsymbol{k}_1^* = (0, \overbrace{\quad \sigma\vec{V} \quad}^{n\cdot l}, \sigma t, \boxed{\kappa}, \eta, 0)_{\mathbb{B}^*},$$
$$\boldsymbol{t}_0^* = (1, \overbrace{\quad r_0\vec{V} \quad}^{n\cdot l}, r_0 t, \boxed{\phi_0}, \varsigma_0, 0)_{\mathbb{B}^*},$$
$$\vdots$$
$$\boldsymbol{t}_{n-t}^* = (1, \overbrace{\quad r_{n-t}\vec{V} \quad}^{n\cdot l}, r_{n-t} n, \boxed{\phi_{n-t}}, \varsigma_{n-t}, 0)_{\mathbb{B}^*},$$
where $\kappa, \phi_0, \ldots, \phi_{n-t} \xleftarrow{U} \mathbb{F}_q$.

Semi-functional ciphertexts:

$$c_1 = (\zeta, \overbrace{\omega\vec{U}^{(b)}}^{n\cdot l}, -\omega, \boxed{\rho}, 0, \varphi)_{\mathbb{B}},\ c_2 = g_T^{\zeta} m^{(b)},\ c_3 = g^{\omega}.$$
where $\rho \xleftarrow{U} \mathbb{F}_q$.

Normal keys:

$$\boldsymbol{k}_1^* = (0, \overbrace{\quad \sigma\vec{V} \quad}^{n\cdot l}, \sigma t, \boxed{0}, \eta, 0)_{\mathbb{B}^*},$$
$$\boldsymbol{t}_0^* = (1, \overbrace{\quad r_0\vec{V} \quad}^{n\cdot l}, r_0 t, \boxed{0}, \varsigma_0, 0)_{\mathbb{B}^*},$$
$$\vdots$$
$$\boldsymbol{t}_{n-t}^* = (1, \overbrace{\quad r_{n-t}\vec{V} \quad}^{n\cdot l}, r_{n-t} n, \boxed{0}, \varsigma_{n-t}, 0)_{\mathbb{B}^*},$$

Normal ciphertexts:

$$c_1 = (\zeta, \overbrace{\omega\vec{U}^{(b)}}^{n\cdot l}, -\omega, \boxed{0}, 0, \varphi)_{\mathbb{B}},\ c_2 = g_T^{\zeta} m^{(b)},\ c_3 = g^{\omega}.$$

C. Outline of proof

At the top-level strategy of the security proof, we employ the dual system encryption originated from Waters [18]. As same as [18], we define ciphertexts and secret keys which also have two forms, normal and semi-functional. The real system only uses normal ciphertexts and normal secret keys, and semi-functional ciphertexts and keys are only used in a sequence of security games for the security proof.

However, there are three main differences between the security proof in [18] and ours. First, in the security proof in [18], the challenge ciphertext and all non-matching keys are changed to semi-functional, i.e., the former has random values in the 4-wise hidden subgroups and the latter has random values in the 3-wise hidden subgroups, respectively, while in ours the challenge ciphertext and all non-matching keys are changed to semi-functional, i.e., they have random values in the 1-dimensional hidden subspaces of DPVS. Second, the security model in [18] only allows non-matching queries while in our security model, we also allow matching queries as well as non-matching ones. Lastly, in the security proof in [18], the first game changes the challenge ciphertext to semi-functional and then all non-matching keys are modified from normal to semi-functional one by one through a sequence of games. After that, challenge ciphertext is turned into an encryption over a random number in the last game while in ours, we change challenge ciphertext and non-matching keys from normal to semi-functional simultaneously. After that, the challenge ciphertext becomes an unbiased form in the last game which is an encryption over a random vector and a random number.

We employ Game 0 (original selective-security game) through Game 2.

Game 0: All matching and non-matching keys are normal. The challenge string $S^{(b)}$ is encoded into $\vec{U}^{(b)}$ and the challenge ciphertext is normal over $(\vec{U}^{(b)}, m^{(b)})$, where $b \in \{0, 1\}$ is a random bit.

Game 1: The matching keys remain normal while the non-matching keys are changed into semi-functional. Meanwhile, the challenge ciphertext is also changed into a semi-functional over $(\vec{U}^{(b)}, m^{(b)})$.

Game 2:

Case 1. Two challenge plaintext are not equal: $m^{(0)} \neq m^{(1)}$. All queried keys are non-matching and semi-functional. The challenge ciphertext is a semi-functional encryption. The challenge ciphertext (c_1, c_2, c_3) is like as follows:

$$c_1 = (\zeta', \overbrace{\omega_0\vec{U}^{(0)} + \omega_1\vec{U}^{(1)}}^{n\cdot l}, -\omega, \rho, 0, \varphi)_{\mathbb{B}},\ c_2 = g_T^{\zeta} m^{(b)},\ c_3 = g^{\omega},$$
where $\zeta', \zeta\omega_0, \omega_1 \xleftarrow{U} \mathbb{F}_q$.

Case 2. Two challenge plaintext are equal: $m^{(0)} = m^{(1)} = m$. The matching keys are normal while non-matching keys are semi-functional. The challenge ciphertext c_1 is a semi-functional encryption. The challenge ciphertext (c_1, c_2, c_3) is like as follows:

$$c_1 = (\zeta, \overbrace{\omega_0\vec{U}^{(0)} + \omega_1\vec{U}^{(1)}}^{n\cdot l}, -\omega, \rho, 0, \varphi)_{\mathbb{B}},\ c_2 = g_T^{\zeta} m,\ c_3 = g^{\omega},$$
where $\omega_0, \omega_1 \xleftarrow{U} \mathbb{F}_q$.

D. Proof

Since the security of the IPE on DPVS [8] we leveraged is almost tightly reduced to the DLIN assumption in the standard model, we also have the following theorem.

Theorem 1: The proposed AHFE scheme is selectively fully attribute-hiding against chosen plaintext attacks under the DLIN assumption.

For any adversary $\mathcal{A}$, there exist probabilistic machines $\mathcal{D}_1$ and $\mathcal{D}_2$, whose running times are essentially the same as that of $\mathcal{A}$, such that for any security parameter λ,

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

$$Adv_{\mathcal{A}}^{AHFE}(\lambda) \leq Adv_{\mathcal{D}_1}^{DLIN}(\lambda) + Adv_{\mathcal{D}_2}^{DLIN}(\lambda) + \epsilon,$$

where $\epsilon = 12/q$.

To prove this theorem, we employ Game 0 through Game 2. Let $Adv_{\mathcal{A}}^{(0)}(\lambda)$ be $Adv_{\mathcal{A}}^{AHFE}(\lambda)$ in Game 0, and $Adv_{\mathcal{A}}^{i}(\lambda)$ $(i = 1, 2)$ be the advantage of $\mathcal{A}$ in Game i. We present two lemmas (Lemma 2 and 3) that evaluate advantage gaps between neighbouring games and Lemma 4 that turns out the adversary in the last game is zero, i.e., $Adv_{\mathcal{A}}^{(2)}(\lambda) = 0$. From these lemmas as well as Lemma 1, we have

$$\begin{aligned} &Adv_{\mathcal{A}}^{AHFE}(\lambda) = Adv_{\mathcal{A}}^{(0)}(\lambda) \\ &\leq \sum_{i=0}^{1} |Adv_{\mathcal{A}}^{(i)}(\lambda) - Adv_{\mathcal{A}}^{(i+1)}(\lambda)| + Adv_{\mathcal{A}}^{(2)}(\lambda) \\ &\leq Adv_{\mathcal{B}}^{P1}(\lambda) + 2/q \\ &\leq Adv_{\mathcal{D}_1}^{DLIN}(\lambda) + Adv_{\mathcal{D}_2}^{DLIN}(\lambda) + 12/q. \end{aligned}$$

Lemma 2: For any adversary $\mathcal{A}$, there exists a probabilistic machine $\mathcal{B}_1$, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ,

$$|Adv_{\mathcal{A}}^{(0)}(\lambda) - Adv_{\mathcal{A}}^{(1)}(\lambda)| \leq Adv_{\mathcal{B}_1}^{P1}(\lambda) + 1/q.$$

proof. In order to prove Lemma 2, we will show the distribution of public key, queried decryption keys and challenge ciphertext in Game 0 and that in Game 1 are indistinguishable. For that purpose, we construct a probabilistic machine $\mathcal{B}_1$ against Problem 1 by using any adversary $\mathcal{A}$ in a security game (Game 0 or 1) as a black box as follows:

1) $\mathcal{B}_1$ is given Problem 1 instance $(param_{\mathbb{V}}, \hat{\mathbb{B}}, \hat{\mathbb{B}}^*, \boldsymbol{h}^*_{\beta,1}, \{\boldsymbol{h}^*_i\}_{i=2,\ldots,n\cdot l}, \boldsymbol{g}^*_{\beta,1}, \boldsymbol{e}_\beta, g^\delta)$.
2) $\mathcal{B}_1$ plays a role of challenger in the security game against an adversary $\mathcal{A}$.
3) When $\mathcal{B}_1$ obtains the challenge strings $S^{(0)}, S^{(1)}$ commited by adversary $\mathcal{A}$ in the step 1 of the game and then transforms them to $\vec{U}^{(0)}, \vec{U}^{(1)}$. $\mathcal{B}_1$ selects challenge bit $b \xleftarrow{U} \{0, 1\}$. For $\vec{e} = (1, 0, \ldots, 0)$ with $n \cdot l$ dimensions and $\vec{e}' = (1, 0, \ldots, 0) = (\vec{e}, 0)$ with $n \cdot l + 1$ dimensions, there exists a matrix $\Pi_1 = (\pi_{i,j}) \in GL(n \cdot l, \mathbb{F}_q)$, $\Pi_2 = (\pi'_{i,j}) \in GL(n \cdot l + 1, \mathbb{F}_q)$ such at $\vec{e} = \vec{U}^{(b)} \cdot \Pi_1$ and $\vec{e}' = (\vec{U}^{(b)}, -1) \cdot \Pi_2$. Since $\vec{U}^{(b)} \neq \vec{0}$, $\mathcal{B}_1$ calculates such Π_1^* and Π_2^*, and sets $\Pi_1^* = (\pi^*_{i,j}) = (\Pi_1^T)^{-1}$ and $\Pi_2^* = (\pi^{*'}_{i,j}) = (\Pi_2^T)^{-1}$. Public parameter mpk is then calculated as follows and $\mathcal{B}_1$ returns mpk to $\mathcal{A}$: $\boldsymbol{d}_i = \sum_{j=1}^{n\cdot l} \pi_{i,j} \boldsymbol{b}_j$ from Π_1, $\boldsymbol{d}^*_i = \sum_{j=1}^{n\cdot l} \pi^*_{i,j} \boldsymbol{b}^*_j$ from Π_1^*, $i = 1, \ldots, n \cdot l$, $\hat{\mathbb{D}} = (\boldsymbol{b}_0, \boldsymbol{d}_1, \ldots, \boldsymbol{d}_{n\cdot l}, \boldsymbol{b}_{n\cdot l+1}, \boldsymbol{b}_{n\cdot l+4})$. $\mathcal{B}_1$ chooses $\alpha \in \mathbb{F}_q$ and sets $mpk = (1^\lambda, param_{\mathbb{V}}, \hat{\mathbb{D}}, \{g_T^{\alpha^i}\}, i \in \{0, \ldots, n\})$.
4) When a key query is issued for a vector $\vec{V}$ transformed from a predicate string S', $\mathcal{B}_1$ calculates $\vec{z} = (z_1, \ldots, z_{n\cdot l}) = \vec{V} \cdot \Pi_1^*$, $\vec{z}' = (z'_1, \ldots, z'_{n\cdot l+1}) = (\vec{V}, d) \cdot \Pi_2^*$, where d is for

$$OverlapDist(S', S^{(0)}) = OverlapDist(S', S^{(1)}) \geq t.$$

$\mathcal{B}_1$ picks up random numbers $\psi, \psi_0, \ldots, \psi_{n-t}, \kappa, \kappa_0, \ldots, \kappa_{n-t}, \eta, \varsigma_0, \ldots, \varsigma_{n-t}$ and calculates:

$$\begin{aligned} \boldsymbol{k}_1^* &= z_1(\psi \boldsymbol{b}_1^* + \kappa z_1' \boldsymbol{h}_{\beta,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi \boldsymbol{b}_i^* + \kappa z_1' \boldsymbol{h}_i^*) + \\ &\quad t(\psi \boldsymbol{b}^*_{n\cdot l+1} + \kappa z_1' \boldsymbol{g}^*_{\beta,1}) + \eta \boldsymbol{b}^*_{n\cdot l+3} \\ \boldsymbol{t}_0^* &= \boldsymbol{b}_0^* + z_1(\psi_0 \boldsymbol{b}_1^* + \kappa_0 z_1' \boldsymbol{h}_{\beta,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi_0 \boldsymbol{b}_i^* + \kappa_0 z_1' \boldsymbol{h}_i^*) + \\ &\quad t(\psi_0 \boldsymbol{b}^*_{n\cdot l+1} + \kappa_0 z_1' \boldsymbol{g}^*_{\beta,1}) + \varsigma_0 \boldsymbol{b}^*_{n\cdot l+3}, \\ &\vdots \\ \boldsymbol{t}_{n-t}^* &= \boldsymbol{b}_0^* + z_1(\psi_{n-t} \boldsymbol{b}_1^* + \kappa_{n-t} z_1' \boldsymbol{h}_{\beta,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi_{n-t} \boldsymbol{b}_i^* + \\ &\quad \kappa_{n-t} z_1' \boldsymbol{h}_i^*) + n(\psi_{n-t} \boldsymbol{b}^*_{n\cdot l+1} + \kappa_{n-t} z_1' \boldsymbol{g}^*_{\beta,1}) + \varsigma_{n-t} \boldsymbol{b}^*_{n\cdot l+3}. \end{aligned}$$

Then, $\mathcal{B}_1$ returns $sk_{S'} = (\boldsymbol{k}_1^*, \boldsymbol{t}_0^*, \ldots, \boldsymbol{t}_{n-t}^*, \mathcal{L}_1 = \{g^{\alpha^i + i(\psi + \kappa \delta z_1')}\}, i \in \{0, \ldots, n\})$ to $\mathcal{A}$.
5) After obtaining the challenge plaintexts $m^{(0)}, m^{(1)}$, $\mathcal{B}_1$ calculates and returns challenge ciphertexts s.t. $c_1 = \zeta \boldsymbol{b}_0 + \boldsymbol{e}_\beta$, $c_2 = g_T^\zeta m^{(b)}$, $c_3 = g^\omega, \mathcal{L}_2 = \{g_T^{\omega\alpha^i}\}, i \in \{0, \ldots, n\}$.
6) After the encryption query, KeyGen oracle simulation for a key query is executed as above.
7) $\mathcal{A}$ outputs bit b'. If $b = b'$, $\mathcal{B}_1$ outputs $\beta' = 1$. Otherwise, $\mathcal{B}_1$ outputs $\beta' = 0$.

Claim 1: Public parameter mpk generated in step 3 above has the same distribution as that in Game 0 (and Game 1).

proof. Let $D = \begin{pmatrix} I_1 & 0 & 0 \\ 0 & \Pi_1 & 0 \\ 0 & 0 & I_2 \end{pmatrix}$ be square $(n \cdot l + 5) \times (n \cdot l + 5)$ matrix composed of Π_1 and the identity matrices I_1, I_2. Then, basis $\mathbb{D} = (\boldsymbol{b}_0, \boldsymbol{d}_1, \ldots, \boldsymbol{d}_{n\cdot l}, \boldsymbol{b}_{n\cdot l+1}, \ldots, \boldsymbol{b}_{n\cdot l+4})$ is obtained from basis $\mathbb{B}$ by the linear transformation determined by D. Hence, its distribution is uniform. Therefore, $\mathbb{D}$ in step 3 has the same distribution as that in Game 0 (and Game 1).

Claim 2: If $\beta = 0$ (resp. $\beta = 1$), the distribution of partical ciphertext (c_1, c_2, c_3) generated in step 5 is the same as that in Game 0 (resp. Game 1).

proof. If $\beta = 0$, $c_1 = \zeta \boldsymbol{b}_0 + \boldsymbol{e}_0 = (\zeta, \omega\vec{e}, -\omega, 0, 0, \varphi)_{\mathbb{B}} = (\zeta, \omega\vec{U}^{(b)} \cdot \Pi_1, -\omega, 0, 0, \varphi)_{\mathbb{B}} = (\zeta, \omega\vec{U}^{(b)}, -\omega, 0, 0, \varphi)_{\mathbb{D}}$,

$c_2 = g_T^\zeta m^{(b)}, c_3 = g^\omega$, where $\vec{e} = (1, 0, \ldots, 0)$. This is the target ciphertext in Game 0 with mpk.

If $\beta = 1$, $c_1 = \zeta \boldsymbol{b}_0 + \boldsymbol{e}_1 = (\zeta, \omega\vec{e}, -\omega, \rho, 0, \varphi)_{\mathbb{B}} = (\zeta, \omega\vec{U}^{(b)} \cdot \Pi_1, -\omega, \rho, 0, \varphi)_{\mathbb{B}} = (\zeta, \omega\vec{U}^{(b)}, -\omega, \rho, 0, \varphi)_{\mathbb{D}}$,

$c_2 = g_T^\zeta m^{(b)}, c_3 = g^\omega$, where $\vec{e} = (1, 0, \ldots, 0)$. This is the target ciphertext in Game 1 with mpk.

Claim 3: If $\beta = 0$ (resp. $\beta = 1$), the distribution of $\boldsymbol{k}_1^*, \boldsymbol{t}_0^*, \ldots, \boldsymbol{t}_{n-t}^*$ generated in step 4 and 6 is the same as that in Game 0 (resp. Game 1 expect with probability $1/q$).

proof. First, it is easy to verify that basis $\mathbb{D}^* = (\boldsymbol{b}_0^*, \boldsymbol{d}_1^*, \ldots, \boldsymbol{d}_{n\cdot l}^*, \boldsymbol{b}_{n\cdot l+1}^*, \cdots, \boldsymbol{b}_{n\cdot l+4}^*)$ obtained by the linear transformation $(D^T)^{-1}$ using Π_1^*.

That is, it is dual orthonormal to basis $\mathbb{D} =$

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

$(\boldsymbol{b}_0, \boldsymbol{d}_1, \ldots, \boldsymbol{d}_{n\cdot l}, \boldsymbol{b}_{n\cdot l+1}, \cdots, \boldsymbol{b}_{n\cdot l+4})$, where D is defined in the proof of Claim 1. Therefore, we can consider $\boldsymbol{k}_1^*, \boldsymbol{t}_0^*, \ldots, \boldsymbol{t}_{n-t}^*$ w.r.t this dual orthonormal to basis.
If $\beta = 0$,

$$\begin{aligned}
\boldsymbol{k}_1^* &= z_1(\psi \boldsymbol{b}_1^* + \kappa z_1^{'} \boldsymbol{h}_{0,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi \boldsymbol{b}_i^* + \kappa z_1^{'} \boldsymbol{h}_i^*) + \\
&\quad t(\psi \boldsymbol{b}_{n\cdot l+1}^* + \kappa z_1^{'} \boldsymbol{g}_{0,1}^*) + \eta \boldsymbol{b}_{n\cdot l+3}^* \\
&= (0, (\psi + \kappa\delta z_1^{'})\vec{z}, (\psi + \kappa\delta z_1^{'})t, 0, \eta + \gamma, 0)_{\mathbb{B}^*} \\
&= (0, (\psi + \kappa\delta z_1^{'})\vec{V} \cdot \Pi_1^*, (\psi + \kappa\delta z_1^{'})t, 0, \eta + \gamma, 0)_{\mathbb{B}^*} \\
&= (0, (\psi + \kappa\delta z_1^{'})\vec{V}, (\psi + \kappa\delta z_1^{'})t, 0, \eta + \gamma, 0)_{\mathbb{D}^*},
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{t}_0^* &= \boldsymbol{b}_0^* + z_1(\psi_0 \boldsymbol{b}_1^* + \kappa_0 z_1^{'} \boldsymbol{h}_{0,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi_0 \boldsymbol{b}_i^* + \kappa_0 z_1^{'} \boldsymbol{h}_i^*) + \\
&\quad t(\psi_0 \boldsymbol{b}_{n\cdot l+1}^* + \kappa_0 z_1^{'} \boldsymbol{g}_{0,1}^*) + \varsigma_0 \boldsymbol{b}_{n\cdot l+3}^* \\
&= (1, (\psi_0 + \kappa_0\delta z_1^{'})\vec{z}, (\psi_0 + \kappa_0\delta z_1^{'})t, 0, \varsigma_0 + \vartheta, 0)_{\mathbb{B}^*} \\
&= (1, (\psi_0 + \kappa_0\delta z_1^{'})\vec{V} \cdot \Pi_1^*, (\psi_0 + \kappa_0\delta z_1^{'})t, 0, \varsigma_0 + \vartheta, 0)_{\mathbb{B}^*} \\
&= (1, (\psi_0 + \kappa_0\delta z_1^{'})\vec{V}, (\psi_0 + \kappa_0\delta z_1^{'})t, 0, \varsigma_0 + \vartheta, 0)_{\mathbb{D}^*},
\end{aligned}$$

$\vdots$

$$\begin{aligned}
\boldsymbol{t}_{n-t}^* &= \boldsymbol{b}_0^* + z_1(\psi_{n-t} \boldsymbol{b}_1^* + \kappa_{n-t} z_1^{'} \boldsymbol{h}_{0,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi_{n-t} \boldsymbol{b}_i^* + \\
&\quad \kappa_{n-t} z_1^{'} \boldsymbol{h}_i^*) + n(\psi_{n-t} \boldsymbol{b}_{n\cdot l+1}^* + \kappa_{n-t} z_1^{'} \boldsymbol{g}_{0,1}^*) + \varsigma_{n-t} \boldsymbol{b}_{n\cdot l+3}^* \\
&= (1, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})\vec{z}, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})n, 0, \varsigma_{n-t} + \vartheta, 0)_{\mathbb{B}^*} \\
&= (1, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})\vec{V} \cdot \Pi_1^*, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})n, 0, \\
&\quad \varsigma_{n-t} + \vartheta, 0)_{\mathbb{B}^*} \\
&= (1, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})\vec{V}, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})n, 0, \varsigma_{n-t} + \vartheta, 0)_{\mathbb{D}^*},
\end{aligned}$$

If $\beta = 1$,

$$\begin{aligned}
\boldsymbol{k}_1^* &= z_1(\psi \boldsymbol{b}_1^* + \kappa z_1^{'} \boldsymbol{h}_{1,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi \boldsymbol{b}_i^* + \kappa z_1^{'} \boldsymbol{h}_i^*) + t(\psi \boldsymbol{b}_{n\cdot l+1}^* + \\
&\quad \kappa z_1^{'} \boldsymbol{g}_{1,1}^*) + \eta \boldsymbol{b}_{n\cdot l+3}^* \\
&= (0, (\psi + \kappa\delta z_1^{'})\vec{z}, (\psi + \kappa\delta z_1^{'})t, \kappa z_1^{'}\tau(z_1 + t), \eta + \gamma, 0)_{\mathbb{B}^*} \\
&= (0, (\psi + \kappa\delta z_1^{'})\vec{V} \cdot \Pi_1^*, (\psi + \kappa\delta z_1^{'})t, \kappa\tau(z_1 + t)((\vec{U}^{(b)}, -1) \cdot \\
&\quad \Pi_2) \cdot ((\vec{V}, d) \cdot \Pi_2^*)^T, \eta + \gamma, 0)_{\mathbb{B}^*} \\
&= (0, (\psi + \kappa\delta z_1^{'})\vec{V}, (\psi + \kappa\delta z_1^{'})t, \kappa\tau(z_1 + t)(\vec{U}^{(b)} \cdot \vec{V} - d), \\
&\quad \eta + \gamma, 0)_{\mathbb{D}^*},
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{t}_0^* &= \boldsymbol{b}_0^* + z_1(\psi_0 \boldsymbol{b}_1^* + \kappa_0 z_1^{'} \boldsymbol{h}_{1,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi_0 \boldsymbol{b}_i^* + \kappa_0 z_1^{'} \boldsymbol{h}_i^*) + \\
&\quad t(\psi_0 \boldsymbol{b}_{n\cdot l+1}^* + \kappa_0 z_1^{'} \boldsymbol{g}_{1,1}^*) + \varsigma_0 \boldsymbol{b}_{n\cdot l+3}^* \\
&= (1, (\psi_0 + \kappa_0\delta z_1^{'})\vec{z}, (\psi_0 + \kappa_1\delta z_1^{'})t, \kappa_0 z_1^{'}\tau(z_1 + t), \varsigma_0 + \vartheta, 0)_{\mathbb{B}^*} \\
&= (1, (\psi_0 + \kappa_0\delta z_1^{'})\vec{V} \cdot \Pi_1^*, (\psi_0 + \kappa_0\delta z_1^{'})t, \kappa_0 z_1^{'}\tau(z_1 + t), \varsigma_0 + \vartheta, 0)_{\mathbb{B}^*} \\
&= (1, (\psi_0 + \kappa_0\delta z_1^{'})\vec{V}, (\psi_0 + \kappa_0\delta z_1^{'})t, \kappa_0 z_1^{'}\tau(z_1 + t)((\vec{U}^{(b)}, -1) \cdot \\
&\quad \Pi_2) \cdot ((\vec{V}, d) \cdot \Pi_2^*)^T, \varsigma_0 + \vartheta, 0)_{\mathbb{D}^*},
\end{aligned}$$

$\vdots$

$$\begin{aligned}
\boldsymbol{t}_{n-t}^* &= \boldsymbol{b}_0^* + z_1(\psi_{n-t} \boldsymbol{b}_1^* + \kappa_{n-t} z_1^{'} \boldsymbol{h}_{1,1}^*) + \sum_{i=2}^{n\cdot l} z_i(\psi_{n-t} \boldsymbol{b}_i^* + \\
&\quad \kappa_{n-t} z_1^{'} \boldsymbol{h}_i^*) + n(\psi_{n-t} \boldsymbol{b}_{n\cdot l+1}^* + \kappa_{n-t} z_1^{'} \boldsymbol{g}_{1,1}^*) + \varsigma_{n-t} \boldsymbol{b}_{n\cdot l+3}^* \\
&= (1, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})\vec{z}, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})n, \kappa_{n-t} z_1^{'}\tau(z_1 + \\
&\quad n), \varsigma_{n-t} + \vartheta, 0)_{\mathbb{B}^*} \\
&= (1, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})\vec{V} \cdot \Pi_1^*, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})n, \kappa_{n-t} z_1^{'}\tau(z_1 + \\
&\quad n), \varsigma_{n-t} + \vartheta, 0)_{\mathbb{B}^*} \\
&= (1, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})\vec{V}, (\psi_{n-t} + \kappa_{n-t}\delta z_1^{'})n, \kappa_{n-t}\tau(z_1 + \\
&\quad n)((\vec{U}^{(b)}, -1) \cdot \Pi_2) \cdot ((\vec{V}, d) \cdot \Pi_2^*)^T, \varsigma_{n-t} + \vartheta, 0)_{\mathbb{D}^*}.
\end{aligned}$$

Since $\psi, \psi_0, \ldots, \psi_{n-t}, \kappa, \kappa_0, \ldots, \kappa_{n-t}, \eta, \varsigma_0, \ldots, \varsigma_{n-t} \xleftarrow{U} \mathbb{F}_q$ are freshly generated, $\psi + \kappa\delta z_1^{'}, \psi_0 + \kappa_0\delta z_1^{'}, \ldots, \psi_{n-t} + \kappa_{n-t}\delta z_1^{'}$, $\kappa\tau$, $\eta + \gamma, \varsigma_0 + \vartheta, \ldots, \varsigma_{n-t} + \vartheta$ are uniformly and independently distributed. $\kappa\tau(z_1 + t), \kappa_0\tau(z_1 + t), \ldots, \kappa_{n-t}\tau(z_1 + n)$ are also uniformly and independently distributed if $\tau \neq 0$.

Note that $z_1^{'} = \vec{e}^{'} \cdot \vec{z}^T = ((\vec{U}^{(b)}, -1) \cdot \Pi_2) \cdot ((\vec{V}, d) \cdot \Pi_2^*)^T = (\vec{U}^{(b)}, -1) \cdot (\Pi_2 \cdot \Pi_2^*)^T \cdot (\vec{V}, d) = (\vec{U}^{(b)}, -1) \cdot (\vec{V}, d) = \vec{U}^{(b)} \cdot \vec{V} - d$.

1) For a matching key query $\vec{U}^{(b)} \cdot \vec{V} - d = 0$, the generated $\boldsymbol{k}_1^*, \boldsymbol{t}_0^*, \ldots, \boldsymbol{t}_{n-t}^*$ is normal. Therefore, the matching keys have the same distribution as that in Game 1.

2) For a non-matching key query, since $\vec{U}^{(b)} \cdot \vec{V} - d \neq 0$ and $\kappa\tau(z_1 + t), \kappa_1\tau(z_1 + t), \ldots, \kappa_{n-t+1}\tau(z_1 + n)$ are uniformly and independently distributed if $\tau \neq 0$, the generated $\boldsymbol{k}_1^*, \boldsymbol{t}_0^*, \ldots, \boldsymbol{t}_{n-t}^*$ are semi-functional. Therefore, the generated keys have the same distribution as that in Game 2 except with probability of $1/q$.

From Claim 1, 2 and 3, when $\beta = 0$, the advantage of $\mathcal{A}$ in the above game is equal to that in Game 0, i.e., $Adv_{\mathcal{A}}^{(0)}(\lambda)$, and is also equal to $\Pr[\mathcal{B}(1^\lambda, \varrho) \to 1 | \varrho \xleftarrow{R} \mathcal{G}_0^{P1}(1^\lambda, n, l, t)]$. Similarly, when $\beta = 1$, we see that the advantage of $\mathcal{A}$ in the above game is equal to that in Game 1, i.e., $Adv_{\mathcal{A}}^{(1)}(\lambda)$, and is also equal to $\Pr[\mathcal{B}(1^\lambda, \varrho) \to 1 | \varrho \xleftarrow{R} \mathcal{G}_1^{P1}(1^\lambda, n, l, t)]$. Therefore, $|Adv_{\mathcal{A}}^{(0)}(\lambda) - Adv_{\mathcal{A}}^{(1)}(\lambda)| \leq Adv_{\mathcal{B}}^{P1}(\lambda) + 1/q$. This completes the proof of Lemma 2.

Lemma 3: For any adversary $\mathcal{A}$, $|Adv_{\mathcal{A}}^{(1)}(\lambda) - Adv_{\mathcal{A}}^{(2)}(\lambda)| \leq 1/q$.

proof. To prove Lemma 3, we will show the distribution of public key, q_s queried private keys and challenge ciphertexts in Game 1 and that in Game 2 are indistinguishable. For that purpose, we define new dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*)$ of $\mathbb{V}$ as follows:

We generate $\omega_0, \omega_1, \theta \xleftarrow{U} \mathbb{F}_q$ and set $\vec{\omega} = \omega_0\vec{U}^{(0)} + \omega_1\vec{U}^{(1)}$, $\vec{\xi} = (\vec{\omega}, -\omega_0 - \omega_1)$, if $m^{(0)} = m^{(1)}, \theta = 0$, and

$$\begin{aligned}
\boldsymbol{d}_{n\cdot l+2} &= \boldsymbol{b}_{n\cdot l+2} - \theta \boldsymbol{b}_0 - \sum_{i=1}^{n\cdot l+1} \xi_i \boldsymbol{b}_i, \\
\boldsymbol{d}_0^* &= \boldsymbol{b}_0^* + \theta \boldsymbol{b}_{n\cdot l+2}^*, \\
\boldsymbol{d}_i^* &= \boldsymbol{b}_i^* + \xi_i \boldsymbol{b}_{n\cdot l+2}^*, i = 1, \ldots, n \cdot l + 1, \\
\vec{\boldsymbol{b}}_1 &= (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n\cdot l+1})^T, \\
\vec{\boldsymbol{b}}_1^* &= (\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_{n\cdot l+1}^*)^T, \\
\vec{\boldsymbol{d}}_1^* &= (\boldsymbol{d}_1^*, \ldots, \boldsymbol{d}_{n\cdot l+1}^*)^T.
\end{aligned}$$

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

That is,

$$\begin{pmatrix} \boldsymbol{b}_0 \\ \boldsymbol{b}_1 \\ \boldsymbol{d}_{n\cdot l+2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & I_{n\cdot l+1} & 0 \\ -\theta & -\vec{\xi} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{b}_0 \\ \boldsymbol{b}_1 \\ \boldsymbol{b}_{n\cdot l+2} \end{pmatrix}$$

$$\begin{pmatrix} \boldsymbol{d}_0^* \\ \boldsymbol{d}_1^* \\ \boldsymbol{b}_{n\cdot l+2}^* \end{pmatrix} = \begin{pmatrix} 1 & 0 & \theta \\ 0 & I_{n\cdot l+1} & \vec{\xi}^T \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{b}_0^* \\ \boldsymbol{b}_1^* \\ \boldsymbol{b}_{n\cdot l+2}^* \end{pmatrix}$$

We set $\mathbb{D} = (\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{n\cdot l+1}, \boldsymbol{d}_{n\cdot l+2}, \boldsymbol{b}_{n\cdot l+3}, \boldsymbol{b}_{n\cdot l+4})$, $\mathbb{D}^* = (\boldsymbol{d}_0^*, \ldots, \boldsymbol{d}_{n\cdot l+1}^*, \boldsymbol{b}_{n\cdot l+2}^*, \boldsymbol{b}_{n\cdot l+3}^*, \boldsymbol{b}_{n\cdot l+4}^*)$. We then easily verify that $\mathbb{D}$ and $\mathbb{D}^*$ are dual orthonormal. In the light of the adversary's view, both $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{D}, \mathbb{D}^*)$ are consistent with public key mpk. We define the following vectors $\vec{e}' = (1, 0, \ldots, 0) = (\vec{U}^{(0)}, -1) \cdot \Pi_3$, $\vec{e}' = (1, 0, \ldots, 0) = (\vec{U}^{(1)}, -1) \cdot \Pi_4$, where $\Pi_3, \Pi_4 \in GL(n \cdot l + 1, \mathbb{F}_q)$ and $\vec{y} = (y_1, \ldots, y_{n\cdot l+1}) = (\vec{V}, d) \cdot (\Pi_3^T)^{-1}$, $\vec{y}' = (y_1', \ldots, y_{n\cdot l+1}') = (\vec{V}, d) \cdot (\Pi_4^T)^{-1}$ similar with Game 1. We also implicitly set $-\sigma^{(h)}((\vec{V}^{(h)}, t) \cdot \vec{\xi}) = \iota(y_1 + y_1')$, $-r_0^{(h)}((\vec{V}^{(h)}, t) \cdot \vec{\xi}) = \iota_0(y_1 + y_1')$, …, $-r_{n-t}^{(h)}((\vec{V}^{(h)}, n) \cdot \vec{\xi}) = \iota_{n-t}(y_1 + y_1')$ and $(y_1 + y_1') = 0$ during key queries, where $\sigma^{(h)}, r_0^{(h)}, \ldots, r_{n-t}^{(h)}, \iota, \iota_0, \ldots, \iota_{n-t} \xleftarrow{U} \mathbb{F}_q$ and $h \in \{1, \ldots q_s\}$.

Claim 4: In Case 1, queried keys and challenge ciphertext can be expressed as in two ways, in Game 1 over bases $(\mathbb{B}, \mathbb{B}^*)$ and in Game 2 over bases $(\mathbb{D}, \mathbb{D}^*)$ except with probability $1/q$.

proof. In Case 1, only non-matching key queries are allowed for the adversary. $(\{\boldsymbol{k}_1^{(h)*}\}_{h=1,\ldots,q_s}, \{\boldsymbol{t}_0^{(h)*}\}_{h=1,\ldots,q_s}, \ldots, \{\boldsymbol{t}_{n-t+1}^{(h)*}\}_{h=1,\ldots,q_s}, c_1, c_2, c_3)$ in Game 1 and in Game 2 respectively are expressed over bases $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{D}, \mathbb{D}^*)$ as:

$$\begin{aligned} \boldsymbol{k}_1^{(h)*} &= (0, \sigma^{(h)}\vec{V}^{(h)}, \sigma^{(h)}t, \kappa^{(h)}, \eta^{(h)}, 0)_{\mathbb{B}^*} \\ &= (0, \sigma^{(h)}\vec{V}^{(h)}, \sigma^{(h)}t, \tilde{\kappa}^{(h)}, \eta^{(h)}, 0)_{\mathbb{D}^*} \end{aligned}$$

where $\tilde{\kappa}^{(h)} = \kappa^{(h)} - \sigma^{(h)}((\vec{V}^{(h)}, t) \cdot \vec{\xi})$,

$$\begin{aligned} \boldsymbol{t}_0^{(h)*} &= (1, r_0^{(h)}\vec{V}^{(h)}, r_0^{(h)}t, \phi_0^{(h)}, \varsigma_0^{(h)}, 0)_{\mathbb{B}^*} \\ &= (1, r_0\vec{V}^{(h)}, r_0^{(h)}t, \tilde{\phi_0}^{(h)}, \varsigma_0^{(h)}, 0)_{\mathbb{D}^*}, \end{aligned}$$

where $\tilde{\phi_0}^{(h)} = \phi_0^{(h)} - \theta - r_0^{(h)}((\vec{V}^{(h)}, t) \cdot \vec{\xi})$,

$$\vdots$$

$$\begin{aligned} \boldsymbol{t}_{n-t}^{(h)*} &= (1, r_{n-t}^{(h)}\vec{V}^{(h)}, r_{n-t}^{(h)}n, \phi_{n-t}^{(h)}, \varsigma_{n-t}^{(h)}, 0)_{\mathbb{B}^*} \\ &= (1, r_{n-t}\vec{V}^{(h)}, r_{n-t}^{(h)}n, \tilde{\phi}_{n-t}^{(h)}, \varsigma_{n-t}^{(h)}, 0)_{\mathbb{D}^*}, \end{aligned}$$

where $\tilde{\phi}_{n-t}^{(h)} = \phi_{n-t}^{(h)} - \theta - r_{n-t}^{(h)}((\vec{V}^{(h)}, n) \cdot \vec{\xi})$.

$$\begin{aligned} c_1 &= (\zeta, \omega\vec{U}^{(b)}, -\omega, \rho, 0, \varphi)_{\mathbb{B}} \\ &= (\zeta', \omega\vec{U}^{(b)} + \rho(\omega_0\vec{U}^{(0)} + \omega_1\vec{U}^{(1)}), -\omega - \rho(\omega_0 + \omega_1), \\ &\quad \rho, 0, \varphi)_{\mathbb{D}} \\ &= (\zeta', (\omega + \rho\omega_b)\vec{U}^{(b)} + \rho\omega_{1-b}\vec{U}^{(1-b)}, -\omega - \rho(\omega_0 + \omega_1), \\ &\quad \rho, 0, \varphi)_{\mathbb{D}}, c_2 = g_T^{\zeta} m^{(b)}, c_3 = g^{\omega}, \end{aligned}$$

where $\zeta' = \zeta + \rho\theta$ is uniformly and independently distributed since $\theta \xleftarrow{U} \mathbb{F}_q$ and if $\rho \neq 0, (\omega + \rho\omega_b)\vec{U}^{(b)} + \rho\omega_{1-b}\vec{U}^{(1-b)}$ is also uniformly and independently distributed in $\mathrm{span}\left\langle \vec{U}^{(0)}, \vec{U}^{(1)} \right\rangle$ since $\omega_0, \omega_1 \xleftarrow{U} \mathbb{F}_q$.

Claim 5: In Case 2, keys and the challenge ciphertext can be expressed as in two ways, in Game 1 over bases $(\mathbb{B}, \mathbb{B}^*)$ and in Game 2 over bases $(\mathbb{D}, \mathbb{D}^*)$ except with probability $1/q$.

proof. In Case 2, both non-matching and matching key queries are allowed for the adversary.

1) Non-matching queried keys are expressed over $(\mathbb{B}, \mathbb{B}^*)$ in Game 1 and $(\mathbb{D}, \mathbb{D}^*)$ in Game 2 as in Case 1, where $\tilde{\phi_0}^{(h)} = \phi_0^{(h)} - r_0^{(h)}((\vec{V}^{(h)}, t) \cdot \vec{\xi}), \ldots, \tilde{\phi}_{n-t}^{(h)} = \phi_{n-t}^{(h)} - r_{n-t}^{(h)}((\vec{V}^{(h)}, n) \cdot \vec{\xi})$.

2) Matching keys are expressed over bases $(\mathbb{B}, \mathbb{B}^*)$ in Game 1 and $(\mathbb{D}, \mathbb{D}^*)$ in Game 2 as follows.

$$\begin{aligned} \boldsymbol{k}_1^{(h)*} &= (0, \sigma^{(h)}\vec{V}^{(h)}, \sigma^{(h)}t, 0, \eta^{(h)}, 0)_{\mathbb{B}^*} \\ &= (0, \sigma^{(h)}\vec{V}^{(h)}, \sigma^{(h)}t, \tilde{\kappa}^{(h)}, \eta^{(h)}, 0)_{\mathbb{D}^*}, \end{aligned}$$

where

$$\begin{aligned} \tilde{\kappa}^{(h)} &= -\sigma^{(h)}((\vec{V}^{(h)}, t) \cdot \vec{\xi}) = \iota(y_1 + y_1') = \iota(\vec{e}' \cdot \vec{y}^T + \vec{e}' \cdot \vec{y}'^T) \\ &= \iota((\vec{U}^{(0)}, -1) \cdot \Pi_3 \cdot (\vec{V}, d) \cdot \Pi_3^{*T} + (\vec{U}^{(1)}, -1) \cdot \Pi_4 \cdot (\vec{V}, d) \cdot \Pi_4^{*T}) \\ &= \iota((\vec{U}^{(0)}, -1) \cdot (\vec{V}, d) + (\vec{U}^{(1)}, -1) \cdot (\vec{V}, d)) \\ &= \iota(\vec{U}^{(0)} \cdot \vec{V} - d + \vec{U}^{(1)} \cdot \vec{V} - d) = 0, \end{aligned}$$

$$\begin{aligned} \boldsymbol{t}_0^{(h)*} &= (1, r_0^{(h)}\vec{V}^{(h)}, r_0^{(h)}t, 0, \varsigma_0^{(h)}, 0)_{\mathbb{B}^*} \\ &= (1, r_0^{(h)}\vec{V}^{(h)}, r_0^{(h)}t, \tilde{\phi_1}^{(h)}, \varsigma_0^{(h)}, 0)_{\mathbb{D}^*} \end{aligned}$$

where $\tilde{\phi_0}^{(h)} = -r_0^{(h)}((\vec{V}^{(h)}, t) \cdot \vec{\xi}) = \iota_0(y_1 + y_1') = 0$.

$$\vdots$$

$$\begin{aligned} \boldsymbol{t}_{n-t}^{(h)*} &= (1, r_{n-t}^{(h)}\vec{V}^{(h)}, r_{n-t}^{(h)}n, 0, \varsigma_{n-t}^{(h)}, 0)_{\mathbb{B}^*} \\ &= (1, r_{n-t}^{(h)}\vec{V}^{(h)}, r_{n-t}^{(h)}n, \tilde{\phi}_{n-t}^{(h)}, \varsigma_{n-t}^{(h)}, 0)_{\mathbb{D}^*}, \end{aligned}$$

where $\tilde{\phi}_{n-t}^{(h)} = -r_{n-t}^{(h)}((\vec{V}^{(h)}, n) \cdot \vec{\xi}) = \iota_{n-t}(y_1 + y_1') = 0$.

As for ciphertext, since $\zeta' = \zeta + \rho\theta$ and $\theta = 0$, we have

$$\begin{aligned} c_1 &= (\zeta, \omega\vec{U}^{(b)} + \rho(\omega_0\vec{U}^{(0)} + \omega_1\vec{U}^{(1)}), -\omega - \rho(\omega_0 + \omega_1), \rho, 0, \varphi)_{\mathbb{D}} \\ &= (\zeta, (\omega + \rho\omega_b)\vec{U}^{(b)} + \rho\omega_{1-b}\vec{U}^{(1-b)}, -\omega - \rho(\omega_0 + \omega_1), \rho, 0, \varphi)_{\mathbb{D}}, \end{aligned}$$

$c_2 = g_T^{\zeta} m$, $c_3 = g^{\omega}$, where $(\omega + \rho\omega_b)\vec{U}^{(b)} + \rho\omega_{1-b}\vec{U}^{(1-b)}$ are uniformly and independently distributed in $\mathrm{span}\left\langle \vec{U}^{(0)}, \vec{U}^{(1)} \right\rangle$ since $\omega_0, \omega_1 \xleftarrow{U} \mathbb{F}_q$, if $\rho = 0$ except with probability $1/q$.

Thus, Game 1 can be conceptually changed to Game 2. This completes the proof of Lemma 3.

Lemma 4: For any adversary $\mathcal{A}$, $Adv_{\mathcal{A}}^{(2)}(\lambda) = 0$.

proof. The value of b is independent from the adversary's view in Game 2. Hence, $Adv_{\mathcal{A}}^{(2)}(\lambda) = 0$.

VII. Performance analysis

In this section, we first implement our scheme to test the time consumption of each algorithm affected by three main factors including the length of strings n, the size of universes l and the threshold t. Our experiments are conducted on Visual Studio 2012 of a PC with Intel Core i7-8700 CPU and 16GB RAM running 64-bit Windows 10 Profession. We write the C++ code by utilizing GNU Multiple Precision Arithmetic

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

(GMP) [19] and Pairing-Based Cryptography (PBC) libraries [20]. The experiment utilizes a 160-bit elliptic curve group built on a curve $y^2 = x^3 + x$ over the filed $\mathbb{F}_q$, where $q = 512$ bits. Second, we provide a theoretical analysis and comparison between the existing fuzzy encryption schemes and ours in terms of performance and security.

A. Implementation

Set $l = 10, t = 10$, Fig. 2 (a) shows the time cost of each algorithm with n ranging from 10 to 50. As we can see from this figure, the time consumption in KeyGen algorithm showed a quadratic growth with n increasing since there are $n - t + 2$ key vectors to be generated in this phase and each of them is $(nl + 5)$ – dimensional. What's more, the trend of time cost for the KeyGen algorithm is changeable significantly with n increasing since the computation of $n - t + 2$ key vectors is dominant during the whole execution in order to realize fuzzy functionality. As for the other three algorithms, the time cost is linear with n.

Set $n = 10, t = 10$, Fig. 2 (b) illustrates the time cost of all algorithms is linear with l where l is ranged from 10 to 50. The larger the size of the universe is, the more time costs are required. As we can see from this figure, although Setup algorithm is the most costly, this algorithm can be preprocessed and thus save the time cost to some extent. When $l = 50$, the time cost of each algorithm is about 1.681 seconds, 0.315 seconds, 0.176 seconds and 1.526 seconds, respectively.

Set $n = 50, l = 10$, Fig. 2 (c) shows the time cost of four algorithms with t ranging from 10 to 50. Most significantly, we can see from this figure that the time consumption of KeyGen algorithm will decrease with the increase of t. The larger t is, the smaller the gap between n and t becomes, then the fewer vectors are required in KeyGen algorithm. As a result, the time cost will become less. Additionally, the time cost of Setup and Enc algorithms are both independent on t, which will be optimal in practice in some scenarios, especially with a large threshold. As for Dec algorithm, the time cost to build a list will be linear with the threshold.

B. Comparison

To further clarify the performance, we also provide a theoretical performance comparison between the existing fuzzy encryption schemes [1], [2], [3], [4], [5], [6] and ours since they all are designed under the public-key setting and all of them support fuzziness based on a threshold but with different similarity metrics. However, we omit the literature [6] from Tab.1 since it focuses on how to design a fuzzy exactor from a general approach and thus falls into another computational architecture different from ours. From Tab. 1, we can see that the fuzzy encryption schemes [1], [2], [3], [4], [5] only achieve payload-hiding security without protecting attributes.

In contrast to the existing fuzzy attribute-hiding scheme [7] closest to ours, which was designed under a composite-order group, our scheme is constructed under a prime-order group. The parameter size and the pairing operations of composite-order group are much larger (between 10 to 24 times) and more expensive (between 10 to 192 times slower) than that in a same scale of prime-order group [21], [22]. On the other hand, [7] requires testing all possible inner products of two vectors no less than a threshold in order to decrypt successfully and therefore the computational complexity of decryption algorithm is exponential with the threshold. However, the computational complexity of our decryption algorithm is linear only with the threshold by taking advantage of the idea of privacy-preserving mapping as we addressed in contribution. Due to these two main reasons, our scheme achieves a higher efficiency. Additionally, it is remarked that the fuzzy identity-based encryption [7] can only be conducted in a binary domain while ours works in an integer domain, and what's worse, their technique is not available for an integer domain if we transform an integer string into a binary string straightforwardly.

VIII. Applications

In this section, we demonstrate two application examples of the proposed AHFE scheme in some scenarios with a large threshold. One is how to design fuzzy keyword search encryption (FKSE) based on a slightly modified version of AHFE and the other is how to realize attribute-hiding closest substring encryption (CSE) leveraging our AHFE. To this end, for the construction of FKSE, we modify the string encoding in Section 5.1 to suit the encoding of an integer keyword and as for CSE scheme, it can be achieved from our construction in Section 5.2 straightforwardly.

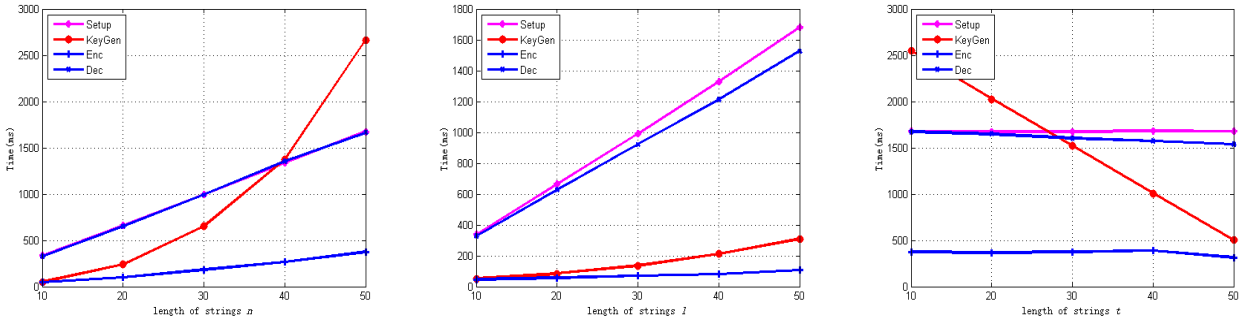
A. Fuzzy keyword search encrytpion

FKSE in this paper refers to a searchable encryption system supporting such a query whether an overlap distance between two integer keywords sets W and Q is no less than a threshold t. Our FKSE will be practical for some applications in real life with a large threshold, especially where the threshold t is as close to the size of universe l as possible. For example, in the *closest keyword search* of FKSE, it will fast search the target file if and only if an overlap distance between two integer keyword sets is $l - 1$, namely they differ in only one keyword. In such an application, we see that a fast search is desirable in PKSE for a practical reason. Therefore, we develop a new encoding and slightly modify our AHFE scheme to conduct this application since in our scheme the computational complexity of decryption is linear with the threshold.

Now, we define a universe $U = \{1, \ldots, l\}$ and l is a positive integer. Let $W = \{w_1, w_2, \ldots, w_{n_1}\}$ and $Q = \{q_1, q_2, \ldots, q_{n_2}\}$ be two integer keyword sets. According to the following rule

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198



(a) (b) (c)

Fig. 4. Implementation result

Tab. I. Performance comparison

	Ciphertext size	Key size	Decryption cost	Security	String domain	Similarity metric
[1]	$(n+1)\|G\| + \|G_T\|$	$2\|G\|$	$n\|G\| + 2e$	Payload	Integer	Mahalanobis distance
[2]	$(n_1+1)\|G\| + \|G_T\|$	$n_2\|G\|$	$(d+2)e$	Payload	Integer	Overlap distance
[3]	$(n_1+l^2+3)\|G^c\| + \|G_T^c\|$	$(n_2+l^2+2)\|G^c\|$	$((n_1+1)(n_2+1)+\tau^2)e^c$	Payload	Integer	Edit distance
[4]	$(2n+4)\|G\|$	$\|G_T\| + \|Z_q\|$	$2ne + n\|G\|$	Payload	Integer	Edit distance
[5]	$(L+2)\|G\| + 2\|Z_q\|$	$\|Z_q\|$	$2e$	Payload	Binary	Designate position
[7]	$(2\sum_{i=0}^{t_{max}}\binom{n}{i}+3)\|G^c\|$	$(2\sum_{i=0}^{t+1}\binom{n}{i}+3)\|G^c\|$	$(2\sum_{i=0}^{t+1}\binom{n}{i}+3)e^c$	Attribute	Binary	Hamming distance
Ours	$(nl+6)\|G\| + n\|G_T\|$	$(n-t+2)(nl+5)\|G\| + (n-t+1)\|G\|$	$2(n-t+1)e + 2(nl+5)e$	Attribute	Integer	Overlap distance

Notes: $|G|$ is the size of a prime-order group G. $|Z_q|$ is the size of a element in Z_q. $|G^c|$ and $|G_T^c|$ denote the size of a composite-order group G^c and a pairing group with composite-order G_T^c, respectively. e and e^c is the bilinear map with a prime-order group and a composite-order group, respectively. n_1 and n_2 are the dimensions of two different strings. τ is the edit distance between two strings. d is the overlap distance between two strings. t_{max} is the maximum of threshold. L is the maximum size of the wildcard set that the ciphertext allows.

II, W and Q can be encoded two l – dimensional vectors $\vec{U}$ and $\vec{V}$ as:

$$\vec{U}_{\{w_1,w_2\ldots w_{n_1}\}} = \begin{cases} u_i = 1, & if \quad i \in W, \quad i = 1,\ldots,l \\ u_i = 0, & \text{otherwise} \end{cases}$$

$$\vec{V}_{\{q_1,q_2,\ldots,q_{n_2}\}} = \begin{cases} v_i = 1, & if \quad i \in Q, \quad i = 1,\ldots,l \\ v_i = 0, & \text{otherwise} \end{cases} \qquad II$$

Now, we show how to design a new FKSE scheme from our AFHE. The new searchable scheme consists of four algorithms: Setup, Trapdoor, PEKS and Test. However, there are three main differences when we modify our AFHE suitable to a new FKES. First, according to the rule *II*, in the new scheme the dimension of encoding vectors $\vec{U}$ and $\vec{V}$ is l rather than $n \cdot l$ in our AFHE and therefore the inputs of Setup algorithm in the new FKES scheme only need a security parameter λ and the size of universe l. Second, a searchable encryption system stresses “Test" capability instead of “Decrypt" capability. Therefore, there is no need to use a private key to decrypt a ciphertext and output a message in a searchable system. As a result, we cancel out the ciphertext c_2 in encryption stage, part of decryption keys from $\boldsymbol{t}_0^*$ to $\boldsymbol{t}_{n-t}^*$, and step 3 in decryption period from the original AHFE scheme. Also, the PEKS algorithm removes the input of m since the new FKES only needs the trapdoor $\boldsymbol{k}_1^*, \mathcal{L}_1$ and the ciphertext $c_1, c_3, \mathcal{L}_2$ to do test functionality. At last, since the upper bound of overlap distance between two keyword sets is the size of universe l, one input of Dec algorithm in our AFHE n will be replaced with the size of universe l in the Test algorithm.

- **Setup**$(1^\lambda, l)$: Taking as input a security parameter λ and the size of the universe l, the setup algorithm computes
$$(mpk, msk) \longleftarrow AHFE.Setup(1^\lambda, l)$$
and outputs a key pair (mpk, msk).
- **Trapdoor**(mpk, msk, Q, t): Taking as input the key pair (mpk, msk), a keyword set Q and a threshold t, the trapdoor algorithm first transforms Q into $\vec{V}$ and then computes
$$T_Q = \{\boldsymbol{k}_1^*, \mathcal{L}_1\} \longleftarrow AHFE.KeyGen(mpk, msk, \vec{V}, t)$$
- **PEKS**(mpk, W): Taking as input the public key mpk, a keyword set W, the PEKS algorithm first transforms W into $\vec{U}$ and then computes:
$$CT_W = \{c_1, c_3, \mathcal{L}_2\} \longleftarrow AHFE.Enc(mpk, \vec{U})$$
- **Test**(mpk, T_Q, CT_W, l, t): Taking as input the public key mpk, the ciphertext CT_W, the token T_Q, the size of an universe l and the threshold t, the test algorithm computes
$$'Flag' \longleftarrow AFHE.Dec(mpk, T_Q, CT_W, l, t)$$
Test algorithm outputs $'Flag = 1'$, if the overlap distance between W and Q is no less than threshold t, there exists one item in $\mathcal{T}$ which must fall into the list $\mathcal{L}_2$ and the search is succesful; otherwise, outputs $'Flag = 0'$.

B. Attribute-hiding closest substring encryption

CSE is defined as one that ciphertext associated with a string S can be decrypted with a private key associated with another

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

string S', if the overlap distance between substrings in S and substrings in S' is no less than a threshold t'. Unfortunately, the existing closest substring encryption [2] is payload-hiding as described in the Introduction. How to design an attribute-hiding closest substring encryption is challenging. Here, we present a new CSE scheme leveraging our AHFE after the original problem is modified slightly, which is shown as follows. Our CSE is practical in some scenarios especially with a large threshold, such as facial recognition, DNA matching and other applications with highly approximate matching requests.

Let U be a universe and the size of U be l. Set $S = \{s_1 s_2 \dots s_{n_1}\}$ and $S^{'} = \{s'_1 s'_2 \dots s'_{n_2}\}(n_1 \leq n_2)$. In order to realize closest substring matching between S and S', both will be first transformed two $(n_2 - n_1 + 1)$ length strings as $S_1 = \{s_1 \dots s_{n_1}, s_1 \dots s_{n_1}, \dots, s_1 \dots s_{n_1}\}$ and $S_1^{'} = \{s_1^{'} \dots s_{n_1}^{'}, s_2^{'} \dots s_{n_1+1}^{'}, \dots, s_{n_2-n_1+1}^{'} \dots, s_{n_2}^{'}\}$, and S_1, S'_1 will be further encoded into two $(n \cdot l)$ – dimensional vectors $\vec{U}$ and $\vec{V}$ accoding to rule I in Section 5.1, where $n = n_1(n_2 - n_1 + 1)$, $t = n_1 t'$ and l ramains unchanged.

IX. Conclusion

In this paper, we propose a new attribute-hiding fuzzy encryption scheme under a public-key setting for privacy-preserving data evaluation, which is able to work in an integer domain with higher efficiency. Compared with the existing fuzzy encryption schemes, our proposal achieves a three-fold functionality simultaneously: *fuzziness*, *attribute-hiding* and *efficiency*. Finally, as two application examples, we show how our AHFE scheme can be used to realize fuzzy keyword search encryption and attribute-hiding closest substring encryption.

More interesting, our effort in this work focused on the IBE system which inherently relies on a fully trusted third party. However, locating such a completely trusted third party proves difficult in real-world scenarios sometimes. Therefore, the challenging problem is designing more attribute-hiding fuzzy encryption schemes supporting different metrics but in a certificateless encryption system or a decentralized ABE one. All these will be left for our future work.

References

[1] Guo F, Susilo W, Mu Y. Distance-based encryption: How to embed fuzziness in biometric-based encryption[J]. IEEE Transactions on Information Forensics and Security, 2015, 11(2): 247-257.

[2] Guo F, Susilo W, Mu Y. Generalized closest substring encryption[J]. Designs, Codes and Cryptography, 2016, 80(1): 103-124.

[3] Phuong T V X, Yang G, Susilo W, et al. Edit distance based encryption and its application[C]//Australasian Conference on Information Security and Privacy. Springer, Cham, 2016: 103-119.

[4] Wang Y, Huang Q, Li H, et al. Public key encryption with fuzzy matching[C]//Provable and Practical Security: 15th International Conference, ProvSec 2021, Guangzhou, China, November 5–8, 2021, Proceedings. Cham: Springer International Publishing, 2021: 39-62.

[5] Zhao Z Z, Guo F, Wu G, et al. Secure Infectious Diseases Detection System With IoT-Based e-Health Platforms[J]. IEEE Internet of Things Journal, 2022, 9(22): 22595-22607.

[6] Dodis Y, Reyzin L, Smith A. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data[C]//Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004. Proceedings 23. Springer Berlin Heidelberg, 2004: 523-540.

[7] Cheung D W, Mamoulis N, Wong W K, et al. Anonymous fuzzy identity-based encryption for similarity search[C]//International Symposium on Algorithms and Computation. Springer, Berlin, Heidelberg, 2010: 61-72.

[8] Okamoto T, Takashima K. Efficient (hierarchical) inner-product encryption tightly reduced from the decisional linear assumption[J]. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 2013, 96(1): 42-52.

[9] Tang Q. Privacy preserving mapping schemes supporting comparison[C]//Proceedings of the 2010 ACM workshop on Cloud computing security workshop. 2010: 53-58.

[10] Sahai A, Waters B. Fuzzy identity-based encryption[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2005: 457-473.

[11] Shamir A. Identity-based cryptosystems and signature schemes[C]//Workshop on the theory and application of cryptographic techniques. Springer, Berlin, Heidelberg, 1984: 47-53.

[12] Al-Riyami S S, Paterson K G. Certificateless public key cryptography[C]//International conference on the theory and application of cryptology and information security. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003: 452-473.

[13] Lewko A, Waters B. Decentralizing attribute-based encryption[C]//Annual international conference on the theory and applications of cryptographic techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011: 568-588.

[14] Boneh D, Waters B. Conjunctive, subset, and range queries on encrypted data[C]//Theory of Cryptography: 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007. Proceedings 4. Springer Berlin Heidelberg, 2007: 535-554.

[15] Katz J, Sahai A, Waters B. Predicate encryption supporting disjunctions, polynomial equations, and inner products[C]//annual international conference on the theory and applications of cryptographic techniques. Springer, Berlin, Heidelberg, 2008: 146-162.

[16] Park J H. Inner-product encryption under standard assumptions[J]. Designs, Codes and Cryptography, 2011, 58(3): 235-257.

[17] Boneh D, Boyen X, Shacham H. Short group signatures[C]//Annual international cryptology conference. Springer, Berlin, Heidelberg, 2004: 41-55.

[18] Waters B. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions[C]//Annual International Cryptology Conference. Springer, Berlin, Heidelberg, 2009: 619-636.

[19] Granlund T. The GNU multiple precision arithmetic library[J]. http://gmplib. org/, 2010.

[20] Lynn B. The pairing-based cryptography library[J]. Internet: crypto. stanford.edu/pbc/[Mar.27,2013], 2006.

[21] Freeman D M. Converting pairing-based cryptosystems from composite-order groups to prime-order groups[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2010: 44-61.

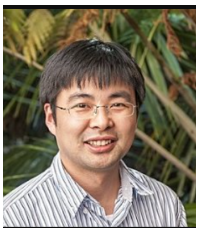
[22] Guillevic A. Comparing the pairing efficiency over composite-order and prime-order elliptic curves[C]//International Conference on Applied Cryptography and Network Security. Springer, Berlin, Heidelberg, 2013: 357-372.

Zhenhua Chen received her Ph. D. degree from Shaanxi Normal University in 2014. She is currently an associate professor with School of Computer Science and Technology, Xi'an University of Science and Technology. Her research interests include public-key cryptography, secure multi-party computation, and secret sharing.

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3376198

Luqi Huang received the M.S. degree from the Xi'an University of Science and Technology, China. She is currently working toward the Ph.D. degree in the University of Wollongong, Australia. Her current research interests include public key cryptography and information security.



Guomin Yang received the PhD degree from the Computer Science Department, City University of Hong Kong in 2009. Formerly, he worked as a Research Scientist in the Temasek Laboratories at National University of Singapore. He is currently an associate professor at the School of Computing and Information Systems, Singapore Management University. In 2015, he was awarded a prestigious Australian Research Council DECRA fellowship. His research interests include cryptography and network security.

Willy Susilo is a distinguished professor with the School of Computing and Information Technology, Faculty of Engineering and Information Sciences, University of Wollongong (UOW), Australia. He is the director of Institute of Cybersecurity and Cryptology, School of Computing and Information Technology, UOW and the head of School of Computing and Information Technology at UOW (2015–now). Prior to this role, he was awarded the prestigious Australian Research Council Future Fellowship in 2009. In 2016, he was awarded the "researcher of the Year at UOW, due to his research excellence and contributions. He is the editor-in-chief of the Elsevier's Computers Standards and Interface and the MDPIs Information journal. He is currently an associate editor of the IEEE Transactions on Dependable and Secure Computing. He has also served as the program committee member of several international conferences.

Xingbing Fu is a lecturer, and he received the Ph.D. degree from University of Electronic Science and Technology of China (UESTC) in 2016. His research interests include cloud computing and cryptography.

Xingxing Jia received the B.S. and Ph.D. degrees from the School of Mathematics and Statistics, Lanzhou University, Lanzhou, China, in 2004 and 2010, respectively. She is currently a lecturer with the School of Mathematics and Statistics, Lanzhou University. Her research interests include secret sharing and visual cryptography.

© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.