1-1-2015

Generalized closest substring encryption

Fuchun Guo
University of Wollongong, fuchun@uow.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Yi Mu
University of Wollongong, ymu@uow.edu.au

Generalized closest substring encryption

Abstract

We propose a new cryptographic notion called generalized closest substring encryption. In this notion, a ciphertext encrypted with a string S can be decrypted with a private key of another string S', if there exist a substring of S, i.e. S^, and a substring of S', i.e. S^', that are "close" to each other measured by their "overlap distance". The overlap distance between S^ and S^' is the number of identical positions at which the corresponding symbols are the same. In comparison with other encryption systems, the closest notion is the Fuzzy-IBE proposed by Sahai and Waters. The main difference is that the Fuzzy-IBE measures the overlap distance between S and S', while ours measures the overlap distance of all of their substrings (including the complete string), and we take the maximum value among those. The overlap distance between their substrings will measure the similarity of S and S' more precisely compared to the overlap distance between the two complete strings. We note that embedding this overlap distance in an encryption is a challenging task, in particular in order to achieve a practical scheme. Therefore, we invent a new approach to develop a practical generalized closest substring encryption system. The novelty of our approach relies on the way we generate ciphertext and private key representing the complete string so that they can still measure the overlap distance of substrings. The size of ciphertext and private key grow linearly only in the length of the input string. We prove the security in the selective model under a generalization of decision q-Bilinear Diffie-Hellman Exponent assumption.

Disciplines

Engineering | Science and Technology Studies

Publication Details

Guo, F., Susilo, W. & Mu, Y. (2015). Generalized closest substring encryption. Designs, Codes and Cryptography, Online First 1-22.

Generalized Closest Substring Encryption

Abstract.

We propose a new cryptographic notion called a generalized closest substring encryption. In this notion, a ciphertext encrypted with a string S can be decrypted with a private key of another string S', if there exist a substring of S, i.e. $\hat{S}$, and a substring of S', i.e. $\hat{S}'$, that are *"close"* to each other measured by their "overlap distance". The overlap distance between $\hat{S}$ and $\hat{S}'$ is the number of identical positions at which the corresponding symbols are the same. In comparison with other encryption systems, the closest notion is the Fuzzy-IBE proposed by Sahai and Waters. The main difference is that the fuzzy-IBE measures the overlap distance between S and S', while ours measures the overlap distance of all of their substrings (including the complete string), and we take the maximum value among those. The overlap distance between their substrings will measure the similarity of S and S' more precisely compared to the overlap distance between the two complete strings. We note that embedding this overlap distance in an encryption is a challenging task, in particular in order to achieve a practical scheme. Therefore, we invent a new approach to develop a practical generalized closest substring encryption system. The novelty of our approach relies on the way we generate ciphertext and private key representing the complete string so that they can still measure the overlap distance of substrings. The size of ciphertext and private key grow linearly only in the length of the input string. We prove the security in the selective model under a generalization of decision q-Bilinear Diffie-Hellman Exponent (BDHE) assumption.

Keywords: Encryption, Closest Substring Problem

1 Introduction

Pattern matching problem has been primarily used in many computational biology applications. In this work, we investigate an important class of pattern matching problem, namely the closest substring problem in the context of encryption. The closest substring problem [21] is formalized as follows. Given k strings $S_1, S_2, \cdots, S_k$ over an alphabet Σ and integers L, d_H, we are to find a string S (of length L) and a substring of each S_i (of the same length L), denoted as $\hat{S}_i$, such that the Hamming distance between S and each $\hat{S}_i$ is *at most* d_H. That is, the number of identical positions at which the corresponding symbols are the same is at least d, where $d = L - d_H$. The value d is known as the "overlap distance" (which is the similar notion used in [30]). The closest substring problem has been found very useful in bioinformatics, such as primer design [20, 27], genetic probe design [18], antisense drug design [11], and motif finding [10], as it can be used to measure the similarity between two strings more precisely via their substrings compared to other measurements like measuring the overlap distance between two complete strings.

In this work, we are motivated to equip an encryption with a closest substring measurement, which is a generalization of the closest substring problem. Informally speaking, an encryptor constructs a ciphertext by encrypting a message using a string S and broadcasts it to all recipients. A recipient of the ciphertext can decrypt it with a private key of string S', if and only if the overlap distance between S and one of the substrings of S' is at least d. A potential application scenario of this encryption exists in bioinformatics. The string S represents a DNA pattern associated with some disease or some specific pattern, while S' is an individual's DNA sequence. A sensitive message is encrypted with the string S and the encryptor wants to ensure that only the owner of the DNA sequence S' that matches the pattern S under the closest substring measurement can decrypt the ciphertext.

To measure similarity of the two strings more precisely, we consider the measurement on their substrings. Subsequently, we generalize the decryption condition where the ciphertext can be decrypted if and only if the overlap distance between one of the substrings of S and one of the substrings of S' is at least d, and hence, the notion of *generalized closest substring encryption*. This generalization allows us to acquire substrings with a larger overlap distance compared to the closest substring problem between a string and a substring. As an example, consider two strings[1] $S = ATCGT$ and $S' = TCGTATGGA$. We find that

[1] The strings used in this example are taken from letters used in DNA molecules, and therefore $\Sigma = \{G, A, T, C\}$, where they denote **G**uanine, **A**denine, **T**hymine and **C**ytosine, respectively.

the overlap distance between S and a substring of S' is at most three (the substring of S' is $ATGGA$, and hence, the overlap distance is obtained from $\mathbf{AT}C\mathbf{G}T$ and $\mathbf{AT}G\mathbf{G}A$), but the overlap distance between a substring of S (i.e, $\mathbf{TCGT}$) and a substring of S' (i.e, $\mathbf{TCGT}$) is four.

The closest cryptographic primitive to the generalized closest substring encryption is the fuzzy identity-based encryption (IBE) [30] introduced by Sahai and Waters. Fuzzy-IBE allows a ciphertext encrypted with a string S to be decrypted with a private key of string S', if and only if the overlap distance between S and S' is at least d. We notice that this distance measurement is not suitable for distance measurement between substrings. In the example above, when we use the definition of an overlap distance between $S = ATCGT$ and $S' = TCGTATGGA$ as defined by Sahai and Waters [30], we will obtain zero and then judge these two strings as irrelevant. This is because the comparison is performed between the string $ATCGT$ and $TCGTA$ only. In the following table, we list the comparison of decryption conditions among fuzzy-IBE, closest substring encryption (CSE) and generalized closest substring encryption (GCSE).

Encryption Notions	**Decryption Conditions** $d \leq d_X$
Fuzzy IBE	$d_F =$ the overlap distance between S and S'.
CSE	$d_C =$ the maximum value of overlap distance between S and a substring of S'.
GCSE	$d_G =$ the maximum value of overlap distance between a substring of S and a substring of S'.

Table 1. Various encryption notions. Here, we assume a ciphertext is encrypted with a string S and we want to decrypt the ciphertext using a private key of another string S', where d denotes a threshold value and d_X is the defined distance in the corresponding encryption notion. As an example, let $S = ATCGT$ and $S' = TCGTATGGA$. We have $d_F = 0, d_C = 3$ and $d_G = 4$. The definition of d_G measures the similarity of S and S' more precisely than the other two definitions.

Trivial Construction A trivial construction of GCSE can be obtained by modifying fuzzy-IBE. In comparison with fuzzy-IBE, the difficulty of constructing a GCSE system is the fact that we do not know which two substrings will produce the overlap distance larger than the threshold value until both strings are provided. To address this problem, we can split the string into many independent strings. Suppose n is the length of string. Let $S = s_1 s_2 \cdots s_n$ and the i-th substring of S be $\hat{S}_{[i]} = s_i s_{i+1} \cdots s_n$, where s denotes a symbol from an alphabet. More precisely, the trivial construction works as follows. We split the string S into the set of strings $\{\hat{S}_{[1]}, \hat{S}_{[2]}, \cdots, \hat{S}_{[n]}\}$ and S' into the set of strings $\{\hat{S}'_{[1]}, \hat{S}'_{[2]}, \cdots, \hat{S}'_{[n]}\}$. To encrypt a message with S, we generate all ciphertexts of fuzzy-IBE encrypted with all produced strings of S. Similarly, we compute private keys of fuzzy-IBE on all produced strings of S'. If there exist a substring $\hat{S}$ of S and a substring $\hat{S}'$ of S' where their overlap distance is at least d, the recipient of ciphertexts will select the ciphertext encrypted with $\hat{S}$ and the private key of $\hat{S}'$ to extract the message.

Unfortunately, the above trivial construction is inefficient and accompanied with $O(n)$ numbers of ciphertexts and private keys of fuzzy-IBE. We note that the most efficient fuzzy-IBE scheme (or more generally threshold attribute-based encryption [16]) in the literature can achieve constant-size ciphertext, but the private key size grows linearly at least in the length of total attribute number. It means this trivial construction for GCSE gives $O(n^2)$ size of private keys. Notice that $O(n^2)$-length private key is quite impractical especially when the encryption is proposed for the aforementioned applications. In a gene sequence, the average length of a gene is 27 thousands [22]. If we want to generate a private key for such a long gene sequence, we have $n = 27,000$ and $O(n^2)$ is impractical in terms of implementation. Another drawback of this trivial construction is in the encryption phase. When each ciphertext computation costs $O(n)$ time (e.g. [30, 16]), it requires $O(n^2)$ time in the encryption phase of GCSE. Hence, this trivial construction, although straightforward, is undesirable.

Our Contributions We formalize the notion of *Generalized Closest Substring Encryption* (GCSE in short). Briefly, a ciphertext encrypted with a string S can be decrypted with a private key of another string S', if and only if the overlap distance between one of the substrings of S and one of the substrings of S' is at least d. The number d is determined by either the key authority during key generation or an encryptor in ciphertext generation, but in this work, when referring to the definition and construction of GCSE, we focus on the first type, unless it is specified otherwise. We note that this encryption system *does not* restrict the identical length of S and S'. It is also applicable to those scenarios where the string S' of the private key is much shorter than the input string S used in the encryption phase.

We invent an entirely new approach in the construction of GCSE, which is much more efficient compared to the trivial construction. We map strings S, S' into two specific integer strings E, K respectively, with the same length. If there exist a substring of S and a substring of S' such that their overlap distance is at least d, we can find d pairs (For each pair, one is from E and the other one is from K) of integers such that their sums are all equal to some specific integer. There is no restriction (e.g. identical positions) in choosing integers, and hence we do not need to split strings S and S' into many strings. In our construction, a ciphertext encrypted with an n_1-length string S consists of $n_1 + 2$ group elements. A private key for an n_2-length string S' only consists of n_2 group elements. Both ciphertext and private key therefore grow only linearly in the length of string. Our encryption and decryption also cost linear time only in the length of input string. We prove the security of our system under the selective model. The security is based on a generalization of decision q-Bilinear Diffie-Hellman Exponent (BDHE) assumption. Additionally, we also construct a GCSE system of the second type, which allows flexibility for the encryptor to determine the overlap distance. The trade-off is a bit longer ciphertext and private key in comparison to the first scheme.

1.1 Overview of Our Construction

In this section, we give a high level overview of the main ideas of our construction. For strings with alphabet size $|\Sigma| = c$, we use an integer from the integer range $[1, c]$ to represent each symbol. In this work, we assume that each symbol is from the space $[1, c]$.

Let $S = s_1\ s_2\ \cdots\ s_{n_1} \in [1, c]^{n_1}$ and $S' = s'_1\ s'_2\ \cdots\ s'_{n_2} \in [1, c]^{n_2}$, where s_i denotes the i-th symbol of string S. If there exist a substring of S and a substring of S' such that their overlap distance is at least d, denoted by $CS(S, S') \geq d$, then there exist integers a_1, a_2, L such that the overlap distance between the following two L-length substrings is at least d.

$$d_O\Big(s_{a_1+1}\ s_{a_1+2}\ \cdots\ s_{a_1+L},\ \ s'_{a_2+1}\ s'_{a_2+2}\ \cdots\ s'_{a_2+L}\Big) \geq d.$$

In other words, we can find at least d number of identical locations (integers) $\{b_1, b_2, \cdots, b_d\}$ from $\{1, 2, \cdots, L\}$ such that $s_{a_1+b_i} = s'_{a_2+b_i}$ for all $b_i \in \{b_1, b_2, \cdots, b_d\}$.

The main ideas of our construction are inspired from the following observation. Let E and K be two particular strings defined as

$$\begin{aligned} E = E[1]E[2]\ E[3]\cdots E[n_1]: &\quad E[i] = c \cdot (n_1 - i) \\ K = K[1]K[2]K[3]\cdots K[n_2]: &\quad K[i] = c \cdot i, \end{aligned}$$

where $E[i], K[i]$ denote the i-th symbol of E and K, respectively. We observe that the sum of $E[a_1 + b_i]$ and $K[a_2 + b_i]$ is equal to $c \cdot (n_1 - a_1 + a_2)$, which is independent of b_i. Next, we consider E and K embedded with S and S' respectively, which are defined as

$$\begin{aligned} E = E[1]E[2]\ E[3]\cdots E[n_1]: &\quad E[i] = c \cdot (n_1 - i) + s_i \\ K = K[1]K[2]K[3]\cdots K[n_2]: &\quad K[i] = c \cdot i - s'_i. \end{aligned}$$

We have $E[i] + K[i'] = c \cdot (n_1 - i + i') + s_i - s'_{i'}$. The sum of $E[i]$ and $K[i']$ falls into the following two cases associated with $CS(S, S')$.

– Case 1: If $CS(S, S') \geq d$. Then, there exist at least d pairs $(E[i], K[i'])$ with distinct i' satisfying the equation $E[i] + K[i'] = c \cdot j$ for some integer j. From the definition of $CS(S, S')$, they are

$$\Big(E[a_1 + b_i], K[a_2 + b_i]\Big) : \ \ i = 1, 2 \cdots, d, \ \ j = (n_1 - a_1 + a_2).$$

– Case 2: $CS(S, S') < d$ otherwise. In this case, for any integer j, we found that the number of pairs $(E[i], K[i'])$ satisfying $E[i] + K[i'] = c \cdot j$ is less than d. In particular, when $s_i \neq s'_{i'}$ for any i and i', we have $E[i] + K[i'] = c \cdot (n_1 - i + i') + s_i - s'_{i'} \neq 0 \ (mod \ c)$. It means that the number of pairs satisfying $E[i] + K[i'] = c \cdot j$ for any j is zero.

We note that the above two strings E and K have a similar classification under the overlap distance between S and S', where $a_1 = a_2 = 0$. If $d_O(S, S') \geq d$, there exist at least d pairs $(E[i], K[i'])$ with distinct i' satisfying $E[i] + K[i'] = c \cdot j$ for some j. They are $(E[b_1], K[b_1]), (E[b_2], K[b_2]), \cdots, (E[b_d], K[b_d])$. Otherwise, the number of pairs $(E[i], K[i'])$ satisfying $E[i] + K[i'] = c \cdot j$ for $j = n_1$ is less than d. The difference is that j must be equal to n_1 instead of any integer in Case 2 . Therefore, a GCSE system based on these embedded integer strings could imply a fuzzy-IBE system if the changed definition on j in Case 2 will not affect the security.

Our GCSE construction is based on the above observation. In particular, we use embedded E in encryption for string S and embedded K in private key generation for string S'. To understand our idea clearly, we will first introduce the generation of private key as follows.

The private key $sk_{S'}$ of $S' = s'_1 \ s'_2 \ \cdots \ s'_{n_2}$ is defined as $g^{f(i)\alpha^{c \cdot i - s'_i}}$ for $i = 1, 2, \cdots, n_2$. Here, g is a group element of pairing group with p prime order, $f(x) \in \mathbb{Z}_p[x]$ is a random $(d-1)$-degree polynomial function and $f(0) = \tau$ is constant. The master secret key is (α, τ).

The ciphertext encrypted with $S = s_1 \ s_2 \ \cdots \ s_{n_1}$ is generated as follows. The encryption algorithm first selects a random number $r \in \mathbb{Z}_p$ and computes $e(g, g)^r$ to encrypt message. Next, it generates two other parts of ciphetext for decryptors to compute $e(g, g)^r$. The first part of ciphertext is defined as $g^{r\alpha^{c(n_1 - i) + s_i}}$ for all $i = 1, 2, \cdots, n_1$, where g^{α^l} for all potential integers l are the master public key. The second part of ciphertext is $g^{\beta r}$, where g^{β} and $g^{\beta^{-1}(1 - \tau\alpha^{c \cdot j})}$ for all $j = 1, 2, \cdots, 2n$ are the master public key.

The decryption on the ciphertext for S by using the private key $sk_{S'}$ is available *iff* $CS(S, S') \geq d$. Let $a_1, a_2, b_1, b_2, \cdots, b_d$ be integers satisfying $s_{a_1 + b_i} = s'_{a_2 + b_i}$. Using the result in Case 1, the decryption algorithm first picks $g^{r\alpha^{c(n_1 - a_1 - b_i) + s_{a_1 + b_i}}}$ from the ciphertext and $g^{f(a_2 + b_i)\alpha^{c(a_2 + b_i) - s'_{a_2 + b_i}}}$ from the private key for all b_i. Then, it computes $e(g, g)^{rf(a_2 + b_i)\alpha^{c(n_1 - a_1 + a_2)}}$ for all $\{b_1, b_2, \cdots, b_d\}$. Next, it computes $e(g, g)^{rf(0)\alpha^{c \cdot (n_1 - a_1 + a_2)}}$ by running the Lagrange polynomial interpolation. Finally, the decryption algorithm picks $j = n_1 - a_1 + a_2$ and computes $e(g^{\beta r}, g^{\beta^{-1}(1 - \tau\alpha^{c \cdot (n_1 - a_1 + a_2)})})$, which is equal to $e(g, g)^{r - r\tau\alpha^{c(n_1 - a_1 + a_2)}}$. The multiplication $e(g, g)^{rf(0)\alpha^{c \cdot (n_1 - a_1 + a_2)}} \cdot e(g, g)^{r - r \cdot \tau\alpha^{c(n_1 - a_1 + a_2)}}$ is the final encryption key $e(g, g)^r$, which is used to extract the message.

The security requires that an adversary cannot decrypt a ciphertext encrypted with S using a private key $sk_{S'}$ for (S', d) if $CS(S, S') < d$. In the following, we provide an intuitive analysis to show why this security holds. Since the adversary can easily compute $e(g, g)^{r - r \cdot \tau\alpha^{c \cdot j}}$ from the master public key and ciphertext for any $j \in \{1, 2, \cdots, 2n\}$, it must be hard to compute $e(g, g)^{r \cdot \tau\alpha^{c \cdot j}}$ without a valid private key.

Given $g^{r\alpha^{c(n_1 - i) + s_i}}$ for any i from the ciphertext and $g^{f(i')\alpha^{c \cdot i' - s'_{i'}}}$ for any i' from the private key, the adversary can compute $e(g, g)^{rf(i')\alpha^{c(n_1 - i + i') + s_i - s'_{i'}}}$. The complete computation results are provided in the

following array

$$\begin{array}{llll} e(g,g)^{rf(x_{1,1})\alpha^1} & e(g,g)^{rf(x_{1,2})\alpha^1} & \cdots & e(g,g)^{rf(x_{1,l_1})\alpha^1} \\ e(g,g)^{rf(x_{2,1})\alpha^2} & e(g,g)^{rf(x_{2,2})\alpha^2} & \cdots & e(g,g)^{rf(x_{2,l_2})\alpha^2} \\ \cdots & & & \\ e(g,g)^{rf(x_{2nc,1})\alpha^{2nc}} & e(g,g)^{rf(x_{2nc,2})\alpha^{2nc}} & \cdots & e(g,g)^{rf(x_{2nc,l_{2nc}})\alpha^{2nc}} \end{array}, \quad (1)$$

where $1 \leq x_{i,k} \leq n_1$ are distinct in each row and the maximum exponent of α is not more than $2nc$. In this array, we use l_i to denote the number of group elements in the i-th row. If $CS(S, S') < d$, we deduce $l_{c\cdot j} < d$ for any positive integer $j \in \{1, 2, \cdots, 2n\}$ according to Case 2. The computation task of the adversary is to compute $e(g,g)^{rf(0)\alpha^{c\cdot j}}$ for any j from computation results in the array (1) where $l_{c\cdot j} < d$.

We found the above computation task is not easier than computing $e(g,g)^r$ from given group elements $e(g,g)^{r\alpha^i}$ for all $i = \pm 1, \pm 2, \cdots, \pm q$ and $q = 2n + 1$, which is a variant of q-BDHE problem defined in [5]. To prove this transformation, we program the function $f(x)$ as

$$f(x) = w\alpha + \sum_{j=1}^{2n} \Big(\prod_{i=1}^{l_{c\cdot j}} (x - x_{c\cdot j,i}) \frac{w_j}{\alpha^{c\cdot j}} \Big),$$

where w is a random number and $w_j^{-1} = \prod_{i=1}^{l_{c\cdot j}} (0 - x_{c\cdot j,i})$. We have $f(0) = w\alpha + \sum_{j=1}^{2n} \frac{1}{\alpha^{c\cdot j}}$. It is not hard to verify that each group element in the array (1) can be seen as a multi-exponentiation from $e(g,g)^{r\alpha^i}$ for $i = \pm 1, \pm 2, \cdots, \pm q$ without the need of $e(g,g)^{r\alpha^0}$. Meanwhile, $e(g,g)^r$ can be extracted from $e(g,g)^{rf(0)\alpha^{c\cdot j}}$ for any $j \in \{1, 2, \cdots, 2n\}$. That is, if $e(g,g)^{rf(0)\alpha^{c\cdot j}}$ for any j is computable from the array (1), we are able to program the reduction to solve the q-BDHE problem.

The security assumption we finally adopt is the generalization of decision q-BDHE problem. Let $h = g^r$. The assumption says that it is hard to distinguish $e(g,h)^{\alpha^0}$ from a random element, when given

$$g^{\alpha^x}, x \in X, \quad h^{\alpha^y}, y \in Y$$

where X, Y are two integer sets chosen by the adversary satisfying $|x| \leq q, |y| \leq q$ and $x + y \neq 0$ for $\forall x \in X, \forall y \in Y$. This assumption is one type of decision (P, Q, f)-general Diffie-Hellman assumptions [3]. We give a proof that the adopted assumption holds in the generic group model. The formal security proof of our system is based on the above intuitive analysis and programmed function $f(x)$ in each private key simulation.

1.2 Other Related Work

Identity-based encryption is the basic form of functional encryption which is proposed from a designated receiver. The first fully secure construction was proposed by Boneh and Franklin [4] using bilinear pairing.

Functional encryption is an emerging paradigm for public-key encryption. In a functional encryption system a user will learn $F(k, M)$ from decryption which is determined by the functionality $F(\cdot, \cdot)$ of the encrypted message M and the input private key of value k issued by some authority. The beginning of functional encryption can be traced back to the fuzzy-IBE [30] and attribute-based encryption (ABE) [15]. The forms of ABE fall into Key-Policy ABE and Ciphertext-Policy ABE. In KP-ABE, a ciphertext is associated with a set of attributes and a private key is associated with a function. The decryption will be successful if the function accepts the set of attributes. On the contrary, in CP-ABE, a ciphertext is associated with a function while a private key is associated with a set of attributes.

Over the past several years, there have been significant works on functional encryption including definition [26, 6], adaptive security proof [19, 24], revocation of private keys [29], multi authorities [7, 8] and

the expression of function. The core of functionality has also been developed to support various operations, such as threshold [30, 16], AND gates [9, 32], monotone span programs [15, 2], inner product [17, 23, 19, 25], circuit [13, 14] and DFA (Deterministic Finite Automata) [31]. The proposed constructions [30, 1] of fuzzy-IBE in the literature can be seen as one type of ABE with threshold functionality. Among the existing functional encryption schemes, fuzzy-IBE is the only notion introduced to measure a certain distance of two strings that supports error-tolerance. There is no other encryption systems which provide a more advanced distance measurement.

2 Definitions and Assumption

2.1 Definition of Overlap Distance Between Substrings

Let strings be over alphabet Σ with size c. We use an integer from $[1, c]$ to denote each symbol in Σ. Based on this knowledge, let $S = s_1\ s_2\ \cdots\ s_{n_1} \in [1, c]^{n_1}$ and $S' = s'_1\ s'_2\ \cdots\ s'_{n_2} \in [1, c]^{n_2}$, where s_i denotes the i-th symbol of string S. We say S and S' have a substring with overlap distance d, denoted by $CS(S, S') \geq d$, if there exist integers a_1, a_2, L such that

$$d_O\Big(s_{a_1+1}\ s_{a_1+2}\ \cdots\ s_{a_1+L},\ \ s'_{a_2+1}\ s'_{a_2+2}\ \cdots\ s'_{a_2+L}\Big) \geq d.$$

Here, $d_O(\cdot, \cdot)$ denotes the overlap distance of two given L-length strings and is used to count the number of identical positions at which the corresponding symbols are the same. We note that the overlap distance is equivalent to the length of string subtracting their Hamming distance.

When $CS(S, S') \geq d$, there exist identical locations (integers) $\{b_1, b_2, \cdots, b_d\} \subseteq \{1, 2, \cdots, L\}$ such that $s_{a_1+b_i}$ is equal to $s'_{a_2+b_i}$ for $i = 1, 2, \cdots, d$. Let $k_i = a_1 + b_i$ and $k'_i = a_2 + b_i$ be the positions of symbols $s_{a_1+b_i}$ and $s'_{a_2+b_i}$ in the complete string S and S', respectively. We have

$$k'_1 - k_1 = k'_2 - k_2 = k'_3 - k_3 = \cdots = k'_d - k_d = a_2 - a_1.$$

2.2 Definition of Generalized Closest Substring Encryption

A generalized closest substring encryption comprises the following four algorithms.

Setup The setup algorithm takes as input the security parameter and integers (n, c), where n denotes the maximum length of string and c is the size of alphabet of strings. It returns a master public key MPK and a master secret key MSK.

Encryption The encryption algorithm takes as input the master public key MPK, a string S with $|S| \leq n$ and a message M. It outputs a ciphertext CT.

Key Generation The key generation algorithm takes as input the master secret key MSK, a string S' with $|S'| \leq n$ and an overlap distance d. It returns a private key $sk_{S'}$ for (S', d).

Decryption The decryption takes as input a ciphertext CT encrypted with S, the master public key MPK and the private key $sk_{S'}$ for (S', d). The decryption algorithm attempts to decrypt the ciphertext and outputs the ciphertext if $CS(S, S') \geq d$. Otherwise, it simply returns the symbol $\perp$.

Correctness for Generalized Closest Substring Encryption Consider all (MPK, MSK), (n, c), (S, M) and $(S', d, sk_{S'})$. Suppose CT is generated from (MPK, S, M) and $sk_{S'}$ is computed from (MSK, S', d). If $CS(S, S') \geq d$, we have the decryption on CT using the private key $sk_{S'}$ which will output message M.

Security for Generalized Closest Substring Encryption Without a valid private key for (S', d) satisfying $CS(S, S') \geq d$, it requires that an adversary knows nothing about the message in CT encrypted with S.

2.3 Definition of Security Model

We now describe the security model for the GCSE. The security model is similar to an identity-based encryption. An adversary can only query private keys for (S_i, d_i) satisfying $CS(S^*, S_i) < d_i$ when S^* is the challenge string. Otherwise, the adversary can trivially win the game by decrypting the challenge ciphertext by itself. The game between an adversary and a challenger is defined as follows.

Setup. The challenger first runs the setup algorithm, gives MPK to the adversary and keeps MSK by itself.

Phase 1. The adversary makes private keys queries for any string and overlap distance (S_i, d_i) adaptively chosen by itself. The challenger runs the key generation algorithm and returns the private key sk_{S_i} to the adversary.

Challenge. The adversary submits a string S^* and two messages M_0, M_1 for challenge. It requires that $CS(S^*, S_i) < d_i$ holds for all (S_i, d_i) queried in the phase 1. Then, the challenger flips a random coin $coin \in \{0, 1\}$, and computes $CT^* = Encrypt[S^*, M_{coin}]$. The challenge ciphertext CT^* is given to the adversary.

Phase 2. Phase 1 is repeated with the restriction on any (S_i, d_i) satisfying $CS(S^*, S_i) \geq d_i$.

Guess. The adversary outputs a guess $coin'$ of $coin$.

The advantage of the adversary in this game is defined as $\epsilon = \Pr[coin' = coin] - \frac{1}{2}$.

Definition 1. *A generalized closest substring encryption system is (t, q_k, ϵ) semantically secure if for all t polynomial time adversaries who make q_k private key queries have a negligible advantage ϵ in the above game. We say that the system is selectively secure if we need to add a selective-S phase before the setup phase, where the adversary commits to the challenge string S^*.*

2.4 Complexity Assumption

Let $\mathbb{BG} = (\mathbb{G}, \mathbb{G}_T, e, p, g)$ be a bilinear group and g be a group element from $\mathbb{G}$. Here, $\mathbb{G}, \mathbb{G}_T$ are groups with prime order p and e is the bilinear map. The security proof of our system is based on the generalization of q-Bilinear Diffie-Hellman Exponent problem. The q-Bilinear Diffie-Hellman Exponent problem, which was first introduced in [5], is to compute $e(h, g)^{\alpha^q}$, when given $h, g, g^{\alpha}, \cdots, g^{\alpha^{q-1}}, g^{\alpha^{q+1}}, \cdots, g^{\alpha^{2q}}$. This problem is equivalent to computing $e(h, g)$ when given h, g^{α^i} for all $i = \pm 1, \pm 2, \cdots, \pm q$. We note that all exponents of α (i.e. 0 in the h base and $\pm 1, \pm 2, \cdots, \pm q$ in the g base) in the given instance are not determined by an adversary.

In the generalization of q-Bilinear Diffie-Hellman Exponent problem (q-GBDHE in short), all exponents of α in the given instance are conditionally determined by the adversary. In this problem, the given instance is generated by a challenge oracle. An adversary is allowed to query two integer sets X, Y to this challenge oracle that returns g^{α^x} for any $x \in X$ and returns h^{α^y} for any $y \in Y$, where the two integer sets X, Y satisfy the following restrictions.

- All absolute values of integers in X and Y are not larger than q.
- The sum of x and y is nonzero for $\forall x \in X$ and $\forall y \in Y$.

Then, it must remain hard to compute $e(g, h)^{\alpha^0}$.

We say that an adversary who outputs the guess on Z has advantage ϵ in the solving the decision q-GBDHE problem if

$$\left| \Pr\left[\mathcal{A}^{O_c(\cdot), X, Y}(Z = e(g, h)) = 0\right] - \Pr\left[\mathcal{A}^{O_c(\cdot), X, Y}(Z = R) = 0\right] \right| = \epsilon.$$

The integer sets in the variant of q-BDHE problem are $X = \{\pm 1, \pm 2, \cdots, \pm q\}$ and $Y = \{0\}$. Boneh, Boyen and Goh [3] introduced a general Diffie-Hellman exponent assumption (P, Q, f)-GDHE, which has included many assumptions including the q-BDHE assumption. Actually, our assumption is also one type of the decision (P, Q, f)-GDHE assumptions with a specific P definition. In the Appendix, we review the general GDHE problem and give a proof that the decision q-GBDHE assumption holds in the generic group model.

Definition 2 (Decision q-GBDHE). *We say that the decision generalization of q-Bilinear Diffie-Hellman Exponent assumption holds with (t, ϵ) if no t-polynomial time adversary has a non-negligible advantage ϵ in solving the problem.*

3 Construction

In this construction, the authority determines the overlap distance d for input string during the key generation. A random $(d-1)$-degree polynomial function $f(x) \in \mathbb{Z}_p[x]$ is chosen in the key generation where $f(0)$ is constant for all strings. The interpolation polynomial in the Lagrange form is used for decryption. Let $\Delta_{b_i,d}$ be the Lagrange coefficient for $b_i \in \{b_1, b_2, \cdots, b_d\} \in \mathbb{Z}_p$ to compute $f(0)$. We define $f(0)$ is computed as $f(0) = \prod_{i=1}^{d} f(b_i)\Delta_{b_i,d}$.

3.1 Algorithms

Setup The setup algorithm takes as input a security parameter 1^λ and integers (n, c), where n denotes the maximum length of string and c denotes the size of alphabet Σ. It first chooses a bilinear group $\mathbb{BG} = (\mathbb{G}, \mathbb{G}_T, e, p)$ and a random group element $g \in \mathbb{G}$. The algorithm then chooses random α, β, τ from $\mathbb{Z}_p$. Finally, for all $i = 1, 2, \cdots, cn$ and $j = 1, 2, \cdots, 2n$, group elements g_i, u_0, u_j, e_0 are computed as follows

$$
\begin{aligned}
&g_i = g^{\alpha^i} : && i = 1, 2, \cdots, cn \\
&u_0 = g^{\beta} \\
&u_j = g^{\frac{1}{\beta}(1-\tau\alpha^{cj})} : && j = 1, 2, \cdots, 2n \\
&e_0 = e(g, g).
\end{aligned}
$$

The master secret key includes (g, α, τ), and the master public key are the description of bilinear group along with $(n, c, g_i, u_0, u_j, e_0)$ for all $i = 1, 2, \cdots, cn$ and $j = 1, 2, \cdots, 2n$.

Encryption The encryption algorithm takes as input the master public key, an n_1-length string $S = s_1\ s_2\ \cdots\ s_{n_1} \in [1, c]^{n_1}$ for any $n_1 \leq n$, and a message $M \in \mathbb{G}_T$. The encryption algorithm chooses a random r from $\mathbb{Z}_p$ and creates the ciphertext as follows.

$$
\begin{aligned}
C_m &= e(g, g)^{-r} \cdot M \\
C_0 &= u_0^r \\
C_i &= g^r_{c(n_1-i)+s_i} : i = 1, 2, \cdots, n_1
\end{aligned}
$$

The output ciphertext is $CT = (C_m, C_0, C_1, \cdots, C_{n_1})$.

Key Generation The key generation algorithm takes as input the master secret key, an n_2-length string $S' = s'_1\ s'_2\ \cdots\ s'_{n_2} \in [1, c]^{n_2}$ for any $n_2 \leq n$ and an overlap distance d. The algorithm begins by randomly choosing a $(d-1)$-degree polynomial function $f(x) \in \mathbb{Z}_p[x]$ such that $f(0) = \tau$. Then, the algorithm

computes $g_{c\cdot i-s'_i} = g^{\alpha^{c\cdot i-s'_i}}$ and $f_i = f(i)$ for all $i = 1, 2, \cdots, n_2$. Finally, it computes the private key of S' as follows.

$$sk_{S'} = (sk_1, sk_2, \cdots, sk_{n_2}) = \Big(g^{f_1}_{c\cdot 1-s'_1},\ g^{f_2}_{c\cdot 2-s'_2},\ \cdots,\ g^{f_{n_2}}_{c\cdot n_2-s'_{n_2}}\Big).$$

Decryption Suppose that a ciphertext CT is encrypted with S and we have a private key $sk_{S'}$ for (S', d) where $CS(S, S') \geq d$. Then, there exist integers a_1, a_2, L such that

$$d_O\Big(s_{a_1+1}\ s_{a_1+2}\ \cdots\ s_{a_1+L},\ \ s'_{a_2+1}\ s'_{a_2+2}\ \cdots\ s'_{a_2+L}\Big) \geq d.$$

and $s_{a_1+b_i} = s'_{a_2+b_i}$ for at least d number of locations b_i in $\{b_1, b_2, \cdots, b_d\} \subseteq \{1, 2, \cdots, L\}$.

The decryption algorithm begins by computing:

$$e(C_{a_1+b_i}, sk_{a_2+b_i}) = e\Big(g^r_{c(n_1-a_1-b_i)+s_{a_1+b_i}}, g^{f_{a_2+b_i}}_{c(a_2+b_i)-s'_{a_2+b_i}}\Big) = e(g, g)^{\alpha^{c(n_1-a_1+a_2)}rf(a_2+b_i)}.$$

Then, the ciphertext can be decrypted as

$$\begin{aligned} & C_m \cdot e\Big(C_0, u_{n_1-a_1+a_2}\Big) \cdot \prod_{i=1}^{d} \Big(e(g, g)^{\alpha^{c(n_1-a_1+a_2)}rf(a_2+b_i)}\Big)^{\Delta_{a_2+b_i,d}} \\ &= e(g, g)^{-r} \cdot M \cdot e(g, g)^{r-r\cdot\tau\alpha^{c(n_1-a_1+a_2)}} \cdot e(g, g)^{rf(0)\alpha^{c(n_1-a_1+a_2)}} \\ &= M. \end{aligned}$$

3.2 CCA Security and Flexible Overlap Distance

The construction is proposed for chosen-plaintext security. We can extend it to chosen-ciphertext security by applying the techniques of using simulation-sound NIZK proofs [28], or the Fujisaki-Okamoto transformation [12] based on random oracles. The corresponding security model is similar to identity-based encryption, where it allows the adversary to make any decryption query except the challenge ciphertext.

In this construction, the overlap distance is determined by the private key generator. It will be more flexible if an encryptor can decide the overlap distance in practice. That is, the encryption takes as input S and an integer d_e. A private key of any string S' can decrypt the ciphertext if $CS(S, S') \geq d_e$. Fortunately, our construction can be extended to encryption with a flexible overlap distance d_e decided by the encryptor. We present the construction with flexible overlap distance in the Appendix. In our construction, the encryption takes as input (S, d_e) and the private key generation takes as (S', d), where d is a fixed value for all strings and $d_e \leq d$. The corresponding decryption will be successful if $CS(S, S') \geq d_e$. The extension additionally generates $d - d_e$ group elements in ciphertext and $n + n_2 + d$ group elements in private key.

4 Security Proof

In this section, we prove the security of our construction. To simplify the proof, we first give three lemmas to explain why the simulation will be successful without abortion in the selective model.

Definition 1 *Let n, n^*, n_i, c be positive integers satisfying $n_i, n^* \leq n$ and $2 \leq c$. Let $S^* = s_1 s_2 \cdots s_{n^*} \in [1, c]^{n^*}$ and $S_i = s^i_1 s^i_2 \cdots s^i_{n_i} \in [1, c]^{n_i}$. Define $\mathbb{S}^i_j$ for all $j = 1, 2, \cdots, 2n$ to be subsets of $\{1, 2, \cdots, n_i\}$, where $k \in \mathbb{S}^i_j$ if there exists $k^* \in \{1, 2, \cdots, n^*\}$ satisfying*

$$c(n^* - k^*) + s_{k^*} + c \cdot k - s^i_k = c \cdot j.$$

Define $\mathbb{T}^i_k$ for all $k = 1, 2, \cdots, n_i$ to be subsets of $\{1, 2, \cdots, 2n\}$, where $j \in \mathbb{T}^i_k$ if $k \notin \mathbb{S}^i_j$.

In the above definition, $\mathbb{S}_j^i$ is defined to capture all integers $\{x_{cj,1}, x_{cj,2}, \cdots, x_{cj,l_{cj}}\}$ defined in the array in Section 1.1 for string S_i, and $\mathbb{T}_k^i$ is defined to capture remained exponents of α in $g_{c\cdot k - s_k^i}^{f(k)}$. We have

$$c(n^* - k^*) + s_{k^*} + c \cdot k - s_k^i \leq c(n-1) + c + cn - 1 = 2nc - 1 < c(2n)$$

holds for all k^* and k, such that $\mathbb{S}_{2n}^i = \emptyset$. Therefore, we have $2n \in \mathbb{T}_k^i$ because $k \notin \mathbb{S}_{2n}^i$.

Lemma 1 *If $CS(S^*, S_i) < d_i$, we have $|\mathbb{S}_j^i| < d_i$ for all $j = 1, 2, \cdots, 2n$.*

Proof. Otherwise, without loss of generality, suppose $\mathbb{S}_j^i = \{k_1, k_2, \cdots, k_{d_i}\}$ that has d_i integers. According to the definition of $\mathbb{S}_j^i$, there exist $k_1^*, k_2^*, \cdots, k_{d_i}^*$ from $\{1, 2, \cdots, n^*\}$ such that

$$\begin{aligned}
c(n^* - k_1^*) + s_{k_1^*} + c \cdot k_1 - s_{k_1}^i &= c(n^* - k_1^* + k_1) + s_{k_1^*} - s_{k_1}^i = c \cdot j \\
c(n^* - k_2^*) + s_{k_2^*} + c \cdot k_2 - s_{k_2}^i &= c(n^* - k_2^* + k_2) + s_{k_2^*} - s_{k_2}^i = c \cdot j \\
&\cdots \\
c(n^* - k_{d_i}^*) + s_{k_{d_i}^*} + c \cdot k_{d_i} - s_{k_{d_i}}^i &= c(n^* - k_{d_i}^* + k_{d_i}) + s_{k_{d_i}^*} - s_{k_{d_i}}^i = c \cdot j
\end{aligned}$$

From the above equations, we have $s_{k_j^*} \equiv s_{k_j}^i \pmod{c}$ and then deduce $s_{k_j^*} = s_{k_j}^i$ for all $j = 1, 2, \cdots, d_i$ from the definition of $s \in [1, c]$. Then, we also have

$$k_1^* - k_1 = k_2^* - k_2 = \cdots = k_{d_i}^* - k_{d_i} = n^* - j.$$

$$d_O\Big(s_{k_1^*}\ s_{k_1^*+1}\ s_{k_1^*+2}\ \cdots\ s_{k_{d_i}^*},\ \ s_{k_1}^i\ s_{k_1+1}^i\ s_{k_1+2}^i\ \cdots s_{k_{d_i}}^i\Big) \geq d_i,$$

and then $CS(S^*, S_i) \geq d_i$, which is contrary to $CS(S^*, S_i) < d_i$. This completes the proof of Lemma 1. □

Lemma 1 is used in programming a $(d_i - 1)$-degree polynomial function $f(x)$ for the key generation on S_i. The following two lemmas are used to indicate that the two integer sets X, Y chosen by the simulator will be accepted by the challenge oracle of the decision q-GBDHE problem.

Lemma 2 *$\forall k \in \{1, 2, \cdots, n_i\}$, $\forall j \in \mathbb{T}_k^i$ and $\forall k^* \in \{1, 2, \cdots, n^*\}$, we have*

$$\Big(c \cdot k - s_k^i - c \cdot j\Big) + \Big(c(n^* - k^*) + s_{k^*}\Big) \neq 0.$$

Proof. Otherwise, we have $\exists k \in \{1, 2, \cdots, n_i\}$, $\exists j \in \mathbb{T}_k^i$ and $\exists k^* \in \{1, 2, \cdots, n^*\}$ satisfying

$$\Big(c \cdot k - s_k^i - c \cdot j\Big) + \Big(c(n^* - k^*) + s_{k^*}\Big) = 0.$$

According to the definition of $\mathbb{S}_j^i$ and $\mathbb{T}_k^i$, we have $j \in \mathbb{T}_k^i$ and then $k \notin \mathbb{S}_j^i$, which is contrary to $k \in \mathbb{S}_j^i$ from the above equation $c(n^* - k^*) + s_{k^*} + c \cdot k - s_k^i = c \cdot j$. This completes the proof of Lemma 2. □

Lemma 3 *Given any strings $S^*, S_1, S_2, \cdots, S_{q_k}$, let X and Y be two integer sets defined as*

$$\begin{aligned}
X &= \Big\{ -(4nc+1), -2nc \Big\} \cup [1, 2nc-1] \cup [2nc+1, 4nc+1] \\
&\quad \cup \Big\{ ck - s_k^i - cj : k = 1, 2, \cdots, n_i, j \in \mathbb{T}_k^i, i = 1, 2, \cdots, q_k \Big\} \\
Y &= \Big\{ -2nc \Big\} \cup \Big\{ c(n^* - k^*) + s_{k^*} : k^* = 1, 2, \cdots, n^* \Big\}.
\end{aligned}$$

We have $x + y \neq 0$ holds for $\forall x \in X, \forall y \in Y$.

Proof. Otherwise, we have $\exists x^* \in X$ and $\exists y^* \in Y$ satisfying $x^* + y^* = 0$. Let X_1, X_2, X_3, Y_1, Y_2 be subsets of X and Y defined as

$$
\begin{aligned}
X_1 &= \Big\{ -(4nc+1), -2nc \Big\} \\
X_2 &= [1, 2nc-1] \cup [2nc+1, 4nc+1] \\
X_3 &= \Big\{ ck - s_k^i - cj : k = 1, 2, \cdots, n_i, j \in \mathbb{T}_k^i, i = 1, 2, \cdots, q_k \Big\} \\
Y_1 &= \Big\{ -2nc \Big\} \\
Y_2 &= \Big\{ c(n^* - k^*) + s_{k^*} : k^* = 1, 2, \cdots, n^* \Big\}.
\end{aligned}
$$

We have

$$
\begin{aligned}
-2nc \le ck - s_k^i - cj \le cn - 1 - c \le 2nc - 1 \\
1 \le c(n^* - k^*) + s_{k^*} \le c(n^* - 1) + c \le cn.
\end{aligned}
$$

Therefore, the only potential $x^* \in X, y^* \in Y$ satisfying $x^* + y^* = 0$ is choosing x^* from X_3 and y^* from Y_2. However, according to Lemma 2, we have $x + y \neq 0$ for $\forall x \in X_3, \forall y \in Y_2$. Therefore, x^*, y^* satisfying $x^* + y^* = 0$ do not exist. This completes the proof of Lemma 3. □

In our proof, the simulation of private key and challenge ciphertext are closely related to integers defined in Lemma 3. With this property, the simulator is able to get all group elements from the challenge oracle of decision q-GBDHE problem to complete simulation. Our proof is completed in the following theorem.

Theorem 1 *Suppose the Decision q-GBDHE assumption holds with (ϵ, t). Then, no adversary can break our construction with (ϵ, q_k, t') under the selective model, where $q = 4nc + 1$ and $t' = t - O(q_k n^2)$.*

Proof. Suppose we have an adversary $\mathcal{A}$ with advantage ϵ in the selective game against our construction. Then, we construct a reduction algorithm $\mathcal{B}$ that solves the decision q-GBDHE problem with advantage ϵ. $\mathcal{B}$ runs $\mathcal{A}$ and simulates the security game as follows.

Selective-S The adversary declares a string S^* for challenge where $n^* \le n$. Let $S^* = s_1\ s_2\ s_3\ \cdots\ s_{n^*}$. $\mathcal{B}$ first defines two integer sets X^*, Y^* as follows

$$
\begin{aligned}
X^* &= \Big[-(4nc+1),\ 4nc+1 \Big] \setminus \{-Y^*\} \\
Y^* &= \Big\{ -2nc,\ \ c(n^*-1) + s_1,\ \ c(n^*-2) + s_2,\ \ \cdots, c(n^* - n^*) + s_{n^*} \Big\},
\end{aligned}
$$

where $\{-Y^*\}$ denotes all opposite numbers from $\{Y^*\}$, and $A \setminus B$ denotes the complement set of B in A. Then, it queries X^*, Y^* to the challenge oracle to get group elements g^{α^x} and h^{α^y} for all $x \in X^*$ and $y \in Y^*$.

It is easy to verify that $x + y \neq 0$ for $\forall x \in X^*$ and $\forall y \in Y^*$. Let X be the integer set defined in Lemma 3, we have $X \subseteq X^*$. In the following simulation, all required x in g^{α^x} for setup and key generation are from the set X.

Setup Let the master secret key α of GCSE scheme be the same as the secret in the decision q-GBDHE problem. Taking as input integers (n, c), the reduction algorithm first chooses random w_β, w_τ from $\mathbb{Z}_p$ and sets β, τ to be

$$
\beta = w_\beta \alpha^{-2nc},\ \ \tau = w_\tau \alpha + \sum_{l=1}^{2n} \frac{1}{\alpha^{c \cdot l}}.
$$

Then, the master public key can be re-written into

$$
\begin{aligned}
g_i &= g^{\alpha^i} \ : \ \ i = 1, 2, \cdots, cn \\
u_0 &= g^{\beta} = (g^{\alpha^{-2nc}})^{w_\beta} \\
u_j &= g^{\frac{1}{\beta}(1-\tau\alpha^{cj})} = g^{-\frac{1}{\beta}\left(w_\tau \alpha^{cj+1} + \sum_{l=1, l\neq j}^{2n} \frac{1}{\alpha^{cl-cj}}\right)} \\
&= \left(g^{\alpha^{2nc+cj+1}}\right)^{-\frac{w_\tau}{w_\beta}} \cdot \prod_{l=1, l\neq j}^{2n} \left(g^{\alpha^{2nc+cj-cl}}\right)^{-\frac{1}{w_\beta}} \ : \ \ j = 1, 2, \cdots, 2n \\
e_0 &= e(g, g) = e\left(g^{\alpha^{-(4nc+1)}}, g^{\alpha^{4nc+1}}\right),
\end{aligned}
$$

where the exponents of α in g_i and u_j satisfy

$$
\begin{aligned}
1 \leq 1, 2, \cdots, cn \leq 2nc - 1 \\
2nc + 1 \leq 2nc + cj + 1 \leq 4nc + 1 \\
1 \leq 2nc + cj - cl \leq 4nc + 1 \\
2nc + cj - cl \neq 2nc.
\end{aligned}
$$

The integers x in all above related to g^{α^x} are from $\{-(4nc+1), -2nc\}$, $[1, 2nc-1]$ and $[2nc+1, 4nc+1]$. Finally, the simulator computes the master public key using g^{α^x} and the above re-written structures.

The reduction algorithm embeds the unknown secret α into the master public key and randomize other secret keys β, τ using w_β, w_τ. They are therefore random and independent from the view of adversary.

Key Generation We now describe how the reduction algorithm can simulate a private key for (S_i, d_i) where $CS(S^*, S_i) < d_i$. The most complicated part of key simulation is the choice of $(d_i - 1)$-degree polynomial function $f(x) \in \mathbb{Z}_p[x]$.

Let $S_i = s_1^i \ s_2^i \ \cdots s_{n_i}^i$. The private key of S_i is

$$
(sk_1, sk_2, \cdots, sk_{n_i}) = \left(g_{c\cdot 1 - s_1^i}^{f_1}, \ g_{c\cdot 2 - s_2^i}^{f_2}, \ \cdots, \ g_{c\cdot n_i - s_{n_i}^i}^{f_{n_i}}\right),
$$

where

$$
sk_k = g_{c\cdot k - s_k^i}^{f_k} = g^{f_k \cdot \alpha^{c\cdot k - s_k^i}} = g^{\alpha^{c\cdot k - s_k^i} \cdot f(k)}, \ \ f(0) = w_\tau \alpha + \sum_{j=1}^{2n} \frac{1}{\alpha^{c\cdot j}}.
$$

That is, the private key of each sk_k is computed from a mutli-exponentiation of $g^{\alpha^{x_i}}$. To respond the private key query, we must make sure that all x_i are in X^* in order to simulate the private key.

The exponent of sk_k might contain $\alpha^{c\cdot k - s_k^i - c\cdot j}$ for some $j = 1, 2, \cdots, 2n$. According to Lemma 3, all queries on integers $c \cdot k - s_k^i - c \cdot j$ for those $j \in \mathbb{T}_k^i$ are within X^*. Therefore, we must eliminate all queries to get $g^{\alpha^{c\cdot k - s_k^i - c\cdot j}}$ for other $j \notin \mathbb{T}_k^i$ (i.e. $j \in \{1, 2, \cdots, 2n\} \setminus \mathbb{T}_k^i$). Suppose $f_k \cdot \alpha^{c\cdot k - s_k^i}$ in the simulation can be re-written into

$$
f_k \cdot \alpha^{c\cdot k - s_k^i} = \prod_{j=1}^{2n} A_j \alpha^{c\cdot k - s_k^i - cj} = \prod_{j \in \mathbb{T}_k^i} A_j \alpha^{c\cdot k - s_k^i - cj} + \prod_{j \notin \mathbb{T}_k^i} A_j \alpha^{c\cdot k - s_k^i - cj},
$$

where A_j are coefficients. We don't need $g^{\alpha^{c\cdot k - s_k^i - c\cdot j}}$ for those $j \notin \mathbb{T}_k^i$ if A_j is equal to zero.

We achieve this property by programming the polynomial function $\prod_{x_i \in \mathbb{S}_j^i} (x - x_i)$ with regard to $\frac{1}{\alpha^{c\cdot j}}$. Let $F_j(x)$ be

$$
F_j(x) = \alpha^{c\cdot k - s_k^i} \prod_{x_i \in \mathbb{S}_j^i} (x - x_i) \frac{1}{\alpha^{c\cdot j}} = \alpha^{c\cdot k - s_k^i - c\cdot j} \prod_{x_i \in \mathbb{S}_j^i} (x - x_i).
$$

When $j \notin \mathbb{T}_k^i$, we have $k \in \mathbb{S}_j^i$ (Definition 1) such that

$$F_j(k) = \alpha^{c\cdot k - s_k^i - c\cdot j} \prod_{x_i \in \mathbb{S}_j^i} (k - x_i) = 0 \cdot \alpha^{c\cdot k - s_k^i - c\cdot j} = 0.$$

When $CS(S^*, S_i) < d_i$, we have $|\mathbb{S}_j^i| < d_i$ according to Lemma 1 such that $F_j(x)$ is a polynomial function with degree $d_i - 1$ at most. This means we can embed such a polynomial function in the $(d_i - 1)$-degree polynomial function $f(x)$.

$\mathcal{B}$ starts by randomly choosing integer sets $\mathbb{U}_j^i$ for all $j = 1, 2, \cdots, 2n$ with $d_i - 1 - |\mathbb{S}_j^i|$ elements from $\mathbb{Z}_p^* \setminus \{1, 2, \cdots, n\}$, and computing w_j as

$$w_j = \Big(\prod_{x_i \in \mathbb{S}_j^i \cup \mathbb{U}_j^i} (0 - x_i) \Big)^{-1} \neq 0.$$

Next, $\mathcal{B}$ sets the polynomial function $f(x)$ for S_i as

$$f(x) = w_\tau \alpha + \sum_{j=1}^{2n} \Big(\prod_{x_i \in \mathbb{S}_j^i \cup \mathbb{U}_j^i} (x - x_i) \frac{w_j}{\alpha^{c\cdot j}} \Big).$$

If $\sum_{j=1}^{2n} \frac{w_j}{\alpha^{c\cdot j}} = 0$, we deduce $w_{2n} = -\sum_{j=1}^{2n-1} w_j \alpha^{c(2n-j)}$. Then, we can immediately solve the hard problem by computing $e(g, h)$ as

$$e(g, h) = e\Big(\prod_{j=1}^{2n-1} g^{-w_j \alpha^{4nc - cj}}, h^{\alpha^{-2nc}} \Big)^{\frac{1}{w_{2n}}},$$

from given instance where $2nc + 1 \leq 4nc - cj \leq 4nc + 1$. Otherwise, $f(x)$ is a $(d_i - 1)$-degree polynomial function and $f(0) = \tau$. The exponent $f_k \cdot \alpha^{c\cdot k - s_k^i}$ of sk_k satisfies

$$\begin{aligned}
\alpha^{c\cdot k - s_k^i} f(k) &= w_\tau \alpha^{c\cdot k - s_k^i + 1} + \sum_{j=1}^{2n} \Big(\prod_{x_i \in \mathbb{S}_j^i \cup \mathbb{U}_j^i} w_j (k - x_i) \alpha^{c\cdot k - s_k^i - c\cdot j} \Big) \\
&= w_\tau \alpha^{c\cdot k - s_k^i + 1} + \sum_{j=1}^{2n} \Big(A_j \alpha^{c\cdot k - s_k^i - c\cdot j} \Big) \\
&= w_\tau \alpha^{c\cdot k - s_k^i + 1} + \sum_{j \in \mathbb{T}_k^i} A_j \alpha^{c\cdot k - s_k^i - c\cdot j}.
\end{aligned}$$

We have $1 \leq ck - s_k^i + 1 \leq 2nc - 1$ and then

$$\Big\{ ck - s_k^i + 1,\ c\cdot k - s_k^i - c\cdot j :\ \ k = 1, 2, \cdots, n_i, \quad j \in \mathbb{T}_k^i \Big\} \subseteq X^*.$$

Finally, $\mathcal{B}$ computes the private key sk_k of s_k^i for S_i as

$$sk_k = \Big(g^{\alpha^{c\cdot k - s_k^i + 1}} \Big)^{w_\tau} \cdot \prod_{j \in \mathbb{T}_k^i} \Big(g^{\alpha^{ck - s_k^i - c\cdot j}} \Big)^{A_j}.$$

Challenge The adversary returns $M_0, M_1 \in \mathbb{G}_T$ for challenge on the challenge string $S^* = s_1 s_2 \cdots s_{n^*}$. The reduction algorithm randomly chooses $coin \in \{0, 1\}$ and creates the ciphertext CT^* as

$$\begin{aligned} CT^* &= (C_m,\ C_0,\ C_1,\ C_2,\ \cdots,\ C_{n^*}) \\ &= \Big(Z^{-1} \cdot M_{coin},\ \ (h^{\alpha^{-2nc}})^{w_\beta},\ \ h^{\alpha^{c(n^*-1)+s_1}},\ \ h^{\alpha^{c(n^*-2)+s_2}},\ \ \cdots,\ \ h^{\alpha^{c(n^*-n^*)+s_{n^*}}}\Big). \end{aligned}$$

Let $r = \log_g^h$, if $Z = e(g, h)$, we have

$$\begin{aligned} Z^{-1} \cdot M_{coin} &= e(g, g)^{-r} \cdot M_{coin} \\ (h^{\alpha^{-2nc}})^{w_\beta} &= g^{\beta r} = u_0^r \\ h^{\alpha^{c(n^*-i)+s_i}} &= g^r_{c(n^*-i)+s_i} \end{aligned}$$

such that CT^* is an encryption of GCSE for M_{coin}.

Guess The adversary $\mathcal{A}$ eventually returns a guess $coin'$ on $coin$. If $coin' = coin$, the reduction algorithm $\mathcal{B}$ outputs 1 to guess that $Z = e(g, h)$; Otherwise, it outputs 0 to indicate that Z is a random element in $\mathbb{G}_T$.

This completes the reduction proof of our construction. According to Lemma 3 and the simulation, $\mathcal{B}$ perfectly simulates private keys and the challenge ciphertext. If $Z = e(g, h)$, CT^* is a valid GCSE encryption on M_{coin} and $\mathcal{A}$ will guess $coin$ correctly with probability $\frac{1}{2} + \epsilon$, where ϵ is the advantage of adversary. When Z is a random element in $\mathbb{G}_T$, CT^* can be seen as a one-time encryption on M_{coin}. The ciphertext reveals no information about $coin$ to the adversary, and the adversary will guess $coin$ with probability $1/2$ only. Therefore, the reduction algorithm can solve the decision q-GBDHE problem with advantage ϵ. The simulation time is mainly dominated by the key simulation, where each sk_k requires $O(n)$ point multiplications in $\mathbb{G}$. We therefore prove the Theorem 1. □

5 Conclusion

The closest substring problem is a useful pattern matching problem, which measures the similarity of two given strings more precisely via their substrings compared to others like measuring the overlap distance between two complete strings. We proposed a new type of encryption system motivated by this measurement. In this system, a ciphertext encrypted with a string S cannot be decrypted with a private key of string S', unless the overlap distance between one of the substrings of S and one of the substrings of S' is at least d, which is either determined by the key authority or the encryptor. We proposed concrete schemes of generalized closest substring encryption systems, which are efficient and practical. Both ciphertext and private key grow linearly only in the length of string. We proved the security of our systems in the selective model under the generalization of q-Bilinear Diffie-Hellman problem.

References

1. Baek, J., Susilo, W., Zhou, J.: New constructions of fuzzy identity-based encryption. In: Bao, F., Miller, S. (eds.) ASIACCS. pp. 368–370. ACM (2007)
2. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy 2007. pp. 321–334. IEEE Computer Society (2007)
3. Boneh, D., Boyen, X., Goh, E.J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)
4. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005)

6. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011)
7. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
8. Chase, M., Chow, S.S.M.: Improving privacy and security in multi-authority attribute-based encryption. In: Al-Shaer, E., Jha, S., Keromytis, A.D. (eds.) ACM Conference on Computer and Communications Security 2009. pp. 121–130. ACM (2009)
9. Cheung, L., Newport, C.C.: Provably secure ciphertext policy ABE. In: ACM Conference on Computer and Communications Security 2007. pp. 456–465. ACM (2007)
10. Davila, J., Balla, S., Rajasekaran, S.: Fast and practical algorithms for planted (l, d) motif search. IEEE/ACM Trans. Comput. Biology Bioinform. 4(4), 544–552 (2007)
11. Deng, X., Li, G., Li, Z., Ma, B., Wang, L.: Genetic design of drugs without side-effects. SIAM J. Comput. 32(4), 1073–1090 (2003)
12. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. J. Cryptology 26(1), 80–101 (2013)
13. Garg, S., Gentry, C., Halevi, S., Sahai, A., Waters, B.: Attribute-based encryption for circuits from multilinear maps. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 479–499. Springer, Heidelberg (2013)
14. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) STOC 2013. pp. 545–554. ACM (2013)
15. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., di Vimercati, S.D.C. (eds.) ACM Conference on Computer and Communications Security 2006. pp. 89–98. ACM (2006)
16. Herranz, J., Laguillaumie, F., Ràfols, C.: Constant size ciphertexts in threshold attribute-based encryption. In: Nguyen, P.Q., Pointcheval, D. (eds.) Public Key Cryptography 2010. LNCS, vol. 6056, pp. 19–34. Springer, Heidelberg (2010)
17. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
18. Lanctôt, J.K., Li, M., Ma, B., Wang, S., Zhang, L.: Distinguishing string selection problems. Inf. Comput. 185(1), 41–55 (2003)
19. Lewko, A.B., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
20. Lucas, K., Busch, M., Mossinger, S., Thompson, J.A.: An improved microcomputer program for finding gene- or gene family-specific oligonucleotides suitable as primers for polymerase chain reactions or as probes. Computer Applications in the Biosciences 7(4), 525–529 (1991)
21. Marx, D.: The closest substring problem with small distances. In: FOCS 2005. pp. 63–72. IEEE Computer Society (2005)
22. Morse, S.P.: Selected Lectures on Genealogy: An Introduction to Scientific Tools, chap. DNA to genetic genealogy, pp. 57–82. Weizmann Institute of Science, Rehovot, Israel (2013) (`http://www.stevemorse.org/genetealogy/dna.htm`)
23. Okamoto, T., Takashima, K.: Hierarchical predicate encryption for inner-products. In: Matsui, M. (ed.) ASIACRYPT. LNCS, vol. 5912, pp. 214–231. Springer, Heidelberg (2009)
24. Okamoto, T., Takashima, K.: Fully secure functional encryption with general relations from the decisional linear assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
25. Okamoto, T., Takashima, K.: Adaptively attribute-hiding (hierarchical) inner product encryption. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT. LNCS, vol. 7237, pp. 591–608. Springer, Heidelberg (2012)
26. O'Neill, A.: Definitional issues in functional encryption. IACR Cryptology ePrint Archive 2010, 556 (2010)
27. Proutski, V., Holmes, E.C.: Primer master: a new program for the design and analysis of pcr primers. Computer Applications in the Biosciences 12, 253–255 (1996)
28. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: FOCS 1999. pp. 543–553. IEEE Computer Society (1999)
29. Sahai, A., Seyalioglu, H., Waters, B.: Dynamic credentials and ciphertext delegation for attribute-based encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 199–217. Springer, Heidelberg (2012)
30. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
31. Waters, B.: Functional encryption for regular languages. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 218–235. Springer, Heidelberg (2012)
32. Zhou, Z., Huang, D.: On efficient ciphertext-policy attribute based encryption and broadcast encryption: extended abstract. In: ACM Conference on Computer and Communications Security 2010. pp. 753–755. ACM (2010)

Appendix A: Generic Security of Decision q-GBDHE Assumption

In this section, we prove the intractability of the decision q-GBDHE problem involved in the decision (P, Q, f)-GDHE problem [3]. We first review the definition of this problem.

Definition 3 (Decision (P, Q, f)-GDHE Problem). *Given the tuple*

$$\Big(g^{P(x_1,x_2,\cdots,x_m)},\ e(g,g)^{Q(x_1,x_2,\cdots,x_m)},\ Z\Big),$$

decide whether $Z = e(g,g)^{f(x_1,x_2,\cdots,x_m)}$. Here, $P, Q \in \mathbb{F}_p[X_1, X_2, \cdots, X_m]^s$ are two s-tuples of m-variate polynomials over $\mathbb{F}_p$, and $f \in \mathbb{F}_p[X_1, X_2, \cdots, X_m]$.

Since P, Q are just two lists containing s multivariate polynomials, we can write them into $P = (p_1, p_2, \cdots, p_s)$ and $Q = (q_1, q_2, \cdots, q_s)$, where $p_1 = q_1 = 1$. We say f depends on (P, Q), denoted by $f \in \langle P, Q\rangle$, if there exist $\{a_{i,j}\}_{1\le i,j\le s}$ and $\{b_i\}_{1\le i\le s}$ such that

$$f = \sum_{1\le i,j\le s} a_{i,j} \cdot p_i \cdot p_j + \sum_{1\le i\le s} b_i q_i.$$

The decision (P, Q, f)-GDHE problem is proved to be hard in [3] when $f \notin \langle P, Q\rangle$.

The adopted q-GBDHE problem is to decide whether $Z = e(g, h)$ when given

$$\{g^{\alpha^x} : x \in X\},\quad \{h^{\alpha^y} : y \in Y\},$$

where $|x|, |y| \le q$ and $x + y \ne 0$ for any $x \in X$ and $y \in Y$. Let $h = g^\eta$, $X = \{x'_1, x'_2, \cdots, x'_{q_x}\}$ and $Y = \{y'_1, y'_2, \cdots, y'_{q_y}\}$. Our problem can be reformulated as the (P, Q, f)-GDHE problem where

$$\begin{aligned} P &= \Big(\alpha^{x'_1},\ \alpha^{x'_2},\ \cdots,\ \alpha^{x'_{q_x}},\ \eta\alpha^{y'_1},\ \eta\alpha^{y'_2},\ \cdots,\ \eta\alpha^{y'_{q_y}}\Big)\\ Q &= 1\\ f &= \eta, \end{aligned}$$

and thus $m = 2$ and $s = q_x + q_y$. To claim the hardness of the q-GBDHE problem, we must prove $f \notin \langle P, Q\rangle$. By making all possible products of two polynomials from P that are multiples of η, we want to prove no linear combination among the polynomials from the following list R below leads to f.

$$R = \Big(\eta\alpha^{x'_i+y'_j} : x'_i \in X, y'_j \in Y\Big).$$

Note that all exponents of α in the R list are non-zero according to the definition of the q-GBDHE problem, and f is equivalent to $\eta\alpha^0$. We therefore have f, which is independent of any linear combination from R and $f \notin \langle P, Q\rangle$.

Appendix B: Construction with Flexible Overlap Distance

Algorithms

Setup As in the basic scheme, the setup algorithm generates the basic master secret key and master pubic key. In addition, it randomly chooses $\gamma \in \mathbb{Z}_p$ and computes $v_k = g^{\gamma\alpha^{ck}}$ for all $k = 1, 2, \cdots, n$. The master secret key includes $(g, \alpha, \tau, \gamma)$, and the master public key are the basic master public key $(n, c, g_i, u_0, u_j, e_0)$ along with v_k for all $k = 1, 2, \cdots, n$.

Encryption The encryption algorithm takes as input the master public key, a string $S = s_1 s_2 \cdots s_{n_1} \in [1, c]^{n_1}$ for any $n_1 \le n$, an integer $d_e (\le d)$ and a message $M \in \mathbb{G}_T$. As in the basic scheme, the encryption algorithm generates the basic ciphertext CT. Additionally, using the chosen random number r in creating CT, the algorithm computes

$$\Big(C^F_{d_e+1}, C^F_{d_e+2}, \cdots, C^F_d\Big) = \Big(v^r_{d_e+1}, v^r_{d_e+2}, \cdots, v^r_d\Big).$$

The output ciphertext is $CT_F = (CT, C^F_{d_e+1}, C^F_{d_e+2}, \cdots, C^F_d)$.

Key Generation As in the basic scheme, the key generation algorithm generates the basic private key $sk_{S'}$ for string S'. Let $f(x)$ be the $(d-1)$-polynomial function chosen in the basic private key generation. In addition, the algorithm computes

$$\{sk^F_l\} = \Big\{ g^{\gamma^{-1}\alpha^{c\cdot l} f(2n+l)} : \ \ l = 1-d, -d, \cdots, n+n_2-1 \Big\}.$$

The output private key $sk^F_{S'}$ of S' is composed of $sk_{S'}$ and $\{sk^F_l\}$.

Decryption Suppose that a ciphertext CT_F is encrypted with (S, d_e) and we have a private key $sk_{S'}$ of S' satisfying $CS(S, S') \geq d_e$. As in the basic scheme, the decryption algorithm computes

$$e(C_{a_1+b_i}, sk_{a_2+b_i}) = e\Big(g^r_{c(n_1-a_1-b_i)+s_{a_1+b_i}}, g^{f_{a_2+b_i}}_{c(a_2+b_i)-s'_{a_2+b_i}}\Big) = e(g,g)^{\alpha^{c(n_1-a_1+a_2)} r f(a_2+b_i)}$$

for all $b_i \in \{b_1, b_2, \cdots, b_{d_e}\} \subseteq \{1, 2, \cdots, L\}$. Then, for all $i = d_e+1, d_e+2, \cdots, d$, the algorithm computes

$$\begin{aligned} e(C^F_i, sk^F_{n_1-a_1+a_2-i}) &= e\Big(g^{\gamma\alpha^{c\cdot i} r}, g^{\gamma^{-1}\alpha^{c(n_1-a_1+a_2-i)} f(2n+n_1-a_1+a_2-i)}\Big) \\ &= e(g,g)^{\alpha^{c(n_1-a_1+a_2)} r f(2n+n_1-a_1+a_2-i)}. \end{aligned}$$

Let $a_2 + b_i = 2n + n_1 - a_1 + a_2 - i$ for all $i = d_e+1, d_e+2, \cdots, d$. Finally, the algorithm decrypts the ciphertext the same as the basic scheme by

$$\begin{aligned} & C_m \cdot e\Big(C_0, u_{n_1-a_1+a_2}\Big) \cdot \prod_{i=1}^{d} \Big(e(g,g)^{\alpha^{c(n_1-a_1+a_2)} r f(a_2+b_i)}\Big)^{\Delta_{a_2+b_i,d}} \\ &= e(g,g)^{-r} \cdot M \cdot e(g,g)^{r - r\cdot\tau\alpha^{c(n_1-a_1+a_2)}} \cdot e(g,g)^{r\cdot f(0)\alpha^{c(n_1-a_1+a_2)}} \\ &= M. \end{aligned}$$

Correctness

For all $0 \leq n_1, a_1, a_2 \leq n$, $0 \leq d \leq n$, $1 \leq d_e \leq d$ and $d_e + 1 \leq i \leq d$, we have

$$1 - d \leq d_e - i \leq (n_1 - a_1 + a_2) - i \leq n_1 + n_2 - d_e - i \leq n + n_2 - 1$$

so that the encryptor can always find the key $sk^F_{n_1-a_1+a_2-i}$ from $sk^F_{S'}$. Also, we have

$$a_2 + b_i \leq n < 2n + n_1 - a_1 + a_2 - i.$$

Therefore, $a_2 + b_i$ for all $\{b_1, b_2, \cdots, b_d\}$ are distinct and $f(0)$ can be computed using interpolation polynomial $f(a_2 + b_i)$ in the Lagrange form.

Security Proof

Definition 2 *Define $\mathbb{S}^F_j$ for all $j = 1, 2, \cdots, 2n$ to be subsets of $\{-n+1, -n+2, \cdots, 2n-1\}$, where $l \in \mathbb{S}^F_j$ if there exists $l^* \in \{d^*_e + 1,\ d^*_e + 2,\ \cdots,\ d\}$ satisfying $l + l^* = j$. Define $\mathbb{T}^F_l$ for all $l = -n+1, -n+2, \cdots, 2n-1$ to be subsets of $\{1, 2, \cdots, 2n\}$, where $j \in \mathbb{T}^i_l$ if $l \notin \mathbb{S}^i_j$.*

Lemma 4 *$|\mathbb{S}^F_j| = d - d^*_e$ holds for all $j = 1, 2, \cdots, 2n$.*

Proof. Given any $j \in \{1, 2, \cdots, 2n\}$ and $l^* \in \{d_e^* + 1, d_e^* + 2, \cdots, d\}$, we have

$$1 - n \leq j - d \leq j - l^* \leq j - d_e^* - 1 \leq 2n - 1$$

such that

$$\mathbb{S}_j^F = \Big\{ j - (d_e^* + 1), j - (d_e^* + 2), \cdots, j - d \Big\}$$

and then $|\mathbb{S}_j^F| = d - d_e^*$. This completes the proof of Lemma 4. □

Lemma 5 *$\forall l \in \{-n+1, -n+2, \cdots, 2n-1\}$, $\forall j \in \mathbb{T}_l^F$ and $\forall l^* \in \{d_e^* + 1, d_e^* + 2, \cdots, d\}$, we have*

$$(l - j) + l^* \neq 0.$$

Proof. Otherwise, we have $\exists l \in \{-n+1, -n+2, \cdots, 2n-1\}$, $\exists j \in \mathbb{T}_l^F$ and $\exists l^* \in \{d_e^* + 1, d_e^* + 2, \cdots, d\}$ satisfying

$$(l - j) + l^* = 0.$$

According to the definition of $\mathbb{S}_j^F$ and $\mathbb{T}_l^F$, we have $j \in \mathbb{T}_l^F$ and then $l \notin \mathbb{S}_j^F$, which is contrary to $l \in \mathbb{S}_j^F$ from $l + l^* = j$. This completes the proof of Lemma 5. □

Lemma 6 *Let X and Y be the two integer sets defined in Lemma 3. Let X^F, Y^F be two integer sets defined as*

$$\begin{aligned}
X^F &= X \cup \{6nc + c \cdot k : k = 1, 2, \cdots, n\} \\
&\quad \cup \Big\{ -6nc + c(l - j) : l = -n + 1, -n + 2, \cdots, 2n - 1, j \in \mathbb{T}_l^F \Big\} \\
&\quad \cup \Big\{ -6nc + c \cdot l + 1 : l = -n + 1, -n + 2, \cdots, 2n - 1 \Big\} \\
Y^F &= Y \cup \Big\{ 6nc + c \cdot l^* : l = d_e^* + 1, d_e^* + 2, \cdots, d \Big\}.
\end{aligned}$$

We have $x + y \neq 0$ holds for $\forall x \in X^F, \forall y \in Y^F$.

Proof. Otherwise, we have $\exists x^* \in X$ and $\exists y^* \in Y$ satisfying $x^* + y^* = 0$. Let X_4, X_5, X_6, Y_3 be subsets of X^F, Y^F defined as

$$\begin{aligned}
X_4 &= \Big\{ 6nc + c \cdot k : k = 1, 2, \cdots, n \Big\} \\
X_5 &= \Big\{ -6nc + c(l - j) : l = -n + 1, -n + 2, \cdots, 2n - 1, j \in \mathbb{T}_l^F \Big\} \\
X_6 &= \Big\{ -6nc + c \cdot l + 1 : l = -n + 1, -n + 2, \cdots, 2n - 1 \Big\} \\
Y_3 &= \Big\{ 6nc + c \cdot l^* : l^* = d_e^* + 1, d_e^* + 2, \cdots, d \Big\}.
\end{aligned}$$

Since $n \geq 1$ and $c \geq 2$, we have

$$\begin{aligned}
&2nc + 1 \leq 6nc + c \cdot k \leq 7nc \\
&\quad -9nc \leq -6nc + c(l - j) \leq -6nc + c(2n - 1 - 1) \leq -2nc - 1 \\
&\quad -9nc \leq -6nc + c \cdot l + 1 \leq -6nc + c(2n - 1) + 1 \leq -2nc - 1 \\
&4nc + 2 \leq 6nc + c \cdot l^* \leq 7nc.
\end{aligned}$$

Therefore, $\forall x \in X, \forall y \in Y$, we have

$$\begin{aligned}
&|x| \leq 4nc + 1 < y' : \quad \forall y' \in Y_3 \\
&|y| \leq 2nc < |x'| : \qquad \forall x' \in X_4 \cup X_5 \cup X_6.
\end{aligned}$$

Lemma 3 says $x + y \neq 0$ for $\forall x \in X$ and $\forall y \in Y$. Thus, the only potential $x^* \in X, y^* \in Y$ satisfying $x^* + y^* = 0$ is choosing x^* from $X_5 \cup X_6$ and y^* from Y_3. If $x^* \in X_5$ and $y^* \in Y_3$, we have $x^* + y^* \neq 0$ according to the Lemma 5. Otherwise, $x^* \in X_6$ and $y^* \in Y_3$, we have $x^* + y^* = 1 (\mathsf{mod}\ c)$ and then $x^* + y^* \neq 0$. This completes the proof of Lemma 6. □

Theorem 2 *Suppose the Decision q-GBDHE assumption holds with (ϵ, t). Then, no adversary can break our construction with (ϵ, q_k, t') under the selective model, where $q = 9nc$ and $t' = t - O(q_k n^2)$.*

Proof. Suppose we have an adversary $\mathcal{A}$ with advantage ϵ in the selective game against our flexible construction. Then, we construct a reduction algorithm $\mathcal{B}$ that solves the decision q-GBDHE problem with advantage ϵ. $\mathcal{B}$ runs $\mathcal{A}$ and simulates the security game as follows.

Selective-S The adversary declares (S^*, d_e^*) for challenge where $n^* \leq n$ and $d_e^* \leq d$. Let $S^* = s_1 s_2 \cdots s_{n^*}$. Then, the reduction algorithm defines two integer sets X^*, Y^* as follows

$$\begin{aligned} X^* &= \Big[-9nc,\ 9nc \Big] \setminus \{-Y^*\} \\ Y^* &= \Big\{ -2nc,\ \ c(n^*-1)+s_1,\ \ c(n^*-2)+s_2,\ \ \cdots, c(n^*-n^*)+s_{n^*} \Big\} \\ &\quad \cup \Big\{ 6nc + c(d_e^*+1),\ \ 6nc + c(d_e^*+2),\ \ \cdots,\ \ 6nc + c\cdot d \Big\}. \end{aligned}$$

Then, $\mathcal{B}$ queries X^*, Y^* to the challenge oracle to get group elements g^{α^x} and h^{α^y}.

We also have $x + y \neq 0$ for $\forall x \in X^*$ and $\forall y \in Y^*$ and $X \subseteq X^*$, where X is defined in Lemma 6. In the following simulation, all required x in g^{α^x} for key setup and generation are from the set X.

Setup As in the proof for the basic scheme, $\mathcal{B}$ simulates the basic master public key. To simulate $v_k = g^{\gamma\alpha^{ck}}$ for all $k = 1, 2, \cdots, n$, $\mathcal{B}$ begins by randomly choosing $w_\gamma \in \mathbb{Z}_p$ and then setting $\gamma = w_\gamma \alpha^{6nc}$. We have $\Big\{ 6nc + c\cdot k : k = 1, 2, \cdots, n \Big\} \subseteq X^*$. Finally, it computes the additional master public keys v_k as

$$v_k = g^{\gamma\alpha^{ck}} = (g^{\alpha^{6nc+ck}})^{w_\gamma} : k = 1, 2, \cdots, n.$$

Key Generation We now describe how to simulate a private key $sk_{S_i}^F$ for $S_i = s_1^i\ s_2^i\ \cdots s_{n_i}^i$ where $CS(S^*, S_i) < d_e^*$. Let $\mathbb{S}_j^i$ be sets defined in Definition 1 and let $\mathbb{S}_j^F$ be sets defined in Definition 2 for all $j = 1, 2, \cdots, 2n$.

$\mathcal{B}$ starts by randomly choosing integer sets $\mathbb{U}_j^i$ for all $j = 1, 2, \cdots, 2n$ with $d - 1 - |\mathbb{S}_j^i| - |\mathbb{S}_j^F|$ elements from $\mathbb{Z}_p^* \setminus \{1, 2, \cdots, 4n\}$, and computing w_j as

$$w_j = \Big(\prod_{x_i \in \mathbb{S}_j^F} (0 - x_i - 2n) \cdot \prod_{x_i \in \mathbb{S}_j^i \cup \mathbb{U}_j^i} (0 - x_i) \Big)^{-1} \neq 0.$$

Next, it sets the polynomial function $f(x)$ for S_i as

$$f(x) = w_\tau \alpha + \sum_{j=1}^{2n} \Big(\prod_{x_i \in \mathbb{S}_j^F} (x - x_i - 2n) \cdot \prod_{x_i \in \mathbb{S}_j^i \cup \mathbb{U}_j^i} (x - x_i) \frac{z_j}{\alpha^{c\cdot j}} \Big).$$

The same analysis as the proof for Theorem 1, we have $f(x)$ is a $(d-1)$-degree polynomial function. Then, $\mathcal{B}$ simulates the basic private key sk_{S_i} the same as in the Theorem 1.

Finally, to compute the extension key

$$sk_l^F = g^{\gamma^{-1}\alpha^{cl} f(2n+l)} :\ l = 1-d, -d, \cdots, n + n_2 - 1,$$

where

$$sk_l^F = g^{\frac{1}{w_\gamma}\alpha^{-6nc+cl}f(2n+l)},$$

and the exponent $\frac{1}{w_\gamma}\alpha^{-6nc+cl}f(2n+l)$ of sk_l^F satisfies

$$\begin{aligned}
&\frac{1}{w_\gamma}\alpha^{-6nc+cl}f(2n+l)\\
&= \frac{w_\tau}{w_\gamma}\alpha^{-6nc+cl+1} + \sum_{j=1}^{2n}\Big(\prod_{x_i\in\mathbb{S}_j^F}(2n+l-x_i-2n)\cdot\prod_{x_i\in\mathbb{S}_j^i\cup\mathbb{U}_j^i} w_j(2n+l-x_i)\alpha^{-6nc+c(l-j)}\Big)\\
&= \frac{w_\tau}{w_\gamma}\alpha^{-6nc+cl+1} + \sum_{j=1}^{2n}\Big(A_j\alpha^{-6nc+c(l-j)}\Big)\\
&= \frac{w_\tau}{w_\gamma}\alpha^{-6nc+cl+1} + \sum_{j\in\mathbb{T}_l^F}\Big(A_j\alpha^{-6nc+c(l-j)}\Big),
\end{aligned}$$

$\mathcal{B}$ computes sk_l^F for all l as follows.

$$sk_l^F = \Big(g^{\alpha^{-6nc+cl+1}}\Big)^{\frac{w_\tau}{w_\gamma}}\cdot\prod_{j\in\mathbb{T}_l^F}\Big(g^{\alpha^{-6nc+c(l-j)}}\Big)^{A_j}.$$

The simulation on the extension key will not abort since

$$\Big\{-6nc+cl+1, -6nc+c(l-j):\ \ l=-n+1,-n+2,\cdots,2n-1,\quad j\in\mathbb{T}_l^F\Big\}\subseteq X^*.$$

Challenge The adversary returns $M_0, M_1\in\mathbb{G}_T$ for challenge on (S^*, d_e^*). The reduction algorithm randomly chooses $coin\in\{0,1\}$ and creates the ciphertext CT_F^* as

$$CT_F^* = \Big(CT^*, C_{d_e^*+1}^{*F}, C_{d_e^*+2}^{*F}, \cdots, C_d^{*F}\Big) = \Big(CT^*, h^{w_\gamma\alpha^{6nc+c\cdot(d_e^*+1)}}, h^{w_\gamma\alpha^{6nc+c\cdot(d_e^*+2)}}, \cdots, h^{w_\gamma\alpha^{6nc+c\cdot d}}\Big),$$

where CT^* is simulated the same as the proof for basic scheme.

Let $r=\log_g^h$, we have

$$h^{w_\gamma\alpha^{6nc+c\cdot i}} = g^{w_\gamma\alpha^{6nc}\cdot\alpha^{ci}r} = g^{\gamma\alpha^{ci}r} = v_i^r:\ \ i=d_e^*+1, d_e^*+2,\cdots,d.$$

Therefore, the challenge cipheretext is valid for (S^*, d_e^*).

Guess The adversary $\mathcal{A}$ eventually returns a guess $coin'$ on $coin$. If $coin'=coin$, the reduction algorithm $\mathcal{B}$ outputs 1 to guess that $Z=e(g,h)$; Otherwise, it outputs 0 to indicate that Z is a random element in $\mathbb{G}_T$.

This completes the reduction proof of our construction. Following the similar analysis as the proof in Theorem 1, we then prove the Theorem 2. □