Anonymous Fuzzy Identity-based Encryption for Similarity Search

Ye Zhang, Nikos Mamoulis, David W. Cheung, S.M. Yiu, and W.K. Wong

Department of Computer Science, The University of Hong Kong

Abstract. In this paper, we consider the problem of predicate encryption and focus on the predicate for testing whether the hamming distance between the attribute X of a data item and a target V is equal to (or less than) a threshold t where X and V are of length m. Existing solutions either do not provide attribute protection or produce a big ciphertext of size $O(m2^m)$. For the equality version of the problem, we provide a scheme which is match-concealing (MC) secure and the sizes of the ciphertext and token are both $O(m)$. For the inequality version of the problem, we give two practical schemes. The first one, also achieving MC security, produces ciphertext with size $O(m^{t_{max}})$ if the maximum value of t, t_{max}, is known in advance and is a constant. We also show how to update the ciphertext if the user wants to increase t_{max} without constructing the ciphertext from scratch. On the other hand, in many real applications, the security requirement can be lowered from MC to MR (match-revealing). Our second scheme, which is MR secure, produces ciphertext of size $O(m)$ and token of size $O((t+1)m)$ only.

Key words: predicate encryption, anonymous fuzzy identity-based encryption, inner-product encryption

1 Introduction

It is getting more popular for a data owner to take advantage of the storage and computing resources of a data center to hold the data in encrypted form. Users will be given a token (by the owner) to access the data so that only authorized records can be retrieved and later be decrypted on the user site. Due to the privacy and security concern, it is obvious that the data will not be decrypted at the data center and checked against the criteria one by one. Thus computation is required to be carried out on encrypted data directly. Examples are retrieval of encrypted documents based on keyword matching, selection of encrypted audit logs using multi-dimensional range query on authorized IP addresses or port numbers, and hamming distance based similarity search on encrypted DNA sequence data. The problem, in fact, has received much attention from both database community [16, 3, 17, 13, 18, 27] and cryptography community [26, 4, 24, 10, 19].

In general, the problem can be stated as follows. For each data item M, there is an associated attribute value X (X may not be part of the record M)

and let $f : \{0,1\}^* \rightarrow \{0,1\}$ be a predicate which represents the computation we want to carried out on ciphertexts such that the data item M can be successfully decrypted if and only if $f(X) = 1$. Authorized users will obtain a token generated by the owner in order to perform the predicate evaluation. The predicate can take additional parameters, so a different token can be generated for a different parameter value which increases the flexibility of the data owner to provide different access power to different users. Here is an example. Each medicate record (M) is encrypted along with a selected region of the DNA sequence (X) of the person. When a research team is authorized to investigate the relationship between a certain DNA sequence V with diseases, this team would acquire a token which corresponds to the predicate f such that $f(X) = 1$ if and only if $HammingDist(X, V) \leq t$, say $t = 5$. By using the token, the research team would decrypt all medicate records for which the corresponding DNA sequence is similar to V. In the above motivating example, it is obvious that the research team should not infer any information on records for which the corresponding attribute X which is far away from V (i.e. $HammingDist(X, V) > 5$) since they are not authorized to do so. And it is desirable that the ciphertext $E(pk, I, M)$, where pk is the public key generated by the data owner, is the same for different V and t values such that the encryption of data items needs only to be done once. This emerging branch of encryption schemes are referred as *predicate encryption*.

Here we focus on the predicate f that tests whether the hamming distance between V and X is equal to (or less than) a certain threshold t, where V and X can be assumed as bit vectors of equal length m. Similarity search based on hamming distance[1] is an important searching criterion for record retrieval. This leads to many interesting applications in databases, bioinformatics, and other areas. Note that V and t can vary and will be given to the owner for the generation of a token independent of the ciphertext $E(pk, I, M)$.

The security of predicate encryption [19] can be classified into (1) protecting the data item only; and (2) protecting both the data item and attributes. Attribute protection is usually referred as *anonymous* in general and can be further classified into two levels: *match-revealing (MR)* [24] and *match-concealing*[2] *(MC)* [10, 19]. The difference between MR and MC is that attributes will remain hidden in MC level even if it satisfies the predicate. While in MR level, if attribute X satisfies the predicate f (i.e. $f(X) = 1$), some more information on X other than the information of $f(X) = 1$ can be leaked. In our "medicate record" example, we sometimes require the encryption scheme to be anonymous such that the DNA sequence is protected. In such cases, for example, DNA sequence may contain genetic disorder information which should be kept private for individuals. It depends on applications whether we require MC or MR level of security. For example, if attribute X is part of data item M, when X satisfies the predicate, data item M will be properly decrypted and therefore people

[1] It is well known that hamming distance of two bit vectors can provide a good necessary condition for the corresponding edit distance [11, 2] which would be useful in many database applications.

[2] In [19], match-concealing is called attribute-hiding.

can see the entire X (this is allowed by that application semantics). In such case, MR security seems to be a proper choice. So far, the predicate encryption scheme supporting this predicate is the one in [25], called "Fuzzy Identity-Based Encryption". However, it does not provide the property of anonymity (i.e., attribute protection). In this paper, we propose "anonymous fuzzy identity-based encryption" schemes to handle both the equality threshold and the inequality threshold (less than or equal to) versions of the predicate.

It is not trivial how to make the scheme in [25] anonymous. On the other hand, there is a generic solution [10] (see Appendix A) that can support the predicate we study with the property of anonymity and is MC secure. Their scheme provides a general construction to support any polynomial computable predicate. However, their scheme embeds (pre-computes for) every possible value of V and t in the ciphertext even for the equality threshold version of the problem (the same applies to the inequality version), thus the size of each ciphertext is $O(m2^m)$ which is impractical even for moderate m although the token size is constant.

1.1 Our contributions

For the equality threshold version, we provide an anonymous fuzzy identity-based encryption scheme achieving the MC level of security with both the sizes of ciphertext and token equal to $O(m)$. The construction is based on an inner-product encryption scheme in [19]. The core idea is to represent the hamming distance computation as an inner product such that X and V can be separated into the ciphertext and the token, respectively, so that V can be given only when the token is needed to be generated.

For the inequality threshold version, we provide two practical schemes to solve the problem. In many applications (e.g. in bioinformatics applications), $t << m$. Even assuming that we know the maximum value of t (t_{max}) in advance and is a constant, the size of the ciphertext produced by the solution based on [10] is still $O(2^m)$. In our first scheme, also achieving the MC security level, the sizes of ciphertext is only $O(m^{t_{max}})$ (precisely, $\sum_{i=0}^{t_{max}+1} \binom{m}{i}$) which is much smaller than $O(2^m)$ if $t_{max} << m$. The core of this scheme is to come up with an inner product expression with a total number of $\sum_{i=0}^{t+1} \binom{m}{i}$ terms to express whether $HammingDist(X, V) \leq t$ and modifying the scheme in [19] to a new primitive to support our encryption scheme. We also show how to update the ciphertext if the user wants to increase the value of t_{max} without recomputing the ciphertext from scratch.

On the other hand, in many applications (in particular for those where the attribute X is part of the data item M), we only require the schemes to be MR secure. By lowering the security requirement to MR, we provide another scheme in which the sizes of ciphertext and token are only $O(m)$ and $O((t + 1)m)$, respectively which is attractive for real applications.

1.2 Related Works

The predicate that was studied in the very beginning is "exact keyword matching". That is, whether the value hidden by the token is equal to the attribute value hidden in the ciphertext. Schemes that only provide data item security are basically "Identity-Based Encryption" [22, 6]. Schemes protecting both the data item and the attributes were initiated by Song *et al.* [26] in the private-key setting and by Boneh *et al.* [5] in the public-key setting. Relationship between [5] and "Anonymous Identity-Based Encryption" [9, 14] was revisited in [1].

Then, range query as the predicate was also considered. Boneh *et al.* devised an Augmented Broadcast Encryption [8] which allows checking if the attribute value falls within a range on encrypted data. Their scheme also provides attribute protection. Then, Boneh and Waters [10] extended it to multi-dimensional range query. Shi *et al.* [24] devised a more efficient scheme for multi-dimensional range query, but the scheme is MR secure.

The predicate investigated in this paper was initiated by [25] which only protects the data item. This predicate is powerful and has many applications other than those stated in [25]. However, there is no practical scheme supporting this predicate with attribute protection in a public-key setting. Park *et al.* [21] investigated this problem in the private-key setting and is IND2-CKA secure. Liesdonk [20] also investigated this problem in his master thesis. His scheme is in a public-key setting. However, the scheme requires the threshold value t to be fixed in the setup time.

Our work is using [19] as a framework. [19] provided schemes for handling predicates represented as inner products. Their formulation of using inner products with bounded disjunction is powerful. We show how to reduce inner products to hamming distance similarity comparison predicate, then derive a slightly different encryption scheme for better performance when considering the inequality case. In our work, we consider the problem of attribute protection in public-key setting. In some applications, people may also want to provide protection to predicate ("the token"), which is inherently unachievable in public-key setting. Note that a predicate encryption supporting inner product in private-key setting has been devised in [23] which can provide predicate privacy.

1.3 Paper Organization

The rest of this paper is organized as follows. Section 2 introduces the framework of the encryption scheme, the security models and the hard problem assumption. Section 3 presents the scheme for the equality threshold version (i.e., $HammingDist(V, X) = t$) of the problem and Section 4 deals with the inequality threshold version (i.e., $HammingDist(V, X) \leq t$) of the problem. We conclude the paper in Section 5.

2 Preliminaries

We assume that the attribute X is represented as a bit vector. The attribute V (referred as the *target attribute*) provided by the user to generate the token is also a bit vector of the same length as X. In the rest of the paper, for simplicity, we focus on predicate-only encryption, that is, we assume that we only have X without M. So, the scheme will output "1" to indicate the decryption is successful ($f(X) = 1$) and "0" otherwise. Note that extending solutions for predicate-only encryption to include the data item M can be done easily [19]. Also, there exist applications that we only need to encrypt the attribute X and based on the decryption result to retrieve the corresponding records separately.

2.1 Framework

An anonymous fuzzy identity-based encryption scheme Π consists of the following four probabilistic polynomial-time (PPT) algorithms.

- $\texttt{Setup}(1^n)$: On an unary string input 1^n where n is a security parameter, it produces the public-private key pair (pk, sk).
- $\texttt{Encrypt}(pk, X)$: Taking the public key pk and the attribute vector X, it outputs the ciphertext C.
- $\texttt{GenTK}(pk, sk, V, t)$: The token generation algorithm takes the public key pk, private key sk, outputs the token TK for the vector V and threshold t.
- $\texttt{Test}(pk, TK, C)$: Given the ciphertext C, the token TK, and the public key pk, it outputs "1" if the hamming distance between the vector X associated with C and the vector V associated with TK is equal to t (is less than or equal to t for the inequality version); "0" otherwise.

2.2 Security models

We define MR and MC security in the Selective-ID [12, 10, 24, 19] model as follows.

Definition 1. (Selective-ID secure in the match-concealing model) *An anonymous fuzzy identity-based encryption scheme $\Pi = (\texttt{Setup}, \texttt{Encrypt}, \texttt{GenTK}, \texttt{Test})$ is MC secure if for all probabilistic polynomial-time Turing machine (adversary) $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following game is negligible.*

Setup: Adversary $\mathcal{A}(1^n)$ outputs two possible equal-length vectors X_0 and X_1 to challenger $\mathcal{C}$. The challenger $\mathcal{C}$ takes a security parameter n and runs $\texttt{Setup}$ to generate pk and sk. $\mathcal{C}$ sends pk to $\mathcal{A}$.
Challenge: The challenger $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$, computes and returns $C^* \xleftarrow{\$} \texttt{Encrypt}(pk, X_b)$ to adversary $\mathcal{A}$.
Phase 1: Adversary $\mathcal{A}$ may adaptively request polynomially bounded number of tokens ("TK") for any (V_i, t_i), with the restriction that $t_i = HammingDist(V_i, X_j)$ for both $j = 0, 1$ or $t_i \neq HammingDist(V_i, X_j)$ for both $j = 0, 1$ (for inequality threshold, $t_i < HammingDist(V_i, X_j)$ for both $j = 0, 1$ or $t_i \geq HammingDist(V_i, X_j)$ for both $j = 0, 1$).
Guess: The adversary $\mathcal{A}$ outputs a guess bit b'. The advantage $\mathbf{Adv}^{\mathsf{MC}}_{\Pi,\mathcal{A}}(n)$ of $\mathcal{A}$ is defined as $|\Pr[b' = b] - \frac{1}{2}|$.

Recall that the only difference between MC and MR security is as follows. MC security requires that adversary $\mathcal{A}$ cannot gain more information on attribute X than the value of predicate $f(X)$ even when $f(X) = 1$. MR security requires this only when $f(X) = 0$. In a security definition, the above difference is formalized as given X_0 and X_1, MR security does not allow adversary $\mathcal{A}$ to request tokens from the challenger for predicate f such that $f(X_0) = f(X_1) = 1$. While in MC security, adversary $\mathcal{A}$ can freely request tokens no matter $f(X_0) = f(X_1) = 0$ or $f(X_0) = f(X_1) = 1$.

Definition 2. (Selective-ID secure in the match-revealing model) *An anonymous fuzzy identity-based encryption scheme* $\Pi = (\mathtt{Setup}, \mathtt{Encrypt}, \mathtt{GenTK}, \mathtt{Test})$ *is MR secure if for all probabilistic polynomial-time Turing machine (adversary)* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *in the following game is negligible.*

Setup: Adversary $\mathcal{A}(1^n)$ outputs two possible equal-length vectors X_0 and X_1. The challenger $\mathcal{C}$ takes a security parameter n and runs `Setup` to generate pk and sk.
Challenge: The challenger $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$, computes and returns $C^* \xleftarrow{\$} \mathtt{Encrypt}(pk, X_b)$ to adversary $\mathcal{A}$.
Phase 1: Adversary $\mathcal{A}$ may adaptively request polynomially bounded number of tokens ("TK") for any (V_i, t_i) with the restriction that $t_i \neq HammingDist(V_i, X_j)$ for both $j = 0, 1$ (for inequality threshold, $t_i < HammingDist(V_i, X_j)$ for both $j = 0, 1$).
Guess: The adversary $\mathcal{A}$ outputs a guess bit b'. The advantage $\mathbf{Adv}^{\mathsf{MR}}_{\Pi,\mathcal{A}}(n)$ of $\mathcal{A}$ is defined as $|\Pr[b' = b] - \frac{1}{2}|$.

2.3 The Hard Problem Assumption

The hard problem used in this paper is introduced by [19] and has been shown to "hold in generic bilinear groups of composite order $N = pqr$ as long as finding a non-trivial factor of N is hard".

Let $\mathcal{G}$ be a group generator which takes security parameter n as input and (randomly) outputs $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$, where $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map which can be computed efficiently. We call the group $\mathbb{G}$ bilinear group. $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic and share the same composite order $N = pqr$ where p, q and r are three large primes. We also denote $\mathbb{G}_p$, $\mathbb{G}_q$ and $\mathbb{G}_r$ are the subgroups of $\mathbb{G}$ with order of p, q and r separately. Since $N = pqr$ where p, q and r are primes, $\mathbb{G}_p$, $\mathbb{G}_q$ and $\mathbb{G}_r$ must exist and are cyclic. We define Assumption 1 as follows.

Definition 3. *We say that* $\mathcal{G}$ *satisfies "Assumption 1" if for any probabilistic polynomial-time Turing machine* $\mathcal{A}$*, the advantage of* $\mathcal{A}$*,* $|\Pr[\mathcal{A}(\bar{Z}, T_1 = g_p^{b^2 s} R_3) = 1] - \Pr[\mathcal{A}(\bar{Z}, T_2 = g_p^{b^2 s} Q_3 R_3) = 1]|$*, is negligible in security parameter* n*, where* $\bar{Z}$ *is defined as:*

$$(p, q, r, \mathbb{G}, \mathbb{G}_T, e) \xleftarrow{\$} \mathcal{G}(1^n), N = pqr, g_p \xleftarrow{\$} \mathbb{G}_p, g_q \xleftarrow{\$} \mathbb{G}_q, g_r \xleftarrow{\$} \mathbb{G}_r$$
$$Q_1, Q_2, Q_3 \xleftarrow{\$} \mathbb{G}_q, R_1, R_2, R_3 \xleftarrow{\$} \mathbb{G}_r, a, b, s \xleftarrow{\$} \mathbb{Z}_p \text{ and outputs}$$
$$\bar{Z} = \{g_p, g_r, g_q R_1, h_p = g_p^b, k_p = g_p^{b^2}, g_p^a g_q, g_p^{ab} Q_1, g_p^s, g_p^{bs} Q_2 R_2\}$$

$\mathcal{A}$ is also given $(N, \mathbb{G}, \mathbb{G}_T, e)$.

3 Scheme for Equality Threshold

In this section, we describe our scheme for handling the equality threshold version of the hamming distance predicate. Recall that both the target attribute V and the threshold t will only be known when the user wants to obtain a token from the owner and can vary for different users. Therefore, we need to produce ciphertext based on attribute X; a token based on V and t even after X is encrypted. The `Test`() needs to combine ciphertext and token together to compute hamming distance $HammingDist(X, V)$. To the best of our knowledge, we are aware that only bilinear map can provide such ability while not too powerful to break the security. Intuitively, given g^a and g^b, bilinear map combines a and b by computing $e(g^a, g^b) = e(g, g)^{ab}$. More specifically, if we encrypt attribute X as ciphertext $C = g^{f(X)}$ and generate token $TK = g^{y(V,t)}$ for target attribute V and threshold t, by bilinear map, we can construct `Test`(C, TK) as $e(C, TK) = e(g^{f(X)}, g^{y(V,t)}) = e(g, g)^{f(X)\cdot y(V,t)}$. If we can find $f(X)$ and $y(V, t)$ such that $f(X)y(V, t) = HammingDist(X, V)$, `Test`$(C, TK)$ will function correctly. More generally, $f(X)$ and $y(V, t)$ would output a vector rather than a single number. This is because given two vector $(g^{a_1}, ..., g^{a_i}, ..., g^{a_m})$ and $(g^{b_1}, ..., g^{b_i}, ..., g^{b_m})$, we would combine $\boldsymbol{a} = (a_1, ..., a_m)$ and $\boldsymbol{b} = (b_1, ..., b_m)$ by computing $\prod_{i=1}^{m} e(g^{a_i}, g^{b_i}) = e(g, g)^{\sum_{i=1}^{m} a_i, b_i} = e(g, g)^{\boldsymbol{a}\cdot\boldsymbol{b}}$ where $\boldsymbol{a}\cdot\boldsymbol{b}$ denotes the inner product [19, 7] of $\boldsymbol{a}$ and $\boldsymbol{b}$.

The core step is to represent the hamming distance (see the following lemma, whose proof is in Appendix D) as an inner product of $\boldsymbol{a}\cdot\boldsymbol{b} = f(X)\cdot y(V, t)$ so that the attribute of the data item X and the target attribute V can be encrypted separately into the ciphertext and token.

Lemma 1. *Given two bit vectors X and V of equal length m, $HammingDist(X, V)$ equals $\sum_{i=1}^{m} x_i(1 - 2v_i) + 1 \times \sum_{i=1}^{m} v_i$, where $X = x_1 \dots x_m$ and $V = v_1 \dots v_m$.*

Encryption scheme in [19] (see Appendix B) allows us generating a ciphertext C based on $\boldsymbol{a} = (a_1, ..., a_n)$ and a token TK based on $\boldsymbol{b} = (b_1, ..., b_n)$ such that given C and TK, we can compute $e(g, g)^{s[\sum_{i=1}^{n} a_i b_i]}$, where s is a random number, which gives $1_{\mathbb{G}_T}$ only when the inner product $\sum_{i=1}^{n} a_i b_i = 0$, or a random number otherwise. [19] is MC secure for above inner product predicate which allows us devising encryption schemes based on inner product expression which will be also MC secure. To evaluating whether $HammingDist(X, V) = t$, according to (11), we can check whether $e(g, g)^{s[\sum x_i(1-2v_i)+1\times(\sum v_i - t)]}$ equals $1_{\mathbb{G}_T}$ or not. Equivalently, we construct $f(X) = \boldsymbol{a} = (x_1, ..., x_m, 1)$ and $y(V, t) = \boldsymbol{b} = (1 - 2v_1, ..., 1 - 2v_m, \sum v_i - t)$. The details of the scheme are as follows.

- `Setup`(1^n). It first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$. Then, it randomly selects $g_p \xleftarrow{\$} \mathbb{G}_p$, $g_q \xleftarrow{\$} \mathbb{G}_q$ and $g_r \xleftarrow{\$} \mathbb{G}_r$. It also randomly selects $\{h_{1,i}, h_{2,i}\}_{i\in[1,m]}$, h_3 and h_4 from $\mathbb{G}_p$, and then randomly selects R, $\{R_{1,i}, R_{2,i}\}_{i\in[1,m]}$, R_3 and R_4 from $\mathbb{G}_r$. It outputs

$$pk = \{g_p, g_r, Q = g_q R, [H_{1,i} = h_{1,i} R_{1,i}, H_{2,i} = h_{2,i} R_{2,i}]_{i \in [1,m]}, \\ H_3 = h_3 R_3, H_4 = h_4 R_4\}$$

and

$$sk = \{p, q, r, g_q, [h_{1,i}, h_{2,i}]_{i \in [1,m]}, h_3, h_4\}$$

- `Encrypt`$(pk, X = x_1 ... x_i ... x_m)$. The encryption algorithm first randomly selects s, α, β from $\mathbb{Z}_N$ and $\{R'_{1,i}, R'_{2,i}\}_{i \in [1,m]}, R'_3, R'_4$ from $\mathbb{G}_r$. Then, it outputs the ciphertext C:

$$\{C_0 = g_p^s, [C_{1,i} = H_{1,i}^s Q^{\alpha x_i} R'_{1,i}, C_{2,i} = H_{2,i}^s Q^{\beta x_i} R'_{2,i}]_{i \in [1,m]}, \\ C_3 = H_3^s Q^{\alpha} R'_3, C_4 = H_4^s Q^{\beta} R'_4\}$$

- `GenTK`$(pk, sk, V = v_1 ... v_i ... v_m, t)$. It randomly selects $\{r_{1,i}, r_{2,i}\}_{i \in [1,m]}, r_3, r_4$ and f_1, f_2 from $\mathbb{Z}_N$. Then, it randomly selects Q'' and R'' from $\mathbb{G}_q$ and $\mathbb{G}_r$ respectively. It outputs the token TK:

$$\{K_0 = Q'' R'' h_3^{-r_3} h_4^{-r_4} \prod_{i=1}^{m} h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}}, \\ [K_{1,i} = g_p^{r_{1,i}} g_q^{f_1(1-2v_i)}, K_{2,i} = g_p^{r_{2,i}} g_q^{f_2(1-2v_i)}]_{i \in [1,m]}, \\ K_3 = g_p^{r_3} g_q^{f_1(\sum v_i - t)}, K_4 = g_p^{r_4} g_q^{f_2(\sum v_i - t)}\}$$

- `Test`(pk, TK, C). It outputs 1 if $r = 1_{\mathbb{G}_T}$ and 0 otherwise, where $r = e(C_3, K_3) e(C_4, K_4) e(C_0, K_0) \prod_{i=1}^{m} e(C_{1,i}, K_{1,i}) e(C_{2,i}, K_{2,i})$.

From above scheme, it is easily shown that the sizes of both ciphertext and token are $O(m)$.

Correctness analysis: Our construction is based on Lemma 1 to express the hamming distance as an inner product and then uses the inner-product encryption in [19], so the correctness can be guaranteed by the correctness of the inner-product encryption. The details of the correctness proof can be found in Appendix E.

Security analysis: Our encryption scheme can be proved to be MC secure. The proof is based on a reduction as follows. Assume that there exits an adversary $\mathcal{A}_1$ that can win the MC game of our scheme with non-negligible advantage, we can use $\mathcal{A}_1$ as a subroutine to construct an adversary A_2 that can win the MC game of the scheme in [19] with non-negligible advantage: When $\mathcal{A}_1$ outputs two vectors X_0 and X_1 to be challenged, $\mathcal{A}_2$ forwards $(X_0, 1)$ and $(X_1, 1)$ to challenger. When $\mathcal{A}_1$ asks for token query for (V, t) to $\mathcal{A}_2$, since $HammingDist(X, V) = t$ (or $\neq t$) is corresponding to $\sum x_i(1 - 2v_i) + 1 \times (\sum v_i - t) = 0$ (or $\neq 0$), $\mathcal{A}_2$ is able to answer the query by asking challenger token query for $(1 - 2v_1, ..., 1 - 2v_m, \sum v_i - t)$. We omit the proof in this paper.

4 Scheme for Inequality Threshold

There is a generic solution for solving the case of inequality threshold by using the idea from [10] which can be shown to be MC secure. The details of this generic solution are given in Appendix A. The ciphertext of this solution is of size $O(m2^m)$ although the token size is constant which is not practical. In the following, we provide two practical schemes to handle the inequality threshold version.

4.1 Scheme with known t_{max}

If we can know the maximum value for the threshold t, t_{max}, in advance, we can have a scheme which is better than the generic solution. The sizes of the ciphertext can be reduced to $O(\sum_{i=0}^{t_{max}+1}\binom{m}{i})$. In some applications, $t_{max} << m$ and is a constant. In that case, the size becomes $O(m^{t_{max}})$. The restriction on setting t_{max} seems to be quite stringent. At the end of this section, we show how one can update the ciphertext if the user decides to increase t_{max} without computing ciphertext from scratch. We first present the scheme for known t_{max}.

The idea behind our construction is based on the observation that hamming distance $H \leq t$ if and only if $H(H-1)...(H-t) = 0$. Then, if we evaluate $e(g,g)^{sH(H-1)...(H-t)}$ as $\texttt{Test}()$ result where "s" is a random number, when $H \leq t$, $\texttt{Test}()$ will be $1_{\mathbb{G}_T}$ (no information is leaked rather than the fact that $H \leq t$); when $H > t$, $H(H-1)...(H-t) \neq 0$, $\texttt{Test}()$ will output a random number (still no information is leaked rather than the fact $H > t$ since $\texttt{Test}() \neq 1_{\mathbb{G}_T}$ computationally). Note that although evaluating $H(H-1)...(H-t)$ seems trivial in performance, it helps to ensure no information can be leaked which is required in MC security level.

Since the formula $H(H-1)...(H-t)$ where $H = \sum x_i(1-2v_i) + \sum v_i$ contains both information from ciphertext (i.e. knowledge of x_i) and token (i.e. knowledge of v_i and t) which cannot be available at the same time, we need to split the formula to these two parts (ciphertext and token). Recall that as we discussed in Section 3, we can split the formula to $f(X)$ and $y(V,t)$ whose inner product $f(X) \cdot y(V,t)$ provides the result for $H(H-1) \cdot ... \cdot (H-t)$. The following lemma (proved in Appendix C) expands $H(H-1) \cdot ... \cdot (H-t)$ so that we can find $f(X)$ and $y(V,t)$. We let

$$H(H-1) \cdot ... \cdot (H-t) = a_{t+1}H^{t+1} + a_t H^t + ... + a_1 H \quad (1)$$

where we assume a_k $(k = 1, ..., t+1)$ can be efficiently determined.

Lemma 2. *Given attribute $X = (x_1, ..., x_m)$, target attribute $V = (v_1, ..., v_m)$ and threshold t, we denote H as the hamming distance $HammingDist(X,V)$ and define $a_0 = 0$ and b_j $(j = 0, ..., t+1)$ as*

$$b_j = a_{t+1}\binom{t+1}{t+1-j}(\sum v_i)^{t+1-j} + a_t\binom{t}{t-j}(\sum v_i)^{t-j} + ... + a_j\binom{j}{0} \quad (2)$$

Then, we have $H(H-1)\cdot ... \cdot(H-t)$

$$
\begin{aligned}
&= b_{t+1}(\sum_{k_1+...+k_m=t+1} \frac{(t+1)!}{k_1!\cdot ... \cdot k_m!}(1-2v_1)^{k_1}\cdot ... \cdot (1-2v_m)^{k_m} x_1^{k_1}\cdot ... \cdot x_m^{k_m}) \\
&+ ... + b_j(\sum_{k_1+...+k_m=j} \frac{j!}{k_1!\cdot ... \cdot k_m!}(1-2v_1)^{k_1}\cdot ... \cdot (1-2v_m)^{k_m} x_1^{k_1}\cdot ... \cdot x_m^{k_m}) \\
&+ ... + b_0
\end{aligned} \tag{3}
$$

Now, $H(H-1)\cdot ... \cdot(H-t)$ can be represented as inner product of $f(X)\cdot y(V,t) = \sum f_i(X)\cdot y_i(V,t)$. This is the key idea to our construction for inequality threshold. However, notice that $x_i \in \{0,1\}$, we would future reduce the number of items in equation (3) based on the observation that $x_1^{k_1}\cdot ... \cdot x_m^{k_m} = \prod_{\{i:k_i>0\}} x_i$ if $x_i \in \{0,1\}$. Then, equation (3) can be refined as:

$$
\begin{aligned}
& H(H-1)\cdot ... \cdot(H-t) \\
&= b_0 \\
&+ \sum_{1\le j\le m} [\sum_{1\le k_1\le t+1} b_{k_1}(1-2v_j)^{k_1}] x_j \\
&+ \sum_{1\le j_1<j_2\le m} [\sum_{k_1+k_2\le t+1; k_i\ge 1} \frac{(k_1+k_2)!}{k_1!k_2!} b_{k_1+k_2}(1-2v_{j_1})^{k_1}(1-2v_{j_2})^{k_2}] x_{j_1}x_{j_2} \\
&+ ... \\
&+ \sum_{1\le j_1<...<j_l\le m} [\sum_{k_1+...+k_l\le t+1; k_i\ge 1} \frac{(k_1+...+k_l)!}{k_1!...k_l!} b_{k_1+...+k_l}(1-2v_{j_1})^{k_1}...(1-2_{j_l})^{k_l}] x_{j_1}...x_{j_l} \\
&+ ... \\
&+ \sum_{1\le j_1<...<j_{t+1}\le m} [(t+1)! b_{t+1}(1-2v_{j_1})...(1-2v_{j_{t+1}})] x_{j_1}...x_{j_{t+1}} \qquad (4) \\
&= B_0 \\
&+ B_1x_1 + B_2x_2 + ... + B_mx_m \\
&+ B_{m+1}x_1x_2 + ... + B_{m+\binom{m}{2}}x_{m-1}x_m \\
&+ ... \\
&+ B_{m+\binom{m}{2}+...+\binom{m}{t}+1}x_1x_2...x_{t+1} + ... + B_{m+\binom{m}{2}+...+\binom{m}{t+1}}x_{m-t}...x_m
\end{aligned}
$$

Where B_i is defined as above. The number of items in equation (4) is $1 + \binom{m}{1} + \binom{m}{2} + ... + \binom{m}{t+1} = \sum_{i=0}^{t+1}\binom{m}{i}$. We now describe a construction based on [19] (see Appendix B) and equation (4) whose ciphertext and token size are both $O(\sum_{i=0}^{t_{max}+1}\binom{m}{i})$.

Recall ($\texttt{Setup}, \texttt{Enc}, \texttt{GenKey}, \texttt{Dec}$) in [19] can support n-dimension vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ such that $C \xleftarrow{\$} \texttt{Enc}(\boldsymbol{a})$ and $TK \xleftarrow{\$} \texttt{GenKey}(\boldsymbol{b})$ where $\texttt{Dec}(C,TK) = 1$ if and only if the inner product $\boldsymbol{a}\cdot\boldsymbol{b} = 0$. In our construction, we let n be $\sum_{i=0}^{t_{max}+1}\binom{m}{i}$. Encryption algorithm $\texttt{Encrypt}(X = x_1, ..., x_m)$ in our construction calls $\texttt{Enc}()$

with input vector:

$$\boldsymbol{a} = (1, x_1, ..., x_m, x_1x_2, ..., x_{m-1}x_m, ..., x_{m-t_{max}} \cdot ... \cdot x_m)^3 \tag{5}$$

Token for v_i and t is generated by calling GenKey() with input vector:

$$\boldsymbol{b} = (B_0, B_1, ..., B_m, B_{m+1}, ..., B_{m+\binom{m}{2}}, ..., B_{m+...+\binom{m}{t}+1}, ..., B_{m+...+\binom{m}{t+1}}, 0..., 0) \tag{6}$$

Note that $\boldsymbol{a}$ and $\boldsymbol{b}$ are constructed according to equation (4). Therefore, according to equation (4), the inner product of $\boldsymbol{a} \cdot \boldsymbol{b} = H(H-1) \cdot ... \cdot (H-t)$.

This construction has ciphertext and token both of size $O(n) = O(\sum_{i=0}^{t_{max}+1} \binom{m}{i})$, however, some items in the token are in fact "0" since t may be less than t_{max}; more specifically, $H(H-1)...(H-t)$ is $t+1$ degree and items in $\boldsymbol{a}$ whose degree larger than $t+1$ (i.e. $x_1x_2...x_{t+2}$ ... $x_{m-t_{max}}...x_m$) will have coefficient "0" in $\boldsymbol{b}$ (see equation (6)). This allows us further reduce the token size. To do so, we devise an encryption scheme slightly different from [19] such that $\boldsymbol{a}$ is still n-dimensional while $\boldsymbol{b}$ can be any n'-dimensional ($n' \leq n$) and decryption will output "1" if and only if the inner product $\sum_{i=1}^{n'} a_ib_i = 0$. We describe this construction as follows:

- Setup(1^n). It first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$. Then, it randomly selects g_p from $\mathbb{G}_p$, g_q from $\mathbb{G}_q$ and g_r from $\mathbb{G}_r$. It also randomly selects $\{h_{1,i}, h_{2,i}\}_{i\in[1,l_{max}]}$ from $\mathbb{G}_p$, and then randomly selects $R, \{R_{1,i}, R_{2,i}\}_{i\in[1,l_{max}]}$ from $\mathbb{G}_r$. It outputs

$$pk = \{g_p, g_r, Q = g_qR, [H_{1,i} = h_{1,i}R_{1,i}, H_{2,i} = h_{2,i}R_{2,i}]_{i\in[1,l_{max}]}\}$$

  and

$$sk = \{p, q, r, g_q, [h_{1,i}, h_{2,i}]_{i\in[1,l_{max}]}\}$$

- Encrypt($pk, X = x_1...x_i...x_{l_{max}}$). The encryption algorithm first randomly selects s, α, β from $\mathbb{Z}_N$ and $\{R'_{1,i}, R'_{2,i}\}_{i\in[1,l_{max}]}$ from $\mathbb{G}_r$. Then, it outputs ciphertext C:

$$\{C_0 = g_p^s, [C_{1,i} = H_{1,i}^sQ^{\alpha x_i}R'_{1,i}, C_{2,i} = H_{2,i}^sQ^{\beta x_i}R'_{2,i}]_{i\in[1,l_{max}]}\}$$

- GenTK($pk, sk, V = v_1...v_i...v_t$). Note that $t \leq l_{max}$. It randomly selects $\{r_{1,i}, r_{2,i}\}_{i\in[1,t]}$ and f_1, f_2 from $\mathbb{Z}_N$. Then, it randomly selects Q'' and R'' from $\mathbb{G}_q$ and $\mathbb{G}_r$ respectively. It outputs token TK:

[3] Note that although there exists x_1, x_2 and x_1x_2 in $\boldsymbol{a}$, given ciphertext for x_1 and x_2, we cannot reuse x_1 and x_2 to compute x_1x_2. This is because bilinear map is able to do only one multiplication (on encrypted data), however, we have used this ability to combine ciphertext and token, therefore, such redundancy in $\boldsymbol{a}$ seems to be necessary.

$$\left\{\begin{array}{c} K_0 = Q''R'' \prod_{i=1}^{\boxed{t}} h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}} \\ \left[K_{1,i} = g_p^{r_{1,i}} g_q^{f_1 v_i}, K_{2,i} = g_p^{r_{2,i}} g_q^{f_2 v_i}\right]_{i \in \boxed{[1,t]}} \end{array}\right\}$$

– `Test`(pk, TK, C). It computes $r = e(C_0, K_0) \prod_{i=1}^{\boxed{t}} e(C_{1,i}, K_{1,i}) e(C_{2,i}, K_{2,i})$. If $r = 1_{\mathbb{G}_T}$, it will output 1; otherwise it outputs 0.

Applying the above encryption scheme instead of the original scheme of [19] to (5) and (6), we obtain the final construction. Note that the above scheme also makes our security analysis of the final construction much easier (see Appendix F). The final construction Π_1 is described as follows.

– `Setup`(1^n): It first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$. Then, it randomly selects g_p from $\mathbb{G}_p$, g_q from $\mathbb{G}_q$ and g_r from $\mathbb{G}_r$. It also randomly selects $\{h_{1,l,i}, h_{2,l,i}\}_{l \in [1, t_{max}+1], i \in [1, \binom{m}{l}]}$ from $\mathbb{G}_p$. Then it randomly selects h_3, h_4 from $\mathbb{G}_p$. It also randomly selects R, $\{R_{1,l,i}, R_{2,l,i}\}_{l \in [1, t_{max}+1], i \in [1, \binom{m}{l}]}$, R_3, R_4 from $\mathbb{G}_r$. It outputs

$$pk = \left\{\begin{array}{c} g_p, g_r, Q = g_q R, \\ [H_{1,l,i} = h_{1,l,i} R_{1,l,i}, H_{2,l,i} = h_{2,l,i} R_{2,l,i}]_{l \in [1, t_{max}+1], i \in [1, \binom{m}{l}]}, \\ H_3 = h_3 R_3, H_4 = h_4 R_4 \end{array}\right\}$$

and

$$sk = \left\{\begin{array}{c} p, q, r, g_q, \\ [h_{1,l,i}, h_{2,l,i}]_{l \in [1, t_{max}+1], i \in [1, \binom{m}{l}]}, \\ h_3, h_4 \end{array}\right\}$$

– `Encrypt`$(pk, X = x_1 ... x_m)$: Encryption algorithm first randomly selects s, α, β from $\mathbb{Z}_N$ and $\{R'_{1,l,i}, R'_{2,l,i}\}_{l \in [1, t_{max}+1], i \in [1, \binom{m}{l}]}$, R'_3, R'_4 from $\mathbb{G}_r$. Then it outputs ciphertext C:

$$\left\{\begin{array}{c} C_0 = g_p^s, \\ \left[C_{1,l,i} = H_{1,l,i}^s Q^{\alpha x_{j_1} \ldots x_{j_l}} R'_{1,l,i}, C_{2,l,i} = H_{2,l,i}^s Q^{\beta x_{j_1} \ldots x_{j_l}} R'_{2,l,i}\right]_{l \in \{1 \ldots t_{max}+1\}; 1 \le j_1 < \ldots < j_l \le m}, \\ C_3 = H_3^s Q^{\alpha} R'_3, C_4 = H_4^s Q^{\beta} R'_4 \end{array}\right\}$$

– `GenTK`$(pk, sk, V = v_1 ... v_m, t)$: It randomly selects $\{r_{1,l,i}, r_{2,l,i}\}_{l \in [1, t+1], i \in [1, \binom{m}{l}]}$, r_3, r_4 and f_1, f_2 from $\mathbb{Z}_N$. Then, it randomly selects Q'' and R'' from $\mathbb{G}_q$ and $\mathbb{G}_r$ respectively. It outputs token TK:

$$\left\{\begin{array}{c} K_0 = Q''R''h_3^{-r_3}h_4^{-r_4}\prod_{l=1}^{t+1}\prod_{i=1}^{\binom{m}{l}} h_{1,l,i}^{-r_{1,l,i}}h_{2,l,i}^{-r_{2,l,i}} \\ \left[\begin{array}{cc} K_{1,1,1} = g_p^{r_{1,1,1}} g_q^{f_1 B_1}, & K_{2,1,1} = g_p^{r_{2,1,1}} g_q^{f_2 B_1} \\ \ldots & \ldots \\ K_{1,1,m} = g_p^{r_{1,1,m}} g_q^{f_1 B_m}, & K_{2,1,m} = g_p^{r_{2,1,m}} g_q^{f_2 B_m} \\ K_{1,2,1} = g_p^{r_{1,2,1}} g_q^{f_1 B_{m+1}}, & K_{2,2,1} = g_p^{r_{2,2,1}} g_q^{f_2 B_{m+1}} \\ \ldots & \ldots \\ K_{1,2,\binom{m}{2}} = g_p^{r_{1,2,\binom{m}{2}}} g_q^{f_1 B_{m+\binom{m}{2}}}, & K_{2,2,\binom{m}{2}} = g_p^{r_{2,2,\binom{m}{2}}} g_q^{f_2 B_{m+\binom{m}{2}}} \\ \ldots & \ldots \\ K_{1,t+1,1} = g_p^{r_{1,t+1,1}} g_q^{f_1 B_{m+\binom{m}{2}+\ldots+\binom{m}{t}+1}}, & K_{2,t+1,1} = g_p^{r_{2,t+1,1}} g_q^{f_2 B_{m+\binom{m}{2}+\ldots+\binom{m}{t}+1}} \\ \ldots & \ldots \\ K_{1,t+1,\binom{m}{t+1}} = g_p^{r_{1,t+1,\binom{m}{t+1}}} g_q^{f_1 B_{m+\ldots+\binom{m}{t+1}}}, & K_{2,t+1,\binom{m}{t+1}} = g_p^{r_{2,t+1,\binom{m}{t+1}}} g_q^{f_2 B_{m+\ldots+\binom{m}{t+1}}} \end{array}\right] \\ K_3 = g_p^{r_3} g_q^{f_1 B_0}, K_4 = g_p^{r_4} g_q^{f_2 B_0} \end{array}\right\}$$

– `Test`(pk, sk, TK, C) : It outputs 1 if $r = 1_{\mathbb{G}_T}$ and 0 otherwise, where $r = e(K_0, C_0)e(K_3, C_3)e(K_4, C_4)\prod_{l=1}^{t+1}\prod_{i=1}^{\binom{m}{l}} e(K_{1,l,i}, C_{1,l,i})e(K_{2,l,i}, C_{2,l,i})$.

The size of our ciphertext is still $O(\sum_{i=0}^{t_{max}+1}\binom{m}{i})$ but the size of token is now $O(\sum_{i=0}^{t+1}\binom{m}{i})$ for threshold t. The correctness follows from the fact tht r can be shown to be $e(g_q, g_q)^{(\alpha f_1+\beta f_2)H(H-1)(H-2)\ldots(H-t)}$. The security of the scheme is stated in Theorem 1 and proved in appendix F.

Theorem 1. *Our construction Π_1 in Section 4.1 is Selective-ID secure in the math-concealing model under Assumption 1.*

Lastly, to show that it is feasible to compute the coefficients a_k $(k = 1, ..., t+1)$ in (1), we have implemented an algorithm to calculate a_k. In fact, it can automatically calculate each non-zero elements of vector $\boldsymbol{b}$ in (6). It is written in C++. For example, with input $m = 100$ and $t = 3$, it took about 16 seconds to calculate all elements on an Intel Core 2 Due E6750 2.66GHz CPU platform.

Increasing t_{max}: It may be possible that the user wants to increase t_{max} to T'. The following shows the ideal of how to update the ciphertext without producing ciphertext from scratch, provided that the value α, β and s generated in `Encrypt`() procedure are kept by that user. The main ideal is that, when maximum threshold updates from t_{max} to T', the corresponding vector $\boldsymbol{a}$ in (5) will also need to be updated as:

$$\boldsymbol{a'} = (\boldsymbol{a}, x_1 \cdot \ldots \cdot x_{t_{max}+2}, \ldots, x_{m-T'} \cdot \ldots \cdot x_m) \tag{7}$$

Recall that when maximum threshold is t_{max}, $\boldsymbol{a}$ contains all items whose degree $l \leq t_{max}+1$; now when maximum threshold becomes T', $\boldsymbol{a}$ should contain items whose degree $l \leq T'+1$. Then, we need to produce those items whose degree is within $t_{max}+2$ to $T'+1$, namely $x_{j_1}\ldots x_{j_l}$ where $t_{max}+2 \leq l \leq T'+1$ and $1 \leq j_1 < \ldots < j_l \leq m$ in (7). If we can easily generate ciphertext corresponding to those items, we will update the ciphertext without from scratch. This can be done by calculating $C_{1,i} = H_{1,i}^s Q^{\alpha a_i''} R_{1,i}$ and $C_{2,i} = H_{2,i}^s Q^{\beta a_i''} R_{2,i}$ for $i = 1, ..., k$, given α, β and s. Where we denote vector $\boldsymbol{a''} = (a_1'', ..., a_k'')$ such that $\boldsymbol{a'} = (\boldsymbol{a}, \boldsymbol{a''})$.

The above update procedure can be shown to be MC-secure. Intuitively, when $x_1, ..., x_m$ in $\boldsymbol{a}$ are determined, all items in $\boldsymbol{a}$ (also in $\boldsymbol{a'}$) have been determined since they are the multiplication of two or more items in $\{x_1, ..., x_m\}$. For any possible $t_{max} \geq 0$, $\boldsymbol{a}$ (and therefore $\boldsymbol{a'}$) contains $(x_1, ..., x_m)$ for sure. That means all terms including the one to be generated due to the increase in t_{max} has been fixed although they are not computed yet. In other words, an adaptive attack will not work in our construction since it has no way to adaptively modify how the missing items are generated no matter what T' it provides. This is the intuitive reason that if the scheme for t_{max} is (selective) MC secure, then the update procedure is still (selective) MC secure. The formal proof and the details of how to perform this update can be found in Appendix H.

Note that in the worst case, $t_{max} = m$, the size of the ciphertext (and token) becomes $O(2^m)$. Although it is better than $O(m2^m)$ for the solution in Appendix A, it is not practical. So, this scheme should be used when t_{max} is small.

4.2 Scheme for Inequality Threshold with MR security

In this section, we consider another practical situation in which only a lower security level (the MR security) is required. Recall that the only difference between MC and MR security is that MR security does not require attribute X to be hidden after `Test`() outputs "1". MR security is still reasonable in many real applications. For example, as we mentioned before, the attribute X may be part of the data item M. When `Test`$(X) = 1$, we will properly decrypt M and see the entire X. It is meaningless to require MC security.

In fact, MR security does not define or limit how much information on attribute X can be leaked if `Test`$(X) = 1$. That is, we may leak the entire X in an encryption scheme which is still MR secure. This is the key to optimize the performance of such a scheme. We first show a general idea of how to make use of this property. Intuitively, we can decompose the predicate f (and therefore `Test`(X)) into T cases such that there exist MC-secure (or at least MR-secure) schemes for each case. Then, we can construct an MR secure scheme by encrypting data item M under each encryption scheme for each case. When deriving token, our scheme will generate decryption keys corresponding to those T different cases. If at least one of the cases decrypts successfully, `Test`() outputs "1". Otherwise, `Test`() outputs "0".

The above scheme can be shown to be MR secure. Since the scheme for each case is assumed to be at least MR secure, if `Test`$(X) = 0$, none of the scheme will leak any information about X except that X does not satisfy the query. In other words, given X_0 and X_1, it is not way to tell the difference which achieves MR security. On the other hand, if `Test`$(X) = 1$, how much information can be deduced depends on how we decompose the cases and what scheme is used in each case. Similarly, for performance, it depends on how we decompose the cases (i.e., the number of cases) and the decryption cost for each case. For example, if $\texttt{Test}()$ is equality threshold t testing of hamming distance, the following shows one approach (not a good approach) to decompose it into cases. There will be $\binom{m}{t}$ different $X \in \{0,1\}^m$ satisfying $HammingDist(X,V) = t$. We can decompose the equality threshold testing into $\binom{m}{t}$ cases (We assume they corresponding to $X_1, ..., X_{\binom{m}{t}}$ in $\{0,1\}^m$) that each case is testing whether X is the same as X_i, which can be done by Anonymous Identity-Based Encryption. Because Anonymous Identity-Based Encryption is MR (and also MC) secure, this scheme is MR secure. The scheme has an $O(\binom{m}{t} \times 1) = O(\binom{m}{t})$ decryption cost. And when $\texttt{Test}() = 1$, people can know exactly what X is.

Focusing on our construction in this section, we can simply decompose the predicate f of testing whether $HammingDist(X,V) \le t$ to t+1 cases that whether $HammingDist(X,V) = 0$, whether $HammingDist(X,V) = 1$, ..., and whether $HammingDist(X,V) = t$. Such scheme has decryption complexity of $O((t+1) \times m)$ if we use the (MC-secure) construction in Section 3.

Denote the scheme in Section 3 as $\Pi_3 = (\texttt{Setup}', \texttt{Encrypt}', \texttt{GenTK}', \texttt{Test}')$, we describe our scheme, $\Pi_4 = (\texttt{Setup} = \texttt{Setup}', \texttt{Encrypt} = \texttt{Encrypt}', \texttt{GenTK}', \texttt{Test}')$, with MR security as follows.

– $\texttt{GenTK}(pk, sk, V)$: For each $j \in \{0, ..., t\}$, it runs $TK_j = \texttt{GenTK}'(pk, sk, V, j)$ and returns $TK = (TK_0, ..., TK_t)$.

– $\texttt{Test}(pk, TK, C)$: For each TK_j in TK, it runs $\texttt{Test}'(pk, TK_j, C)$ and checks if any of the results for any j is $1_{\mathbb{G}_T}$, it returns 1, otherwise it returns 0 indicating that $HammingDist(X,V) > t$.

The size of the ciphertext in the above scheme is $O(m)$ and the size of the token for threshold t is $O((t+1)m)$.The correctness of the scheme follows directly from the correctness of the scheme in Section 3. The security of the scheme is given in the following theorem. The formal proof can be found in Appendix G based on a simple reduction (its intuitive ideal has been discussed in the beginning of this section). Recall that the scheme in section 3 is MC secure under Assumption 1.

Theorem 2. *Scheme in Section 4.2 is selective-ID secure in match-revealing model if scheme in Section 3 is selective-ID secure in match-concealing model.*

5 Conclusion

In this paper, we investigated the problem of predicate encryption with attribute protection and focus on a hamming distance similarity comparison predicate. We consider both the equality and inequality versions on a user-specific threshold t. For the equality version, we provide a MC-secure scheme with both the sizes of ciphertext and token equal to $O(m)$ where m is the length of the attribute vector. For the inequality version, we provide two practical schemes, one works for the situation when the maximum value of t (t_{max}) is known and is MC secure.

The other works for applications which require only MR security. The schemes provided in the paper should be applicable to real applications thus allowing the application communities to perform more useful computation on encrypted data.

The sizes of the ciphertext in our MC-secure scheme for the inequality threshold is $\sum_{i=0}^{t_{max}+1}\binom{m}{i}$. We leave it as an open problem whether it could be improved to $O((t+1)m)$ as in the MR-secure scheme.

References

1. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier and H. Shi.: Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. In Proceedings of CRYPTO'05.
2. A. Arasu, V. Ganti and R. Kaushik.: Efficient Exact Set-Similarity Joins. In Proceedings of VLDB'06.
3. R. Agrawal, J. Kiernan, R. Srikant and Y. Xu.: Order Preserving Encryption for Numeric Data. In Proceedings of SIGMOD'04.
4. M. Bellare, A. Boldyreva and A. O'neill.: Deterministic and Efficiently Searchable Encryption. In Proceedings of CRYPTO'07.
5. D. Boneh, G.D. Crescenzo, R. Ostrovsky, G. Persiano.: Public Key Encryption with keyword Search. In Proceedings of EUROCRYPT'04.
6. D. Boneh and M. Franklin.: Identity-Based Encryption from the Weil Pairing. In Proceedings of CRYPTO'01.
7. D. Boneh, E.J. Goh and K. Nissim.: Evaluating 2-DNF Formulas on Ciphertexts. In Proceedings of TCC'05.
8. D. Boneh and B. Waters.: A Fully Collusion Resistant Broadcast, Trace, and Revoke System. In Proceedings of CCS'06.
9. X. Boyen and B. Waters.: Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In Proceedings of CRYPTO'06.
10. D. Boneh and B. Waters.: Conjunctive, Subset, and Range Queries on Encrypted Data. In Proceedings of TCC'07.
11. S. Chaudhuri, V. Ganti and R. Kaushik.: A Primitive Operator for Similarity Joins in Data Cleaning. In Proceedings of ICDE'06.
12. R. Canetti, S. Halevi and J. Katz.: A forward-secure public-key encryption scheme. In Proceedings of EUROCRYPT'03.
13. E. Damiani, S.D.C. Vimercati, S. Jajodia, S. Paraboschi and P. Samarati.: Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs. In Proceedings of CCS'03.
14. C. Gentry.: Practical Identity-Based Encryption Without Random Oracles. In Proceedings of EUROCRYPT'06.
15. V. Goyal, O. Pandey, A. Sahai and B. Waters.: Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data. In Proceedings of CCS'06.
16. H. Hacigumus, B. Iyer and S. Mehrotra.: Providing Database as a Service. In Proceedings of ICDE'02.
17. H. Hacigumus, B. Iyer, C. Li and S. Mehrotra.: Executing SQL over Encrypted Data in the Database-Service-Provider Model. In Proceedings of SIGMOD'02.
18. B. Hore, S. Mehrotra and G. Tsudik.: A Privacy-Preserving Index for Range Queries. In Proceedings of VLDB'04.

19. J. Katz, A. Sahai and B. Waters.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In Proceedings of EUROCRYPT'08.
20. P.P. van Liesdonk.: Anonymous and Fuzzy Identity-Based Encryption. Master thesis, Eindhoven University.
21. H.A. Park, B.H. Kim, D.H. Lee, Y.D. Chung and J. Zhang.: Secure Similarity Search. In Proceedings of IEEE International Conference on Granular Computing'07.
22. A. Shamir.: Identity-Based Cryptosystems and Signature Schemes. In Proceedings of CRYPTO'84.
23. E. Shen, E. Shi and B. Waters.: Predicate Privacy in Encryption Systems. In Proceedings of TCC'09.
24. E. Shi, J. Bethencourt, T-H.H. Chan, D. Song and A. Perrig.: Multi-Dimensional Range Query over Encrypted Data. In Proceedings of IEEE Symposium on Security and Privacy'07.
25. A. Sahai and B. Waters.: Fuzzy Identity-Based Encryption. In Proceedings of EUROCRYPT'05.
26. D.X. Song, D. Wagner and A. Perrig.: Practical Techniques for Searches on Encrypted Data. In Proceedings of IEEE Symposium on Security and Privacy'00.
27. W.K. Wong, D.W. Cheung, B. Kao and N. Mamoulis.: Secure k-NN Computation on Encrypted Databases. In Proceedings of SIGMOD'09.

A A generic construction from [10]

The main idea of this construction is that we generate a ciphertext $(C_0, C_1, ...C_{t_{max}})$ for each possible $V \in \{0,1\}^m$ where we assume that the vector size is m. For each $j \in \mathbb{Z}_{t_{max}+1}$, if $HammingDist(X, V) \leq j$, then C_j will be an encrypted message for "true" based on an IND-CPA secure encryption scheme; otherwise, C_j will be an encrypted message for "false". When we `Test`() for a certain (V, t), we can simply find the ciphertext for V and decrypt the t-th element C_t in that ciphertext. If $HammingDist(X, V) \leq t$, the decryption result should be "true". More specifically, we define the encryption scheme as follows. Let $(\mathtt{G}, \mathtt{E}, \mathtt{D})$ be an IND-CPA secure encryption scheme.

- $\mathtt{Setup}(1^n)$: Run $\mathtt{G}(1^n)$ to generate $(pk_{l,j}, sk_{l,j})_{\{l\in\{0,1\}^m, j\in\mathbb{Z}_{t_{max}+1}\}}$ for $(t_{max}+1)2^m$ times. Return the public-key pk as $\{pk_{l,j}\}_{\{l\in\{0,1\}^m, j\in\mathbb{Z}_{t_{max}+1}\}}$ and the secret key sk as $\{sk_{l,j}\}_{\{l\in\{0,1\}^m, j\in\mathbb{Z}_{t_{max}+1}\}}$.

- $\mathtt{Encrypt}(pk, X = x_1...x_m)$: For each $l \in \{0,1\}^m$, return $(C_0, C_1, ..., C_{t_{max}})_l$ where

$$C_j = \begin{cases} \mathtt{E}_{pk_{l,j}}(``true'') & \text{if } HammingDist(X, l) \leq j; \\ \mathtt{E}_{pk_{l,j}}(``false'') & \text{otherwise.} \end{cases}$$

- $\mathtt{GenTK}(pk, sk, V, t)$: Return $sk_{V,t}$ as the token.

- $\mathtt{Test}(pk, TK, C)$: It first finds $(C_0, C_1, ..., C_m)_V$ and computes $r = \mathtt{D}_{TK}(C_t)$. If r is equal to "true" then return 1; otherwise return 0.

The security of the above solution comes from the IND-CPA secure encryption scheme we used [4], see appendix A of [10] for more details. We can rearrange the ciphertexts so that $C_{l,j}$ in our solution is corresponding to $C_{(t_{max}+1)l+j}$ in [10]'s proof and the rest of the proof is the same. We should also note that $(t_{max}+1)2^m$ should be $poly(n)$ in security parameter n because all algorithms given above should be polynomial-time.

B Predicate-only encryption in [19]

[19] proposes a predicate encryption scheme for inner product predicate. Below we describe its predicate-only version used in this paper for completeness. The scheme consists of $(\texttt{Setup}, \texttt{Enc}, \texttt{GenKey}, \texttt{Dec})$:

- $\texttt{Setup}(1^n)$: The setup algorithm first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$. Next, it computes g_p, g_q and g_r as generators of $\mathbb{G}_p, \mathbb{G}_q$ and $\mathbb{G}_r$, respectively. It then chooses $R_{1,i}, R_{2,i} \in \mathbb{G}_r$ and $h_{1,i}, h_{2,i} \in \mathbb{G}_p$ uniformly at random for $i = 1$ to n, and $R_0 \in \mathbb{G}_r$ uniformly at random. The public parameters include $(N = pqr, \mathbb{G}, \mathbb{G}_T, e)$ along with:

$$PK = (g_p, g_r, Q = g_q \cdot R_0, \{H_{1,i} = h_{1,i} \cdot R_{1,i}, H_{2,i} = h_{2,i} \cdot R_{2,i}\}_{i=1}^n).$$

  The master secret key SK is $(p, q, r, g_q, \{h_{1,i}, h_{2,i}\}_{i=1}^n)$.

- $\texttt{Enc}(PK, \boldsymbol{x})$: Let $\boldsymbol{x} = (x_1, ..., x_n)$ where $x_i \in \mathbb{Z}_N$. This algorithm chooses random $s, \alpha, \beta \in \mathbb{Z}_N$ and $R_{3,i}, R_{4,i} \in \mathbb{G}_r$ $(i = 1, ..., n)$. It outputs the ciphertext

$$C = (C_0 = g_p^s, \{C_{1,i} = H_{1,i}^s \cdot Q^{\alpha \cdot x_i} \cdot R_{3,i}, C_{2,i} = H_{2,i}^s \cdot Q^{\beta x_i} R_{4,i}\}_{i=1}^n).$$

- $\texttt{GenKey}(SK, \boldsymbol{v})$: Let $\boldsymbol{v} = (v_1, ..., v_n)$, and recall $SK = (p, q, r, g_q, \{h_{1,i}, h_{2,i}\}_{i=1}^n)$. This algorithm chooses random $r_{1,i}, r_{2,i} \in \mathbb{Z}_p$ for $i = 1$ to n, random $R_5 \in \mathbb{G}_r$, random $f_1, f_2 \in \mathbb{Z}_q$ and random $Q_6 \in \mathbb{G}_q$. It then outputs

$$SK_{\boldsymbol{v}} = (K = R_5 \cdot Q_6 \cdot \textstyle\prod_{i=1}^n h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}}, \{K_{1,i} = g_p^{r_{1,i}} g_q^{f_1 \cdot v_i}, K_{2,i} = g_p^{r_{2,i} g_q^{f_2 v_i}}\}_{i=1}^n).$$

- $\texttt{Dec}(SK_{\boldsymbol{v}}, C)$: Let $C = (C_0, \{C_{1,i}, C_{2,i}\}_{i=1}^n)$ and $SK_v = (K, K_{1,i}, K_{2,i}{}_{i=1}^n)$ be as above. The decryption algorithm outputs 1 if and only if

$$e(C_0, K) \cdot \textstyle\prod_{i=1}^n e(C_{1,i}, K_{1,i}) \cdot e(C_{2,i}, K_{2,i}) \stackrel{?}{=} 1.$$

[4] The main idea is that we cannot distinguish $E(pk_i, \text{“}true\text{”})$ from $E(pk_i, \text{“}false\text{”})$ in the case of $HammingDist(X_0, V) \leq t$ but $HammingDist(X_1, V) > t$.

C Proof of Lemma 2

By Binomial theorem, we have

$$\begin{aligned} H^k &= (\sum(x_i(1-2v_i)+v_i))^k = (\sum x_i(1-2v_i)+\sum v_i)^k \\ &= (\sum x_i(1-2v_i))^k + ... + \binom{k}{j}(\sum x_i(1-2v_i))^{k-j}(\sum v_i)^j + \\ &\quad ... + (\sum v_i)^k \end{aligned} \tag{8}$$

If we substitute equation (8) for each H^k $(k = 1, ..., t+1)$ in (1), we have

$$\begin{aligned} &H(H-1)\cdot ....\cdot(H-t) \\ &= a_{t+1}[(\sum x_i(1-2v_i))^{t+1} + \binom{t+1}{1}(\sum v_i)(\sum x_i(1-2v_i))^t + ... + (\sum v_i)^{t+1}] \\ &+ ... \\ &+ a_1[\sum x_i(1-2v_i) + \sum v_i] \\ &= b_{t+1}(\sum x_i(1-2v_i))^{t+1} + b_t(\sum x_i(1-2v_i))^t + ... + b_0 \end{aligned} \tag{9}$$

Focusing on each $(\sum x_i(1-2v_i))^l = (x_1(1-2v_1) + ... + x_m(1-2v_m))^l$ $(l = 1, ..., t+1)$ in (9), by applying Multinomial theorem, we have

$$(\sum x_i(1-2v_i))^l = \sum_{k_1+...+k_m=l} \frac{l!}{k_1!...k_m!}(1-2v_1)^{k_1}\cdot ...\cdot(1-2v_m)^{k_m}x_1^{k_1}\cdot ...\cdot x_m^{k_m} \tag{10}$$

We substitute (10) in equation (9), which completes the proof.

D Proof of Lemma 1

Proof. Since $X = x_1, ..., v_i, ..., x_m$ and $V = v_1, ..., v_i, ..., v_m$ are bit vectors, their hamming distance are equal to the summation of exclusive-or ("$\oplus$") at each position i. Keep in mind that we want to represent hamming distance in inner product form and therefore we use the formula that $a \oplus b = a + b - 2ab = a(1-2b) + 1 \times b$ if $a, b \in \{0, 1\}$. Since the number of items will heavily impact on sizes of ciphertext, token and decryption cost, we also want the number can be as small as possible. This motivates us to represent $\sum_{i=1}^m [x_i(1-2v_i)+v_i] = \sum_{i=1}^m x_i(1-2v_i) + 1 \times \sum_{i=1}^m v_i$. The details are as follows.

$$\begin{aligned} HammingDist(X,V) &= \sum_{i=1}^m x_i \oplus v_i = \sum_{i=1}^m (x_i + v_i - 2x_iv_i) \\ &= \sum_{i=1}^m (x_i(1-2v_i)+v_i) \\ &= \sum_{i=1}^m x_i(1-2v_i) + 1 \times (\sum_{i=1}^m v_i) \end{aligned} \tag{11}$$

E Correctness analysis of the scheme for equality threshold

One key property behind the scheme in [19] is that if two elements a and b come from two different (prime) subgroups of $\mathbb{G}$, then $e(a,b) = 1_{\mathbb{G}_T}$. To simplify our correctness analysis, since the token of encryption scheme in Section 3 only involves elements from $\mathbb{G}_p$ and $\mathbb{G}_q$, we investigate the value of $r = \mathtt{Test}(pk, TK, C)$ in subgroup $\mathbb{G}_p$ and $\mathbb{G}_q$ respectively only.

$$
\begin{aligned}
r_q =\ & \\
& e(g_q^{\alpha}, g_q^{-f_1(\sum v_i - t)}) e(g_q^{\beta}, g_q^{-f_2(\sum v_i - t)}) \cdot 1 \cdot \prod e(g_q^{\alpha x_i}, g_q^{f_1(1-2v_i)}) e(g_q^{\beta x_i}, g_q^{f_2(1-2v_i)}) \\
& = e(g_q, g_q)^{(\alpha f_1 + \beta f_2)(\sum v_i - t + \sum (1-2v_i) x_i)} \\
& = e(g_q, g_q)^{(\alpha f_1 + \beta f_2)(HammingDist(x_1 \ldots x_i \ldots x_m, v_1 \ldots v_i \ldots v_m) - t)}
\end{aligned}
$$

and

$$
r_p = e(h_3^s, g_p^{r_3}) e(h_4^s, g_p^{r_4}) e(g_p^s, h_3^{-r_3}) e(g_p^s, h_4^{-r_4}) \prod e(g_p^s, h_{1,i}^{-r_{1,i}}) e(g_p^s, h_{2,i}^{-r_{2,i}}) \cdot \prod e(h_{1,i}^s, g_p^{r_{1,i}}) e(h_{2,i}^s, g_p^{r_{2,i}}) = 1
$$

F Proof of Theorem 1

Definition 4. *The encryption scheme* $\Pi_2 = (\mathtt{Setup}, \mathtt{Encrypt}, \mathtt{GenTK}, \mathtt{Test})$ *defined in Section 4.1 is MC secure if for all probabilistic polynomial-time Turing machine (adversary)* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *in the following game is negligible.*

Setup: The adversary $\mathcal{A}(1^n)$ outputs two possible equal-length (l_{max}-length) vectors X_0 and X_1 to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ takes a security parameter n and runs *Setup* to generate pk and sk. $\mathcal{C}$ sends pk to $\mathcal{A}$.
Challenge: The challenger $\mathcal{C}$ picks a random bit $b \in \{0,1\}$, computes and returns $C^* \xleftarrow{\$} \mathtt{Encrypt}(pk, X_b)$ to adversary $\mathcal{A}$.
Phase 1: The adversary $\mathcal{A}$ may adaptively request polynomially bounded numbers of tokens ("TK") for any $V_i = v_{i,1}...v_{i,t_i}$ where $t_i \leq l_{max}$, subject to the restriction that $0 = \sum_{k=1}^{t_i} x_{j,k} v_{i,k}$ for both $j = 1, 0$ or $0 \neq \sum_{k=1}^{t_i} x_{j,k} v_{i,k}$ for both $j = 0, 1$.
Guess: The adversary $\mathcal{A}$ outputs a guess bit b'. The advantage $\mathbf{Adv}^{\mathsf{MC}}_{\Pi_2, \mathcal{A}}(n)$ of $\mathcal{A}$ is defined as $|\Pr[b' = b] - \frac{1}{2}|$.

Lemma 3. *Our construction* Π_1 *in Section 4.1 is Selective-ID secure in the math-concealing model if* $\Pi_2 = (\mathtt{Setup}, \mathtt{Encrypt}, \mathtt{GenTK}, \mathtt{Test})$ *in Section 4.1 is secure under definition 4.*

Proof. This proof is by contradiction. We will show that if there exist Adversary $\mathcal{A}_1$ with non-negligible adversary ϵ with our construction Π_1, then we can construct an Adversary $\mathcal{A}_2$ also with non-negligible with scheme Π_2 under definition 4.

Setup: The adversary $\mathcal{A}_2(1^n)$ runs $\mathcal{A}_1(1^n)$. $\mathcal{A}_1$ outputs two equal-length vector $X_0 = x_{0,1}...x_{0,k}...x_{0,m}$ and $X_1 = x_{1,1}...x_{1,k}...x_{1,m}$. $\mathcal{A}_1$ passes X_0 and X_1 to $\mathcal{A}_2$. $\mathcal{A}_1$ also submits $t_{max} \leq m$ to $\mathcal{A}_2$.

The adversary $\mathcal{A}_2$ calculates $\tilde{X}_j = \boldsymbol{a}$ in (5):

$$\tilde{X}_j = (1, x_{j,1}, ..., x_{j,m}, x_{j,1}x_{j,2}, ..., x_{j,m-1}x_{j,m}, ..., x_{j,m-t_{max}} \cdot ... \cdot x_{j,m})$$

for both $j = 0, 1$. Then, $\mathcal{A}_2$ submits $\tilde{X}_0$ and $\tilde{X}_1$ to challenger $\mathcal{C}$. Note that $l_{max} = \sum_{l=0}^{t_{max}+1} \binom{m}{l}$.

Challenge: The challenger $\mathcal{C}$ runs $\mathtt{Setup}(1^n)$ to generate pk and sk; $\mathcal{C}$ sends pk to adversary $\mathcal{A}_2$. $\mathcal{A}_2$ rearranges pk into pk' according to the description of *Setup*() in scheme Π_1. $\mathcal{A}_2$ passes pk' to $\mathcal{A}_1$.

The challenger picks a random bit $b \in \{0, 1\}$. $\mathcal{C}$ computes and returns $C^* \xleftarrow{\$} \mathtt{Encrypt}(pk, \tilde{X}_b)$ to $\mathcal{A}_2$. Adversary $\mathcal{A}_2$ rearranges C^* and sends it to $\mathcal{A}_1$.

Phase 1: The adversary $\mathcal{A}_1$ may adaptively request polynomially bounded number of tokens for any (V_i, t_i) subject to the restriction that $t_i < HammingDist(V_i, X_j)$ for both $j = 0, 1$ or $t_i \geq HammingDist(V_i, X_j)$ for both $j = 0, 1$. When receiving valid $(V = v_1...v_m, t)$, $\mathcal{A}_2$ will calculate a $\tilde{t} = \sum_{l=0}^{t+1} \binom{m}{l}$-length vector $\tilde{V} = (a_{t+1}(\sum v_i)^{t+1} + ... + a_1(\sum v_i), ..., \sum_{1 \leq k_1 + ... + k_l \leq t+1; k_i \geq 1} \frac{(k_1+...+k_l)!}{k_1!...k_l!} b_{k_1+...+k_l}(1-2v_{s_1})^{k_1}...(1-2v_{s_l})^{k_l}, ..., (t+1)! b_{t+1}(1-2v_{m-t})(1-2v_{m-t+1})...(1-2v_m))$ We note that $HammingDist(V, X_j) \leq t$ if and only if $\sum_{k=1}^{\tilde{t}} \tilde{X}_{j,k}\tilde{V}_k = 0$ as we explained in Section 4.1. $\mathcal{A}_2$ submits $\tilde{V}$ to the challenger $\mathcal{C}$ to acquire a token TK. $\mathcal{A}_2$ rearranges TK and sends it to $\mathcal{A}_1$.

Guess: $\mathcal{A}_1$ outputs a bit b'. And $\mathcal{A}_2$ passes b' to the challenger as its output.

Recall that $HammingDist(V, X_j) \leq t$ if and only if $\sum_{k=1}^{\tilde{t}} \tilde{X}_{j,k}\tilde{V}_k = 0$, therefore valid token requests for Π_1 are still valid in Π_2. And $\mathbf{Adv}^{\mathsf{MC}}_{\Pi_2,\mathcal{A}_2}(n) = \mathbf{Adv}^{\mathsf{MC}}_{\Pi_1,\mathcal{A}_1}(n)$. That completes our proof.

Lemma 4. $\Pi_2 = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{GenTK}, \mathsf{Test})$ *in Section 4.1 is secure under Definition 4 under Assumption 1.*

Proof. This proof is quite similar to the proof in [19]. Given two equal-length vector $X = x_1...x_i...x_{l_{max}}$ and $Y = y_1...y_i...y_{l_{max}}$, loosely speaking, we try to prove $(X, X) \stackrel{c}{\equiv} (X, 0) \stackrel{c}{\equiv} (X, Y) \stackrel{c}{\equiv} (0, Y) \stackrel{c}{\equiv} (Y, Y)$ in the game defined by Definition 1. Where 0 stands for a l_{max}-length vector $(0, 0, ..., 0)$ and $\stackrel{c}{\equiv}$ is "computationally indistinguishable".

Let us prove $(X, X) \stackrel{c}{\equiv} (X, 0)$ first. Given $\{\bar{Z}, T\}$ where T may be equal to $T_1 = g_p^{b^2 s} R_3$ or $T_2 = g_p^{b^2 s} Q_3 R_3$, the challenger $\mathcal{C}$ answers $\mathtt{Setup}(1^n)$ as follow: It randomly selects $\omega_{1,i}$ and $\omega_{2,i}$ from $\mathbb{Z}_N$. Then, it computes $h_{1,i} = h_p^{x_i} g_p^{\omega_{1,i}} = g_p^{bx_i + \omega_{1,i}}$ and $h_{2,i} = k_p^{x_i} g_p^{\omega_{2,i}} = g_p^{b^2 x_i + \omega_{2,i}}$ for $i = 1, ..., l_{max}$. It outputs pk to adversary $\mathcal{A}$: where $R_{1,i}$ and $R_{2,i}$ are randomly selected from $\mathbb{G}_r$.

$$pk = \{g_p, g_r, Q = g_p R_1, [H_{1,i} = h_{1,i} R_{1,i}, H_{2,i} = h_{2,i} R_{2,i}]_{i=1,...,l_{max}}\}$$

The challenging ciphertext C^* is generated as follow: challenger $\mathcal{C}$ first randomly selects $R'_{1,i}$ and $R'_{2,i}$ from $\mathbb{G}_r$.

$$C = \{C_0 = g_p^s, [C_{1,i} = (g_p^{bs} Q_2 R_2)^{x_i} (g_p^s)^{\omega_{1,i}} R'_{1,i}, C_{2,i} = (T)^{x_i} (g_p^s)^{\omega_{2,i}} R'_{2,i}]_{i=1,\ldots,l_{max}}\}$$

We note that $C_{1,i} = (g_p^{bs} Q_2 R_2)^{x_i} (g_p^s)^{\omega_{1,i}} R'_{1,i} = (g_p^{bx_i+\omega_{1,i}})^s Q_2^{x_i} R_2^{x_i} R'_{1,i} = h_{1,i}^s Q^{\alpha x_i} R_2^{x_i} R'_{1,i}$. where we denote $Q_2 = g_q^\alpha$. And $C_{2,i} = (T)^{x_i} (g_p^s)^{\omega_{2,i}} R'_{2,i} = (g_p^{b^2 x_i+\omega_{2,i}})^s Q^{\beta x_i} R_3^{x_i} R'_{2,i}$ $= h_{2,i}^s Q^{\beta x_i} R_3^{x_i} R'_{2,i}$ where $\beta = 0$ if $T = T_1$ and β is random from $\mathbb{Z}_N$ if $T = T_2$.

When receiving $V = v_1 ... v_i ... v_t$ from adversary, challenger $\mathcal{C}$ generates corresponding token as follows: It firstly randomly selects $\tilde{f}_1$ and $\tilde{f}_2$ from $\mathbb{Z}_N$. It also randomly chooses $r'_{1,i}$ and $r'_{2,i}$ form $\mathbb{Z}_N$. Then, it calculates $K_{1,i} = (g_p^a g_q)^{\tilde{f}_1 v_i} (g_p^{ab} Q_1)^{-\tilde{f}_2 v_i} g_p^{r'_{1,i}} = g_p^{a\tilde{f}_1 v_i - ab\tilde{f}_2 v_i + r'_{1,i}} g_q^{\tilde{f}1 v_i - d\tilde{f}_2 v_i}$. (We denote $Q_1 = g_q^d$.) We denote $r_{1,i} = a\tilde{f}_1 v_i - ab\tilde{f}_2 v_i + r'_{1,i}$ and $f_1 = \tilde{f}1 - d\tilde{f}_2$.

It also calculates $K_{2,i} = (g_p^a g_q)^{\tilde{f}_2 v_i} g_p^{r'_{2,i}} = g_p^{a\tilde{f}_2 v_i + r'_{2,i}} g_q^{\tilde{f}_2 v_i}$ where we denote $r_{2,i} = a\tilde{f}_2 v_i + r'_{2,i}$ and $f_2 = \tilde{f}_2$.

K_0 is calculated by $K_0 = Q'' R'' \prod_{i=1}^t h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}}$
$= QR \prod_{i=1}^t g_p^{-(bx_i+\omega_{1,i})(a\tilde{f}_1 v_i - ab\tilde{f}_2 v_i + r'_{1,i})} g_p^{-(b^2 x_i + \omega_{2,i})(a\tilde{f}_2 v_i + r'_{2,i})}$
$= QR \prod_{i=1}^t g_p^{-(ab\tilde{f}_1 x_i v_i - ab^2 \tilde{f}_2 x_i v_i + br'_{1,i} x_i + a\tilde{f}_1 \omega_{1,i} v_i - ab\tilde{f}_2 \omega_{1,i} v_i + r'_{1,i}\omega_{1,i} + ab^2 \tilde{f}_2 v_i x_i + b^2 r'_{2,i} x_i + a\tilde{f}_2 \omega_{2,i} v_i + r'_{2,i}\omega_{2,i})}$
$= QR \prod_{i=1}^t g_p^{-a(\tilde{f}_1 \omega_{1,i} v_i + \tilde{f}_2 \omega_{2,i} v_i) - ab(\tilde{f}_1 x_i v_i - \tilde{f}_2 \omega_{1,i} v_i) - b(r'_{1,i} x_i) - b^2 (r'_{2,i} x_i) - ab^2(-\tilde{f}_2 v_i x_i + \tilde{f}_2 v_i x_i) - (r'_{1,i}\omega_{1,i} + r'_{2,i}\omega_{2,i})}$
$= QR \prod_{i=1}^t [(g_p^a g_q)^{-(\tilde{f}_1 \omega_{1,i} v_i + \tilde{f}_2 \omega_{2,i} v_i)} \cdot (g_p^{ab} Q_1)^{-(\tilde{f}_1 x_i v_i - \tilde{f}_2 \omega_{1,i} v_i)} \cdot h_p^{-(r'_{1,i} x_i)} \cdot k_p^{-(r'_{2,i} x_i)} \cdot$
$g_p^{-(r'_{1,i}\omega_{1,i} + r'_{2,i}\omega_{2,i})}]$

Now, let us prove $(X, 0) \overset{c}{\equiv} (X, Y)$. The challenger $\mathcal{C}$ answers `Setup`(1^n) as follows: It first randomly selects $\omega_{1,i}$ and $\omega_{2,i}$ from $\mathbb{Z}_N$. It randomly selects $R_{1,i}$ and $R_{2,i}$ from $\mathbb{G}_r$ Then, it calculates $h_{1,i} = h_p^{x_i} g_p^{\omega_{1,i}} = g_p^{bx_i+\omega_{1,i}}$ and $h_{2,i} = k_p^{y_i} g_p^{\omega_{2,i}} = g_p^{b^2 y_i + \omega_{2,i}}$. It outputs pk to adversary:

$$pk = \{g_p, g_r, Q = g_q R_1, [H_{1,i} = h_{1,i} R_{1,i}, H_{2,i} = h_{2,i} R_{2,i}]_{i=1\ldots l_{max}}\}$$

The challenging ciphertext C^* is generated as follows: challenger $\mathcal{C}$ first randomly select $R'_{1,i}$ and $R'_{2,i}$ from $\mathbb{G}_r$.

$$C = \{C_0 = g_p^s, [C_{1,i} = (g_p^{bs} Q_2 R_2)^{x_i} (g_p^s)^{\omega_{1,i}} R'_{1,i}, C_{2,i} = (T)^{y_i} (g_p^s)^{\omega_{2,i}} R'_{2,i}]_{i=1\ldots l_{max}}\}$$

We note that $C_{1,i} = (g_p^{bx_i+\omega_{1,i}})^s Q_2^{x_i} R_2^{x_i} R'_{1,i} = h_{1,i}^s Q^{\alpha x_i} R_2^{x_i} R'_{1,i}$ And $C_{2,i} = (g_p^{b^2 y_i + \omega_{2,i}})^s Q_3^{y_i} R_3^{y_i} R'_{2,i} = h_{2,i}^s Q^{\beta y_i} R_3^{y_i} R'_{2,i}$ where we denote $Q_3 = g_q^\alpha$. $\beta = 0$ if $T = T_1$ and β is random number in $\mathbb{Z}_N$ if $T = T_2$.

When receiving $V = v_1 ... v_i ... v_t$ from adversary $\mathcal{A}$, the challenger $\mathcal{C}$ generates corresponding token as follows. According to Definition 4, V should be subject to (i) $\sum_{i=1}^t y_i v_i = 0 = \sum_{i=1}^t x_i v_i$ or (ii) $\sum_{i=1}^t y_i v_i \neq 0$ and $\sum_{i=1}^t x_i v_i \neq 0$. We handle token generation in this two conditions separately.

Case (i) $\sum_{i=1}^t y_i v_i = 0 = \sum_{i=1}^t x_i v_i$:

The challenger first randomly selects $\tilde{f}_1$, $\tilde{f}_2$ and $r'_{1,i}$, $r'_{2,i}$ from $\mathbb{Z}_N$. Then, it calculates $K_{1,i} = (g_p^a g_q)^{\tilde{f}_1 v_i} g_p^{r'_{1,i}} = g_p^{a\tilde{f}_1 v_i + r'_{1,i}} g_q^{\tilde{f}_1 v_i}$. We denote $r_{1,i} = a\tilde{f}_1 v_i + r'_{1,i}$ and $f_1 = \tilde{f}_1$. It also calculates $K_{2,i} = (g_p^a g_q)^{\tilde{f}_2 v_i} g_p^{r'_{2,i}} = g_p^{a\tilde{f}_2 v_i + r'_{2,i}} g_q^{\tilde{f}_2 v_i}$ where we denote $r_{2,i} = a\tilde{f}_2 v_i + r'_{2,i}$ and $f_2 = \tilde{f}_2$.

K_0 is calculated by $K_0 = Q''R'' \prod_{i=1}^t h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}}$
$= QR \prod_{i=1}^t g_p^{-(bx_i+\omega_{1,i})(a\tilde{f}_1 v_i + r'_{1,i})} g_p^{-(b^2 y_i+\omega_{2,i})(a\tilde{f}_2 v_i + r'_{2,i})}$
$= QR \prod_{i=1}^t g_p^{-(ab\tilde{f}_1 x_i v_i + br'_{1,i} x_i + a\omega_{1,i}\tilde{f}_1 v_i + r'_{1,i}\omega_{1,i} + ab^2 \tilde{f}_2 y_i v_i + b^2 r'_{2,i} y_i + a\tilde{f}_2 \omega_{2,i} v_i + r'_{2,i}\omega_{2,i})}$
$= QR \prod_{i=1}^t g_p^{-a(\omega_{1,i}\tilde{f}_1 v_i + \tilde{f}_2 \omega_{2,i} v_i)} g_p^{-b(r'_{1,i} x_i)} g_p^{-b^2(r'_{2,i} y_i)} g_p^{-(r'_{1,i}\omega_{1,i} + r'_{2,i}\omega_{2,i})}$ since $\sum_{i=1}^t y_i v_i = 0 = \sum_{i=1}^t x_i v_i$. Then, $K_0 = QR \prod_{i=1}^t (g_p^a g_q)^{-(\omega_{1,i}\tilde{f}_1 v_i + \tilde{f}_2 \omega_{2,i} v_i)} h_p^{-(r'_{1,i} x_i)} k_p^{-(r'_{2,i} y_i)} g_p^{-(r'_{1,i}\omega_{1,i} + r'_{2,i}\omega_{2,i})}$

Case (ii) $\sum_{i=1}^t y_i v_i = c_y \neq 0$ and $\sum_{i=1}^t x_i v_i = c_x \neq 0$:

The challenger firstly randomly chooses $\tilde{f}_1$, $\tilde{f}_2$ and $r'_{1,i}$, $r'_{2,i}$ from $\mathbb{Z}_N$. Then, it calculates $K_{1,i} = (g_p^a g_q)^{\tilde{f}_1 v_i} (g_p^{ab} Q_1)^{-c_y \tilde{f}_2 v_i} g_p^{r'_{1,i}} = g_p^{a\tilde{f}_1 v_i - abc_y \tilde{f}_2 v_i + r'_{1,i}} g_q^{\tilde{f}_1 v_i - dc_y \tilde{f}_2 v_i}$ where we denote $Q_1 = g_q^d$. And we denote $r_{1,i} = a\tilde{f}_1 v_i - abc_y \tilde{f}_2 v_i + r'_{1,i}$ and $f_1 = \tilde{f}_1 - dc_y \tilde{f}_2$. It also calculates $K_{2,i} = (g_p^a g_q)^{c_x \tilde{f}_2 v_i} g_p^{r'_{2,i}} = g_p^{ac_x \tilde{f}_2 v_i + r'_{2,i}} g_q^{c_x \tilde{f}_2 v_i}$. Where we denote $r_{2,i} = ac_x \tilde{f}_2 v_i + r'_{2,i}$ and $f_2 = c_x \tilde{f}_2$.

K_0 is generated by $K_0 = Q''R'' \prod_{i=1}^t h_{1,i}^{-r_{1,i}} h_{2,i}^{-r_{2,i}}$
$= QR \prod_{i=1}^t g_p^{-(bx_i+\omega_{1,i})(a\tilde{f}_1 v_i - abc_y \tilde{f}_2 v_i + r'_{1,i})} g_p^{-(b^2 y_i+\omega_{2,i})(ac_x \tilde{f}_2 v_i + r'_{2,i})}$
$= QR \prod_{i=1}^t g_p^{-(ab\tilde{f}_1 x_i v_i - ab^2 c_y \tilde{f}_2 x_i v_i + br'_{1,i} x_i + a\tilde{f}_1 \omega_{1,i} v_i - abc_y \tilde{f}_2 \omega_{1,i} v_i + r'_{1,i}\omega_{1,i} + ab^2 c_x \tilde{f}_2 y_i v_i + b^2 r'_{2,i} y_i + ac_x \tilde{f}_2 v_i \omega_{2,i} + r'_{2,i}\omega_{2,i})}$
$= QR \prod_{i=1}^t g_p^{-a(\tilde{f}_1 \omega_{1,i} v_i + c_x \tilde{f}_2 v_i \omega_{2,i})} g_p^{-ab(\tilde{f}_1 x_i v_i - c_y \tilde{f}_2 \omega_{1,i} v_i)} g_p^{-b(r'_{1,i} x_i)} g_p^{-b^2(r'_{2,i} y_i)} g_p^{-(r'_{1,i}\omega_{1,i} + r'_{2,i}\omega_{2,i})}$
$= QR \prod_{i=1}^t (g_p^a g_q)^{-(\tilde{f}_1 \omega_{1,i} v_i + c_x \tilde{f}_2 v_i \omega_{2,i})} (g_p^{ab} Q_1)^{-(\tilde{f}_1 x_i v_i - c_y \tilde{f}_2 \omega_{1,i} v_i)} h_p^{-(r'_{1,i} x_i)} k_p^{-(r'_{2,i} y_i)} g_p^{-(r'_{1,i}\omega_{1,i} + r'_{2,i}\omega_{2,i})}$

According to the symmetric property of scheme Π_2, $(X,Y) \stackrel{c}{\equiv} (0,Y)$ and $(0,Y) \stackrel{c}{\equiv} (Y,Y)$ can be proved similarly.

G Proof of Theorem 2

Proof. We first assume that the encryption scheme in Section 4.2 is not selective-ID secure in the MR model. That is there exists an PPT Adversary $\mathcal{A}_1$ with non-negligible advantage ϵ in the game of Definition 2. Now we construct a PPT adversary $\mathcal{A}_2$ which acts as Challenger interacting with $\mathcal{A}_1$ and show that it can win the game of Definition 1 also with non-negligible advantage with the scheme $\Pi_3 = (\mathsf{Setup}', \mathsf{Encrypt}', \mathsf{GenTK}', \mathsf{Test}')$.
Setup: The adversary $\mathcal{A}_2(1^n)$ runs adversary $\mathcal{A}_1(1^n)$. Adversary $\mathcal{A}_1$ outputs two possible equal-length vector X_0 and X_1 to adversary $\mathcal{A}_2$.

Adversary $\mathcal{A}_2$ passes X_0 and X_1 to the challenger $\mathcal{C}$.

The challenger $\mathcal{C}$ takes a security parameter n and runs `Setup` to generate pk and sk; C sends pk to adversary $\mathcal{A}_2$. $\mathcal{A}_2$ passes pk to $\mathcal{A}_1$.

Challenge: The challenger $\mathcal{C}$ picks a random bit $b \in \{0,1\}$, computes and returns $C^* \xleftarrow{\$} \texttt{Encrypt}(pk, X_b)$ to adversary $\mathcal{A}_2$. Adversary $\mathcal{A}_2$ passes it to adversary $\mathcal{A}_1$.
Phase 1: The adversary $\mathcal{A}_1$ adaptively requests polynomially bounded tokens for any (V_i, t_i) subject to the restriction that $t_i < HammingDist(V_i, X_j)$ for both $j = 0, 1$. When receiving the request (V_i, t_i), adversary $\mathcal{A}_2$ generates $t_i + 1$ token requests to the challenger $\mathcal{C}$ that $(V_i, 0), ..., (V_i, t_i)$ and receives token $TK_0, ..., TK_{t_i}$. Adversary $\mathcal{A}_2$ answers adversary $\mathcal{A}_1$ with $(TK_0, ..., TK_{t_i})$.
Guess: Adversary $\mathcal{A}_1$ returns with the output bit b'. The adversary $\mathcal{A}_2$ passes b' as its output to the challenger $\mathcal{C}$.

Since we have the restriction in MR definition that $t_i < HammingDist(V_i, X_j)$ for both $j = 0, 1$, $k \neq HammingDist(V_i, X_j)$ for each $k \in \{0, ..., t_i\}$. And therefore, $\mathcal{A}_2$'s token requests satisfy the requirement in MC definition that $t_i = HammingDist(V_i, X_j)$ for both $j = 0, 1$ or $t_i \neq HammingDist(V_i, X_j)$ for both $j = 0, 1$.

The advantage $\mathcal{A}_2$ in the above MC game $|\Pr[b' = b] - \frac{1}{2}| = \mathbf{Adv}^{\mathsf{MR}}_{\Pi_3, \mathcal{A}_1}(n) = \epsilon$ which is non-negligible in security parameter n. However, according to our security analysis in Section 3, the non-negligible advantage is impossible. This completes our security proof.

H Secure t_{max} update

To support t_{max} update, we need do some modifications on the scheme in Section 4.1. The new encryption scheme allowing t_{max} update consists of five PPT algorithms $\Pi_{\mathsf{H}} = (\texttt{Setup}, \texttt{Encrypt}, \texttt{UpdateCipher}, \texttt{GenTK}, \texttt{Test})$:

- $\texttt{Setup}(1^n)$: It first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$. Then, it randomly selects g_p from $\mathbb{G}_p$, g_q from $\mathbb{G}_q$ and g_r from $\mathbb{G}_r$. It also randomly selects $\{h_{1,l,i}, h_{2,l,i}\}$ from $\mathbb{G}_p$ where $l \in [1, m+1]$ and $i \in [1, \binom{m}{l}]$. (So, the total number of terms is 2^m.) Then, it randomly selects h_3, h_4 from $\mathbb{G}_p$. It also randomly selects $R, \{R_{1,l,i}, R_{2,l,i}\}_{l \in [1,m+1], i \in [1, \binom{m}{l})]}$, R_3, R_4 from $\mathbb{G}_r$. It outputs

$$pk = \left\{ \begin{array}{c} g_p, g_r, Q = g_q R, \\ [H_{1,l,i} = h_{1,l,i} R_{1,l,i}, H_{2,l,i} = h_{2,l,i} R_{2,l,i}]_{l \in [1,m+1], i \in [1, \binom{m}{l})]}, \\ H_3 = h_3 R_3, H_4 = h_4 R_4 \end{array} \right\}$$

  and

$$sk = \{p, q, r, g_q, [h_{1,l,i}, h_{2,l,i}]_{l \in [1,m+1], i \in [1, \binom{m}{l})]}, h_3, h_4\}$$

- $\texttt{Encrypt}(pk, X)$: Rather than outputting ciphertext C, it also outputs state $S = \{\alpha, \beta, s\}$ used.
- $\texttt{GenTK}(pk, sk, V)$: The same as the original algorithm.
- $\texttt{UpdateCipher}(pk, T', t_{max}, X, S)$: It randomly selects $\{R'_{1,l,i}, R'_{2,l,i}\}_{l \in [t_{max}+2, T'+1], i \in [1, \binom{m}{l})]}$ from $\mathbb{G}_r$. Then, it outputs δ:

$$\left\{ \begin{bmatrix} C_{1,l,i} = H^s_{1,l,i} Q^{\alpha x_{j_1} \dots x_{j_l}} R'_{1,l,i}, \\ C_{1,l,i} = H^s_{2,l,i} Q^{\beta x_{j_1} \dots x_{j_l}} R'_{2,l,i} \end{bmatrix}_{l \in \{t_{max}+2, T'+1\}, i \in [1, \binom{m}{l}], 1 \leq j_1 < \dots < j_l \leq m} \right\}$$

– $\texttt{Test}(pk, sk, TK, C)$ The same as the original algorithm.

Then, we also need to define a proper security definition concerning t_{max} updates.

Definition 5. (Selective-ID secure in match-concealing mode with t_{max} update capability) *The encryption scheme* $\Pi_{\mathsf{H}} = (\texttt{Setup}, \texttt{Encrypt}, \texttt{UpdateCipher}, \texttt{GenTK}, \texttt{Test})$ *is MC secure if for all probabilistic polynomial-time Turing machine (adversary)* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *in the following game is negligible.*

Setup: The adversary $\mathcal{A}(1^n)$ outputs two possible equal-length (m-length where $2^m = poly(n)$) vectors X_0 and X_1 to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ takes a security parameter n and runs `Setup` to generate pk and sk. Adversary $\mathcal{A}$ is given pk.

Challenge: The challenger $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$, computes $(C^*, S^*) \xleftarrow{\$} \texttt{Encrypt}(pk, X_b)$. $\mathcal{C}$ returns C^* to adversary $\mathcal{A}$.
Phase 1: The adversary $\mathcal{A}$ may adaptively request polynomially bounded number of queries. The types of queries allowed are described as below:

– `GenTK`: Adversary $\mathcal{A}$ can request $\mathcal{C}$ to compute and return tokens of any $(V_i = v_{i,1} \dots v_{i,m}, t_i)$ where $t_i \leq t_{max}$ and V_i subject to the restriction that $t_i < HammingDist(V_i, X_j)$ for both $j = 0$ and 1, or $t_i \geq HammingDist(V_i, X_j)$ for both $j = 0$ and 1.
– `UpdateCipher`: Adversary $\mathcal{A}$ outputs T' ($t_{max} < T' \leq m$) to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ computes and returns $\delta \xleftarrow{\$} UpdateCipher(pk, T', t_{max}, X_b, S^*)$ to adversary $\mathcal{A}$. The challenger also records T' as the new t_{max}.

Guess: Adversary $\mathcal{A}$ outputs a guess bit b'. The advantage of $\mathcal{A}$ is defined as $|\Pr[b' = b] - \frac{1}{2}|$.

Theorem 3. *The encryption scheme* $\Pi_{\mathsf{H}} = (\texttt{Setup}, \texttt{Encrypt}, \texttt{UpdateCipher}, \texttt{GenTK}, \texttt{Test})$ *defined here is Selective-ID secure in match-concealing mode with* t_{max} *update capability.*

Proof. The above theorem is proved in two steps. We first modify the inner-product encryption defined in Section 4.1 to a scheme Π_{I} supporting ciphertext updates. Similar to the method in Lemma 3, we prove in Lemma 5 that if this modified inner-product encryption scheme Π_{I} is Selective-ID secure in match-concealing mode with t_{max} update capability, then, the encryption scheme $\Pi_{\mathsf{H}} = (\texttt{Setup}, \texttt{Encrypt}, \texttt{UpdateCipher}, \texttt{GenTK}, \texttt{Test})$ defined here is also Selective-ID secure in match-concealing mode with t_{max} update capability.

The second step is that we prove this modified inner-product encryption scheme Π_{I} is Selective-ID secure in match-concealing mode with t_{max} update if inner-product encryption scheme Π_2 defined in Section 4.1 is MC secure (Definition 4 and Lemma 4). It is proved by Lemma 6.

To prove the first stage, we first describe the modified inner-product encryption scheme which also consists of five PPT algorithms
$\Pi_{\mathsf{I}} = (\texttt{Setup}, \texttt{Encrypt}, \texttt{UpdateCipher}, \texttt{GenTK}, \texttt{Test})$:

- $\texttt{Setup}(1^n)$. It first runs $\mathcal{G}(1^n)$ to obtain $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$. Then, it randomly selects g_p from $\mathbb{G}_p$, g_q from $\mathbb{G}_q$ and g_r from $\mathbb{G}_r$. It also randomly selects $\{h_{1,i}, h_{2,i}\}_{i\in[1,L_{max}]}$ from $\mathbb{G}_p$, and then randomly selects $R, \{R_{1,i}, R_{2,i}\}_{i\in[1,L_{max}]}$ from $\mathbb{G}_r$. It outputs L_{max} and a description of a deterministic function $g(d, l, X)$ such that $d + l \le L_{max}$ and the output length $|g(d, l, X)| = d$. (This pre-defined function restricts adversary cannot adaptively updates ciphertext.) It also outputs public key pk and secret key sk:

$$pk = \{g_p, g_r, Q = g_q R, [H_{1,i} = h_{1,i}R_{1,i}, H_{2,i} = h_{2,i}R_{2,i}]_{i\in[1,L_{max}]}\}$$

  and

$$sk = \{p, q, r, g_q, [h_{1,i}, h_{2,i}]_{i\in[1,L_{max}]}\}$$

- $\texttt{Encrypt}(pk, X = x_1...x_i...x_{l_{max}})$: Rather than outputting ciphertext C, it also outputs state $S = \{\alpha, \beta, s\}$ used. Where $l_{max} \le L_{max}$.
- $\texttt{UpdateCipher}(pk, X' = x'_1...x'_{L'} = g(L', l_{max}, X), S)$: It randomly selects $\{R'_{1,i}, R'_{2,i}\}_{i\in[1,L']}$ from $\mathbb{G}_r$. Then, it outputs

$$\delta = \{C_{1,i+l_{max}} = H^s_{1,i+l_{max}} Q^{\alpha x'_i} R'_{1,i}, C_{2,i+l_{max}} = H^s_{2,i+l_{max}} Q^{\beta x'_i} R'_{2,i}\}_{i\in[1,L']}$$

- $\texttt{GenTK}(pk, sk, V = v_1...v_i...v_t)$: The same as the original algorithm.
- $\texttt{Test}(pk, TK, C)$: The same as the original algorithm.

We define its security definition as follows.

Definition 6. (Selective-ID secure in match concealing mode with t_{max} update capability) *The encryption scheme* $\Pi_{\mathsf{I}} = (\texttt{Setup}, \texttt{Encrypt}, \texttt{UpdateCipher}, \texttt{GenTK}, \texttt{Test})$ *is MC secure if for all probabilistic polynomial-time Turing machine (adversary)* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *in the following game is negligible.*

Setup: The adversary $\mathcal{A}(1^n)$ outputs two possible equal-length (l_{max}-length) vectors X_0 and X_1. The challenger $\mathcal{C}$ takes a security parameter n and runs Setup to generate pk and sk. $\mathcal{C}$ sends pk to $\mathcal{A}$. Setup also outputs $L_{max} = poly(n)$ and function $g()$.

Challenge: The challenger picks a random bit $b \in \{0, 1\}$, computes $(C^*, S^*) \xleftarrow{\$} \texttt{Encrypt}(pk, X_b)$. C^* is given to adversary $\mathcal{A}$.
Phase 1: The adversary $\mathcal{A}$ may adaptively request polynomially bounded number of queries. The types of queries allowed are described as below:

- GenTK: Adversary $\mathcal{A}$ may adaptively request challenger $\mathcal{C}$ of tokens for any $V_i = v_{i,1}...v_{i,t_i}$ where $t_i \le l_{max}$ and V_i subject to the restriction that $\sum_{k=1}^{t_i} x_{j,k}v_{i,k} = 0$ for both $j = 0, 1$ or $\sum_{k=1}^{t_i} x_{j,k}v_{i,k} \ne 0$ for both $j = 0, 1$.

– `UpdateCipher`: Adversary $\mathcal{A}$ outputs two equal-length (L'-length) vectors X_0' and X_1' where $X_0' = g(L', l_{max}, X_0)$ and $X_1' = g(L', l_{max}, X_1)$. $l_{max} + L' \leq L_{max}$ where L_{max} is decided at `Setup` time. Adversary $\mathcal{A}$ sends X_0' and X_1' to challenger $\mathcal{C}$. The challenger $\mathcal{C}$ computes and returns $\delta \xleftarrow{\$} UpdateCipher(pk, X_b', S^*)$ to adversary $\mathcal{A}$. $\mathcal{C}$ also record $l_{max} + L'$ as the new l_{max}.

Guess: The adversary $\mathcal{A}$ outputs a bit b' to guess b. The advantage of $\mathcal{A}$ is defined as $|\Pr[b' = b] - \frac{1}{2}|$.

Lemma 5. *The encryption scheme Π_H is secure under Definition 5 if inner-product encryption scheme Π_I is secure under Definition 6.*

Proof. We first assume that there exists an adversary $\mathcal{A}_1$ who wins Definition 5 with non-negligible advantage. Then, we try to construct an adversary $\mathcal{A}_2$ who can win Definition 6 also with non-negligible.
Setup: Adversary $\mathcal{A}_2(1^n)$ runs $\mathcal{A}_1(1^n)$. $\mathcal{A}_1(1^n)$ outputs two possible equal-length (m-length) vector X_0 and X_1 to $\mathcal{A}_2$.

A_2 calculates $\sum_{i=0}^{t_{max}+1} \binom{m}{i}$-length vectors $\tilde{X_0}$ and $\tilde{X_1}$ to challenger $\mathcal{C}$. The Challenger $\mathcal{C}$ takes a security parameter n and runs `Setup` to generate pk and sk. Set $L_{max} = 2^m$. $\mathcal{C}$ sends pk to adversary $\mathcal{A}_2$. $\mathcal{A}_2$ rearranges pk and sends it to $\mathcal{A}_1$.
Challenge: The challenger $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$ and computes $(C^*, S^*) \xleftarrow{\$}$ `Encrypt`(pk, X_b). $\mathcal{C}$ sends C^* to adversary $\mathcal{A}_2$. $\mathcal{A}_2$ rearranges C^* and sends to it $\mathcal{A}_1$.
Phase 1: Adversary $\mathcal{A}_1$ may adaptively request polynomially bounded `GenTK` and `UpdateCipher` queries.

– `GenTK`: The adversary $\mathcal{A}_1$ requests tokens to $\mathcal{A}_2$ for any $(V_i = v_{i,1}...v_{i,m}, t_i)$ where $t_i \leq t_{max}$ and V_i subject to the restriction that $t_i < HammingDist(V_i, X_j)$ for both $j = 0$ and 1 or $HammingDist(V_i, X_j) \leq t_i$ for both $j = 0$ and 1.
When receiving (V, t), adversary A_2 calculates $\tilde{V} = \tilde{v}_1, ..., \tilde{v}_{\sum_{i=0}^{t+1} \binom{m}{i}}$ such that $\sum \tilde{v}_i \tilde{x}_{j,i} = 1$ if and only if $HammingDist(V, X_j) \leq t$. $\mathcal{A}_2$ submits $\tilde{V}$ to challenger $\mathcal{C}$ and gets token.
$\mathcal{A}_2$ rearranges the token and sends it to $\mathcal{A}_1$.
– `UpdateCipher`: The adversary $\mathcal{A}_1$ outputs $m \geq T' > t_{max}$ to $\mathcal{A}_2$. $\mathcal{A}_2$ based on X_0 and X_1 and T' to calculate two equal-length ($L' = \sum_{i=t_{max}+2}^{T'+1} \binom{m}{i}$)-length) vectors X_0' and X_1' (according to $\boldsymbol{a}$ in (5) or equivalently, $g(L', l_{max}, X_0)$ and $g(L', l_{max}, X_1)$). Adversary $\mathcal{A}_2$ passes X_0' and X_1' to challenger $\mathcal{C}$.
The challenger $\mathcal{C}$ returns $\delta \xleftarrow{\$} UpdateCipher(pk, X_b', S^*)$ to adversary $\mathcal{A}_2$. Challenger $\mathcal{C}$ also updates l_{max} as $L' + l_{max}$. $\mathcal{A}_2$ rearranges δ and sends to $\mathcal{A}_1$.

Guess: The adversary $\mathcal{A}_1$ outputs a bit b' to guess b. $\mathcal{A}_2$ passes b' to challenger $\mathcal{C}$ as its output.

Then, we prove Lemma 6 to complete the proof for Theorem 3.

Lemma 6. *The scheme* $\Pi_{\mathsf{I}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{UpdateCipher}, \mathsf{GenTK}, \mathsf{Test})$ *is secure under Definition 6 if* $\Pi_2 = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{GenTK}, \mathsf{Test})$ *is secure under Definition 4.*

Proof. This proof is by contradiction. We first assume that there exists an adversary $\mathcal{A}_1$ who wins Selective-ID game in match-concealing mode with t_{max} update capability with non-negligible advantage ϵ, then we can construct an adversary $\mathcal{A}_2$ who wins Definition 4 also with non-negligible advantage ϵ.
Setup: The adversary $\mathcal{A}_2(1^n)$ runs $\mathcal{A}_1(1^n)$. $\mathcal{A}_1(1^n)$ outputs two possible equal-length (l_{max}-length) vectors X_0 and X_1 to $\mathcal{A}_2$. Note that $l_{max} \leq L_{max}$.

The adversary $\mathcal{A}_2$ calculates and outputs two L_{max}-length vectors $X_0^{(max)}$ and $X_1^{(max)}$ to challenger $\mathcal{C}$ where $X_0^{(max)} = g(L_{max}, 0, X_0)$ and $X_1^{(max)} = g(L_{max}, 0, X_1)$.

The challenger $\mathcal{C}$ takes a security parameter n and runs `Setup` to generate pk and sk (for L_{max}-length). $\mathcal{C}$ returns pk to adversary $\mathcal{A}_2(1^n)$. The adversary $\mathcal{A}_2$ passes pk to $\mathcal{A}_1$.
Challenge: The challenger $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$, computes and returns $C^* \xleftarrow{\$} \mathtt{Encrypt}(pk, X_b^{(max)})$ to adversary $\mathcal{A}_2$.

The adversary $\mathcal{A}_2$ takes the first l_{max} components of ciphertext C^* and returns it (denoted as $C^*_{l_{max}}$) to $\mathcal{A}_1$.
Phase 1: The adversary $\mathcal{A}_1$ may adaptively request polynomially-bounded number of `GenTK` and `UpdateCipher` queries.

- `GenTK`: The adversary $\mathcal{A}_1$ may adaptively request tokens for any $V_i = v_{i,1}...v_{i,t_i}$ where $t_i \leq l_{max}$ and V_i subject to the restriction that $\sum_{k=1}^{t_i} x_{j,k}v_{i,k} = 0$ for both $j = 0$ and 1 or $\sum_{k=1}^{t_i} x_{j,k}v_{i,k} \neq 0$ for both $j = 0$ and 1. $\mathcal{A}_1$ sends these requests to $\mathcal{A}_2$.
  The adversary $\mathcal{A}_2$ passes these requests to the challenger $\mathcal{C}$. Note that $t_i \leq l_{max} \leq L_{max}$. The challenger generates tokens and adversary $\mathcal{A}_2$ passes them to $\mathcal{A}_1$.
- `UpdateCipher`: The adversary $\mathcal{A}_1$ outputs two equal-length (L'-length) vectors X_0' and X_1' where $X_0' = g(L', l_{max}, X_0)$ and $X_1' = g(L', l_{max}, X_1)$. The adversary $\mathcal{A}_1$ passes X_0' and X_1' to $\mathcal{A}_2$.
  The adversary $\mathcal{A}_2$ passes $\delta = C^*[l_{max} + 1, l_{max} + L']$ to $\mathcal{A}_1$. The adversary $\mathcal{A}_2$ also records $l_{max} + L'$ as the new l_{max}.

Guess: The adversary $\mathcal{A}_1$ outputs a bit b' to $\mathcal{A}_2$. And $\mathcal{A}_2$ passes b' to the challenger $\mathcal{C}$ as its output. Note that $|\Pr[b' = b] - \frac{1}{2}| = \epsilon$. This completes our proof.