Proximity Searchable Encryption for the Iris Biometric

Sohaib Ahmad* Chloe Cachet† Luke Demarest‡ Benjamin Fuller§ Ariel Hamlin¶

March 1, 2021

Abstract

Biometric databases collect people's information and allow users to perform proximity searches (finding all records within a bounded distance of the query point) with few cryptographic protections. This work studies proximity searchable encryption in the context of biometric databases. Prior work on proximity searchable encryption leaks the (approximate) distance between the query and all stored records (such as Kim et al., SCN 2018). This leakage is especially dangerous; if the server knows a few queried points, this local geometry allows the server to (approximately) recover stored records. Additionally, searchable encryption attacks on nearest neighbor schemes also apply (Kornaropoulos et al., IEEE S&P 2019). Since biometrics cannot be replaced, compromises follow people their entire life.

In this work we present three main contributions:

1. A searchable encryption construction that supports proximity queries for the Hamming metric (a commonly used metric for iris biometrics) built from function-hiding, secret-key, predicate, inner product encryption (Shen, Shi, and Waters, TCC 2009). Our construction only leaks access patterns, and which returned records are the same distance from the query.
2. For the iris, we also show it is possible to set parameters of the system to support high true accept rates and a low probability of multiple biometrics being returned by a query (an important property that minimizes leakage). As part of this analysis we also analyze multiple dimensionality reduction techniques, which improves search speed by the square of the reduction.
3. A technique that reduces the client storage from a quadratic to linear number of group elements in the biometric dimension. This technique applies to multiple inner product encryption schemes and may be of independent interest.

Keywords: Searchable encryption, biometrics, proximity search, inner product encryption.

1 Introduction

Biometrics are both stored in databases for *identification* and used to *authenticate* users' access to their private accounts and devices. This dual use creates security and privacy risks. Learning stored biometric *templates* (the digital object storing the information of a physical biometric) enables an attacker to reverse this value into a convincing biometric [GRGB+12, MCYJ18, AF20], enabling *presentation* attacks [HWKL18, SDDN19, VS11] that can compromise users' accounts and devices. Since biometrics cannot be updated, such a compromise lasts a lifetime.

This work focuses on reducing the risk of server compromise by limiting the server's access to private data using searchable encryption [SWP00, CGKO11, BHJP14, FVY+17]. The goal is for a client to provide an encrypted index to a server along with encrypted data. Later, when the client wishes to query, they produce an encrypted version of their query which the server can use to find the relevant records. The security goal is to keep the (semi-honest) server from being able to infer anything about the stored data and queries. In the above, we assume the server sees which records are returned which is called the *access pattern* [IKK12, CGPR15].

A recent wave of attacks [IKK12, CGPR15, KKNO16, WLD+17, GSB+17, GLMP18] has cast doubt on whether it is possible to design solutions that work across a variety of applications. These attacks result from *leakage*, information

*University of Connecticut. Email sohaib.ahmad@uconn.edu
†University of Connecticut. Email chloe.cachet@uconn.edu
‡University of Connecticut. Email luke.h.demarest@gmail.com
§University of Connecticut. Email benjamin.fuller@uconn.edu
¶Khoury College of Computer Sciences, Northeastern University. Email ahamlin@ccs.neu.edu

a server learns during database operations, such as access patterns or the number of records returned. In some cases, these attacks only require the server to see the access pattern and preventing these *leakage abuse attacks* requires resorting to oblivious RAM [GKL+20] and its high storage and communication overhead. The most efficient of these attacks apply against systems that support range queries. The attacker is able to exploit the underlying geometry of these spaces [KPT19, MT19, KE19, KPT20, FMC+20]. This prompts us to to carefully examine different applications and design leakage for schemes dependent on the underlying data distributions. It is this ethos our construction of searchable encryption for biometric databases embraces.

A crucial characteristic of biometrics is *noise*. When one takes repeated readings of the same physical object one sees similar (not identical) values. This is due to a variety of sources including sensor noise, change of the phenomena (such as eye dilation), and environmental conditions. Noise is modeled by assuming there is some distance metric $\mathcal{D}$; two values x, x' are treated as resulting from the same biometric if $\mathcal{D}(x, x') \leq t$ for some threshold t. For a biometric to be useful for identification or authentication it must be the case that readings from some other biometric y are rarely close to x. Biometrics that are used today including the irises, facial structure, and fingerprints have this property.

When building searchable encryption for biometrics the standard type of query is to return all records that are within distance t of a query point y. We call this *proximity search* and focus on building proximity searchable encryption.[1]

Prior approaches in proximity searchable encryption have one of two major limitations for biometric applications.[2] Either they support either a small distance threshold [LWW+10, WMT+13, BC14] (all close values are searched for as part of the query or stored as part of the record) **or** for any searched value y they allow the server to compute the (approximate) distance [KIK12, KLM+18] between y and *all* stored records.[3] The first approach is inefficient for biometrics where the number of close values is exponential in t. The second approach is equally problematic, establishing a rich geometry on the space of records and paving the way for query and data recovery. We review related approaches in detail in Section 6.

To exploit this information, the adversary needs to reconstruct global geometry from the local geometry revealed by pairwise distances [PBDT05]. This problem has been studied in wireless sensor networks [AEG+06]. In two (or three) dimensional Euclidean space, trilateration has been practiced for hundreds of years: one is assumed to know the location of $y_1, ..., y_k$ and the pairwise distances $\mathcal{D}(y_i, x)$ and is trying to find the location of x. Determining the location of x requires k to be one larger than the dimension. The problem is more difficult but well studied for approximate distances [EA11]. Similar ideas can be applied in discrete metrics with each learned distance reducing the set of possible x. In the Hamming metric of dimension n, $k = \Theta(n)$ suffices [TFL19, LTBL20, Lai20]. While we are not aware of any leakage abuse attacks directly against proximity search, there are attacks against k-nearest neighbor databases [KPT19, KE19]. Approximate distance allows one to easily compute the k-nearest points (with some error) so attacks that can expliot this leakage apply.

While existing schemes directly reveal approximate distance, some structure is implied by just the access pattern, if x_i and x_j are both returned by a query it must be the case that $\mathcal{D}(x_i, x_j) \leq 2t$. Together, the rich geometry implied by proximity search and the study of trilateration seems to make the goal of secure biometric databases hopeless. However, while biometric databases require new functionality in the form of proximity queries biometrics have unique statistical properties. Biometrics are well spread, that is one does not expect readings of two biometrics to be close. Returning at most one biometric is the desired search behavior. Our hypothesis is that such an application setting can curb leakage abuse attacks. The two questions of this work are:

1. Can one design proximity searchable encryption that does not leak approximate distance?
2. Can one use biometric statistical properties to reduce the impact of leakage?

1.1 Our Contribution

This work presents three contributions:

[1] *Only* points within the distance t are returned, *not* the closest point or k-closest points as in nearest neighbor systems [RKV95].

[2] One exception is the work of Zhou and Ren [ZR18] which proposes a mechanism that reveals if the distance is less than t only. However, security is heuristic with no underlying assumption or proof of information theoretic security.

[3] The schemes that allow computation of approximate distance rely on locality sensitive hashes [IM98], allowing the server to see how many hashes match, the number of matches approximates distance.

1. A simple construction of proximity searchable encryption for the Hamming metric based on inner product encryption or IPE [KSW08]. The construction reveals access pattern and whether two returned records are the same distance from the query. That is, for query y, for *returned* values x_i, x_j, the server learns whether $\mathcal{D}(x_i, y) \stackrel{?}{=} \mathcal{D}(x_j, y)$. We call this *distance equality leakage*.

   For biometric dimension n, both the search time and the required client storage are quadratic in n for the most efficient current IPE scheme [BCSW19] (see Table 1). For the iris, where $n = 1024$ this means a search complexity of roughly 2^{18} group operations per stored item. The corresponding client storage is 2^{21} field elements. Our other contributions reduce these costs.

2. Statistical analysis for the iris biometric showing one can set parameters so that the system has a high true accept rate (TAR) while ensuring that for most queries at most one biometric is returned. We additionally analyze dimensionality reduction techniques and show an iris transform with $n = 64$ that only marginally affects the TAR/leakage tradeoff.

3. A inner product encryption technique (and resulting) that reduces the client for reduces the required client storage from quadratic in n to linear in n.

After applying both of these technique one obtains search complexity of approximately 2^{10} group operations per stored item and 2^9 field elements of client storage.

Construction Our construction can be built from any function hiding, secret key, predicate, inner product encryption (which we denote as $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$) such as the recent scheme of Barbosa et al. [BCSW19]. The functionality of such a scheme is that one can produce ciphertexts $c_{\vec{x}}$ and tokens $\mathsf{tk}_{\vec{y}}$ which allow one to effectively check if $\langle \vec{x}, \vec{y} \rangle = 0$ without revealing anything else about $\vec{x}$ or $\vec{y}$. Binary Hamming distance is connected to inner product: if records are encoded as vectors in $\{-1, 1\}$ then

$$\mathcal{D}(x, y) = (n - \langle \vec{x}, \vec{y} \rangle)/2$$

for $|\vec{y}| = n$. One can naturally use $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ to test if the distance between $\vec{x}$ and $\vec{y}$ is equal to i: add an $n + 1^{th}$ element as -1 to $\vec{x}$, denoted $\vec{x}'$, and an $n + 1^{th}$ element to $\vec{y}$, denoted $\vec{y}_i$, that is $n - 2i$. This means that,

$$\langle \vec{x}', \vec{y}_i \rangle = \langle \vec{x} \mid\mid \text{-1}, \vec{y}_i \mid\mid n - 2i \rangle = 0 \Leftrightarrow \mathcal{D}(\vec{x}, \vec{y}) = i.$$

This construction can naturally be extended to check for distance at most t by creating $t + 1$ values y_i for $0 \leq i \leq t$ (these tokens are permuted before being sent to server). Since the server can see if the same tk matches different records, when two records are both within distance t, the server learns if they match the same distance (but not the specific distance). The simplicity and generality of this construction is an advantage, it immediately benefits from efficiency improvements inner product encryption and can be built from multiple different computational assumptions.[4] Indeed, the third contribution of this work is an improvement on the efficiency of inner product encryption.

The scheme requires an $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ ciphertext of dimension $n+1$ and $(t+1)$-$\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ tokens of $(n+1)$-dimension for each query. When instantiated using the scheme of Barbosa et al. [BCSW19] which uses bilinear pairing groups, for a biometric of dimension n, ciphertexts are of size $2n + 2$ group elements and tokens are of size $(t + 1)(2n + 2)$ group elements. In most biometric settings $t = \Theta(n)$. Importantly, Search requires $O(tn)$ group operations per stored record (see Table 1).

Application to the Iris Before introducing our second contribution, we need to distinguish between two types of biometric databases: 1) the database stores a single reading of each biometric and 2) multiple readings may be associated with each biometric (added, for example, when a traveler enters a country). For the Introduction, we consider the simpler case where only one reading is stored but consider the more complex case of multiple readings in Section 4. In both cases the goal is to return a single biometric.

Our statistical analysis for the iris biometric shows how to set parameters so that the system has both a high TAR and low leakage. When at most one biometric is returned, one cannot use access pattern and distance equality leakage to compute any global geometry between biometrics. This analysis crucially uses the fact in biometric databases one

[4] Inner product encryption has been proposed in the past for proximity search (e.g. [KLM+18]). Using a predicate version is critical to prevent the server from learning the distance between the query and all stored records.

sets the distance parameter t so that one rarely expects for a different biometric x that $\mathcal{D}(y, x) \leq t$. The impact of leakage about multiple readings of the same biometric is an important question we consider in Section 4.2.

To improve efficiency of our cryptographic scheme we consider dimensionality reduction techniques on the biometric values (see Section 4.3). For the iris, using state of the art feature extractors [AF19] we show it is possible to reduce the dimension n from 1024 to 64 with a limited impact on TAR and the probability that two biometrics will be returned by a query. This translates to a 16-fold reduction in the database storage size and 256-fold reduction in the size of search tokens and efficiency of the search procedure.

Client Storage Reduction A technical contribution which reduces the required client storage from a quadratic number of group elements to a linear number of group elements. At a technical level, this corresponds to reducing the secret key size in the $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$. In Barbosa et al. [BCSW19] client storage is $2n^2$ group elements which for the iris where 1024 means client storage of 2^{21} group elements (if one reduces to $n = 64$ this corresponds to 2^{13} group elements). In the reduced scheme storage is 2^{13} group elements (2^9 when $n = 64$). We note that this transform does increase the size of ciphertexts and tokens (by at most a factor of 2), so it should only be applied if the client storage is a bottleneck.

Roughly, our technique works on schemes that use a random matrix $\mathbf{A}$ (and $\mathbf{A}^{-1}$) as the secret key (such as schemes based on dual pairing vector spaces), reducing the dimension of the matrix but using multiple pairs of matrices. Importantly, one must ensure that no partial information is leaked which is accomplished by adding a secret sharing component to each vector. We apply our technique to scheme of Barbosa et al. [BCSW19]. To show the generality of this technique, in Appendix A we also apply it to another inner product encryption scheme that is secure in the standard model [OT12, Section 4]. However, this scheme is not a predicate scheme so it cannot be used for proximity search.

Organization The rest of this work is organized as follows, Section 2 describes mathematical and cryptographic preliminaries, Section 3 shows that $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ suffices to build proximity search, Section 4 describes prior leakage attacks and how to configure a biometric database so these attacks are mitigated, Section 5 describes our transform for reducing key size. Finally, Section 7 focuses on the current efficiency level and applicability to different biometric databases.

2 Preliminaries

Let λ be the security parameter throughout the paper. We use $\mathtt{poly}(\lambda)$ and $\mathsf{negl}(\lambda)$ to denote unspecified functions that are polynomial and negligible in λ, respectively. For some $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, \cdots, n\}$. Let $x \xleftarrow{\$} S$ denote sampling x uniformly at random from the finite set S. Let $q = q(\lambda) \in \mathbb{N}$ be a prime, then $\mathbb{G}_q$ denotes a cyclic group of order q. Let $\vec{x}$ denote a vector over $\mathbb{Z}_q$ such that $\vec{x} = (x_1, \cdots, x_n) \in \mathbb{Z}_q^n$, the dimension of vectors should be apparent from context. Consider vectors $\vec{x} = (x_1, \cdots, x_n)$ and $\vec{v} = (v_1, \cdots, v_n)$, their inner-product is denoted by $\langle x, v \rangle = \sum_{i=1}^{n} x_i v_i$. Let X be a matrix, then X^T denotes its transpose.

Hamming distance is defined as the distance between the bit vectors x and y of length n: $d(x, y) = |\{i \,|\, x_i \neq y_i\}|$. We note that if a vector over $\{0, 1\}$ as is encoded as $x_{\pm 1,i} = 1$ if and $x_i = 1$ and $x_{\pm 1,i} = -1$ if $x_i = 0$ then it is true that $\langle x_{\pm 1}, y_{\pm 1} \rangle = n - 2d(x, y)$.

2.1 Inner Product Encryption

Secret-key predicate encryption with function privacy supporting inner products queries was first proposed by Shen et al. [SSW09]. This primitive allows one to check if the inner product between vectors is zero or not. The scheme they presented is both attribute and function hiding, meaning that an adversary running the decryption algorithm gains no knowledge on either the attribute or the predicate.

Definition 1 (Secret key predicate encryption)**.** *Let $\lambda \in \mathbb{N}$ be the security parameter, $\mathcal{M}$ be the set of attributes and $\mathcal{F}$ be a set of predicates. We define $\mathsf{PE} = (\mathsf{PE.Setup}, \mathsf{PE.Encrypt}, \mathsf{PE.TokGen}, \mathsf{PE.Decrypt})$, a secret-key predicate encryption scheme, as follows:*

- $\mathsf{PE.Setup}(1^\lambda) \to (\mathsf{sk}, \mathsf{pp})$,

- $\mathsf{PE.Encrypt}(\mathsf{sk}, x) \to \mathsf{ct}_x$,
- $\mathsf{PE.TokGen}(\mathsf{sk}, f) \to \mathsf{tk}_f$,
- $\mathsf{PE.Decrypt}(\mathsf{pp}, \mathsf{tk}_f, \mathsf{ct}_x) \to b$.

We require the scheme to have the following properties:

Correctness: *For any* $x \in \mathcal{M}, f \in \mathcal{F}$,

$$\Pr\left[f(x) = b \;\middle|\; \begin{array}{c} \mathsf{ct}_x \leftarrow \mathsf{PE.Encrypt}(\mathsf{sk}, x) \\ \mathsf{tk}_f \leftarrow \mathsf{PE.TokGen}(\mathsf{sk}, f) \\ b \leftarrow \mathsf{PE.Decrypt}(\mathsf{pp}, \mathsf{tk}_f, \mathsf{ct}_x) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Security of admissible queries: *Let* $r = \mathtt{poly}(\lambda)$ *and* $s = \mathtt{poly}(\lambda)$. *Any PPT adversary* $\mathcal{A}$ *has only* $\mathsf{negl}(\lambda)$ *advantage in the* $\mathsf{Exp}^{\mathsf{PE}}_{IND}$ *game (defined in Figure 1). Token and encryption queries must meet the following admissibility requirements,* $\forall j \in [1, r], \forall i \in [1, s]$,

$$\mathsf{PE.Decrypt}(\mathsf{pp}, \mathsf{tk}_j^{(0)}, \mathsf{ct}_i^{(0)}) = \mathsf{PE.Decrypt}(\mathsf{pp}, \mathsf{tk}_j^{(1)}, \mathsf{ct}_i^{(1)}).$$

The above definition is called *full security* in the language of Shen, Shi, and Waters [SSW09]. Note that this definition is selective (not adaptive), as the adversary specifies two sets of plaintexts and functions apriori. The relevant primitive for us is $\mathsf{IPE}_{\mathsf{fh},\mathsf{sk},\mathsf{pred}}$ which uses the above definition restricted to the class of predicates $\mathcal{F} = \{f_y \mid y \in \mathbb{Z}_q^n\}$ be the set of predicates such that for all vectors $x \in \mathbb{Z}_q^n$, $f_y(x) = 1$ when $\langle x, y\rangle = 0$, $f_{y,t}(x) = 0$ otherwise. We use $(\mathsf{IPE.Setup}, \mathsf{IPE.Encrypt}, \mathsf{IPE.TokGen}, \mathsf{IPE.Decrypt})$ to refer to the corresponding tuple of algorithms.

Our construction in Section 5 uses asymmetric bilinear groups. We include a definition below.

Definition 2 (Asymmetric Bilinear Group)**.** *Suppose* $\mathbb{G}_1, \mathbb{G}_2$, *and* $\mathbb{G}_T$ *are three groups (respectively) of prime order* q *with generators* $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ *and* $g_T \in \mathbb{G}_T$ *respectively. We denote a value* x *encoded in* $\mathbb{G}_1$ *with either* g_1^x *or* $[x]_1$, *we denote* $\mathbb{G}_2$ *similarly. Let* $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ *be a non-degenerate (i.e.* $e(g_1, g_2) \neq 1$*) bilinear pairing operation such that for all* $x, y \in \mathbb{Z}_q$, $e([x]_1, [y]_2) = e(g_1, g_2)^{xy}$. *We assume the group operations in* $\mathbb{G}_1, \mathbb{G}_2$ *and* $\mathbb{G}_T$ *and the pairing operation* e *are efficiently computable, then* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e)$ *defines an asymmetric bilinear group.*

Our scheme requires that the particular generators are kept private so we omit them from the description of the group. Let $\mathcal{G}_{abg}$ be an algorithm that takes input 1^λ and outputs a description of an asymmetric bilinear groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e)$ with security parameter λ.

2.2 Proximity searchable encryption

In this section we define *proximity searchable encryption (PSE)*, a variant of searchable encryption that supports proximity queries.

Definition 3 (History)**.** *Let* $X \in \mathcal{M}$ *be a list of keywords drawn from space* $\mathcal{M}$, *let* $\mathcal{F}$ *be the set of all predicates over* $\mathcal{M}$, *an* m*-query history over* $\mathcal{W}$ *is a tuple* $\mathsf{History} = (X, F)$, *with* $F = (f_1, \cdots, f_m)$ *a list of* m *predicates,* $f_i \in \mathcal{F}$.

Definition 4 (Access pattern)**.** *Let* X *be a list of keywords, the access pattern induced by an* m*-query history* $\mathsf{History} = (X, F)$ *is the tuple* $\mathsf{AccPatt}(\mathsf{History}) = (f_1(X), \cdots, f_m(X))$

Definition 5 (Distance Equality)**.** *Let* $\mathsf{History}^{(0)}$ *and* $\mathsf{History}^{(1)}$ *be two* m*-query histories for predicates of the type* $f_{y,t}(x) = (\mathcal{D}(x, y) \overset{?}{\leq} t)$. *Let,* $\mathsf{DisEq}(\mathsf{History}^{(0)}, \mathsf{History}^{(1)}) = 1$ *if and only if for each* j *it is true that*

$$\left\{(i,k) \middle| \begin{array}{c} (\mathcal{D}(x_i^{(0)}, y_j^{(0)}) = \mathcal{D}(x_k^{(0)}, y_j^{(0)}) \wedge \mathcal{D}(x_i^{(1)}, y_j^{(1)}) \neq \mathcal{D}(x_k^{(1)}, y_j^{(1)})) \\ \vee \\ (\mathcal{D}(x_i^{(0)}, y_j^{(0)}) \neq \mathcal{D}(x_k^{(0)}, y_j^{(0)}) \wedge \mathcal{D}(x_i^{(1)}, y_j^{(1)}) = \mathcal{D}(x_k^{(1)}, y_j^{(1)})) \end{array}\right\},$$

is the empty set.

Many searchable encryption schemes leak when queries are repeated (*query equality leakage*) but this is not the case with our construction.

1. Draws $\beta \xleftarrow{\$} \{0,1\}$,
2. Computes $(\mathsf{sk}, \mathsf{pp}) \leftarrow \mathsf{PE.Setup}(1^\lambda)$, sends pp to $\mathcal{A}$,
3. For $1 \le i \le s$, $\mathcal{A}$ chooses $x_i^{(0)}, x_i^{(1)} \in \mathcal{M}$,
4. For $1 \le j \le r$, $\mathcal{A}$ chooses $f_j^{(0)}, f_j^{(1)} \in \mathcal{F}$,
5. Denote $R := \left(x_1^{(0)}, x_1^{(1)}\right), \cdots, \left(x_s^{(0)}, x_s^{(1)}\right)$,$S := \left(f_1^{(0)}, f_1^{(1)}\right), \cdots, \left(f_r^{(0)}, f_r^{(1)}\right)$.
6. $\mathcal{A}$ sends R and S to $\mathcal{C}$,
7. $\mathcal{A}$ loses the game if R and S are not admissible,
8. $\mathcal{A}$ receives $C^{(\beta)} := \{ct_i^{(\beta)} \leftarrow \mathsf{PE.Encrypt}\left(\mathsf{sk}, x_i^{(\beta)}\right)\}_{i=1}^r, T^{(\beta)} := \{tk_j^{(\beta)} \leftarrow \mathsf{PE.TokGen}\left(\mathsf{sk}, f_j^{(\beta)}\right)\}_{j=1}^s$
9. $\mathcal{A}$ returns $\beta' \in \{0,1\}$,
10. Her *advantage* is $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Exp}_{IND}^{\mathsf{PE}}}(\lambda) = \left| \Pr[\mathcal{A}(1^\lambda, T^{(0)}, C^{(0)}) = 1] - \Pr[\mathcal{A}(1^\lambda, T^{(1)}, C^{(1)}) = 1] \right|$

Figure 1: Definition of $\mathsf{Exp}_{IND}^{\mathsf{PE}}$ for predicate encryption.

Definition 6 (Proximity Searchable Encryption)**.** *Let*

- $\lambda \in \mathbb{N}$ *be the security parameter,*
- $\mathcal{DB} = (M_1, \cdots, M_\ell)$ *be a database of size* ℓ*,*
- *Keywords* $X = (\vec{x}_1, \cdots, \vec{x}_\ell)$*, such that* $\vec{x}_i \in \mathbb{Z}_q^n$ *is the keyword related to* M_i*,*
- $\mathcal{F} = \{f_{\vec{y},t} \mid \vec{y} \in \mathbb{Z}_q^n,\ t \in N\}$ *be a family of predicates such that, for a keyword* $\vec{x} \in \mathbb{Z}_q^n$*,* $f_{\vec{y},t}(\vec{x}) = 1$ *if* $\mathcal{D}(\vec{x}, \vec{y}) \le t$*,* 0 *otherwise, where* $\mathcal{D}(\vec{x}, \vec{y})$ *denotes the Hamming distance between* $\vec{x}$ *and* $\vec{y}$*.*

The tuple of algorithms $\mathsf{PSE} = (\mathsf{PSE.Setup}, \mathsf{PSE.BuildIndex}, \mathsf{PSE.Trapdoor}, \mathsf{PSE.Search})$ *defines a proximity searchable encryption scheme:*

- $\mathsf{PSE.Setup}(1^\lambda) \to (\mathsf{sk}, \mathsf{pp})$,
- $\mathsf{PSE.BuildIndex}(\mathsf{sk}, X) \to I_X$,
- $\mathsf{PSE.Trapdoor}(\mathsf{sk}, f_{\vec{y},t}) \to \mathsf{tk}_{\vec{y},t}$,
- $\mathsf{PSE.Search}(\mathsf{pp}, Q_{\vec{y},t}, I_X) \to J_{X,\vec{y},t}$.

We require the scheme to have the following properties:

Correctness *Define* $J_{X,\vec{y},t} = \{i | f_{\vec{y},t}(\vec{x}_i) = 1, \vec{x}_i \in X\}$*.* PSE *is correct if for all* X *and* $f_{\vec{y},t} \in \mathcal{F}$*:*

$$\Pr\left[J' = J_{X,\vec{y},t} \middle| \begin{matrix} I_X \leftarrow \mathsf{PSE.BuildIndex}(\mathsf{sk}, X) \\ Q_{\vec{y},t} \leftarrow \mathsf{PSE.Trapdoor}(\mathsf{sk}, f_{\vec{y},t}) \\ J' \leftarrow \mathsf{PSE.Search}(\mathsf{pp}, Q_{\vec{y},t}, I_X) \end{matrix}\right] \ge 1 - \mathsf{negl}(\lambda).$$

Security for Admissible Queries *Any PPT adversary* $\mathcal{A}$ *has only* $\mathsf{negl}(\lambda)$ *advantage in the experiment* $\mathsf{Exp}_{IND}^{\mathsf{PSE}}$ *defined in Figure 2, for* $\ell = \mathtt{poly}(\lambda)$ *and* $m = \mathtt{poly}(\lambda)$*.*

1. Draws $\beta \xleftarrow{\$} \{0,1\}$,
2. Computes $(\mathsf{sk}, \mathsf{pp}) \leftarrow \mathsf{PSE.Setup}(1^\lambda)$ and sends pp to $\mathcal{A}$.
3. $\mathcal{A}$ chooses and outputs $\mathsf{History}^{(0)}, \mathsf{History}^{(1)}$.
4. $\mathcal{A}$ loses the game if $\mathsf{AccPatt}(\mathsf{History}^{(0)}) \neq \mathsf{AccPatt}(\mathsf{History}^{(1)}) \vee \ \mathsf{DisEq}(\mathsf{History}^{(0)}, \mathsf{History}^{(1)}) = 0$
5. $\mathcal{A}$ receives $I^{(\beta)}$ and $Q^{(\beta)}$.
6. $\mathcal{A}$ outputs $\beta' \in \{0,1\}$
7. Her advantage in the game is: $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Exp}_{IND}^{\mathsf{PSE}}}(\lambda) = \Big|\ \Pr[\mathcal{A}(1^\lambda, I^{(0)}, Q^{(0)}) = 1] - \Pr[\mathcal{A}(1^\lambda, I^{(1)}, Q^{(1)}) = 1]\ \Big|$

Figure 2: Definition of $\mathsf{Exp}_{IND}^{\mathsf{PSE}}$ for proximity searchable encryption.

3 IPE to PSE

As mentioned in Section 2, Hamming distance can be calculated using the inner product between the two biometric vectors. As such, we can use a range of possible inner product values as the distance threshold.

Secret key inner product predicate encryption [SSW09] allows one to test if the inner product between two vectors is equal to zero. By appending a value to the first vector and -1 to the second vector, we can support equality testing for non-zero values. Generating several tokens or ciphertexts, one per distance in the range, allows to test if the inner product is below the chosen threshold.
We show that one can use IPE to construct PSE for Hamming distance. At a high level, each keyword is encoded as a {-1, 1} vector and -1 is appended to it, which in turn is encrypted with IPE. Keywords are similarly encoded but this time a distance from the range is appended to them, and tokens generated as part of the underlying IPE scheme.

Construction 1 (Proximity Searchable Encryption)**.** *Fix the security parameter $\lambda \in \mathbb{N}$. Let $\mathsf{IPE} = (\mathsf{IPE.Setup}, \mathsf{IPE.TokGen}, \mathsf{IPE.Encrypt}, \mathsf{IPE.Decrypt})$ be a predicate function hiding IPE scheme over $\mathbb{Z}_q^{n+1}$. Let $\vec{x}_i \in \mathbb{Z}_q^n$ and $X = (\vec{x}_1, \cdots, \vec{x}_\ell)$ be the list of keywords. Let $\mathcal{F}$ be the set of all predicates such that for any $\vec{x}_i \in X$, $f_{\vec{y},t}(\vec{x}_i) = 1$ if the Hamming distance between $\vec{x}_i$ and the query vector $\vec{y} \in \mathbb{Z}_q^n$ is less or equal to some chosen threshold $t \in \mathbb{Z}_q$, $f_{\vec{y},t}(\vec{x}_i) = 0$ otherwise. Figure 3 is a proximity searchable encryption scheme for the Hamming distance.*

Theorem 1 (PSE main theorem)**.** *Let $\mathsf{IPE} = (\mathsf{IPE.Setup}, \mathsf{IPE.TokGen}, \mathsf{IPE.Encrypt}, \mathsf{IPE.Decrypt})$ be an IND-secure function-hiding inner product predicate encryption scheme over $\mathbb{Z}_q^{n+1}$. Then there exists $\mathsf{PSE} = (\mathsf{PSE.Setup}, \mathsf{PSE.BuildIndex}, \mathsf{PSE.Trapdoor}, \mathsf{PSE.Search})$, a secure proximity searchable encryption scheme for the Hamming distance, such that for any PPT adversary $\mathcal{A}_{\mathsf{PSE}}$ for $\mathsf{Exp}_{IND}^{\mathsf{PSE}}$, there exists a PPT adversary $\mathcal{A}_{\mathsf{IPE}}$ for $\mathsf{Exp}_{IND}^{\mathsf{IPE}}$, such that for any security parameter $\lambda \in \mathbb{N}$,*

$$\mathsf{Adv}_{\mathcal{A}_{\mathsf{PSE}}}^{\mathsf{Exp}_{IND}^{\mathsf{PSE}}} = \mathsf{Adv}_{\mathcal{A}_{\mathsf{IPE}}}^{\mathsf{Exp}_{IND}^{\mathsf{IPE}}}$$

Proof. The correctness of the scheme follows from the correctness of the underlying IPE scheme. Assume there exists $\vec{x}_i \in X$, $i \in [1, \ell]$, such that $f_{\vec{y},t}(\vec{x}_i) = 1$. That is $\mathcal{D}(\vec{y}, \vec{x}_i) \leq t$ with $\mathcal{D}(\vec{y}, \vec{x}_i)$ the Hamming distance between vectors $\vec{y}$ and $\vec{x}_i$. Then there exists a unique $\mathsf{tk}_j \in Q_{\vec{y},t}$ such that $b_j \leftarrow \mathsf{IPE.Decrypt}(\mathsf{pp}, \mathsf{tk}_j, \mathsf{ct}_i)$ and $b = 1$ with overwhelming probability by the correctness of the IPE scheme. Now assume that for some $\vec{x}_i \in X$, $i \in [1, \ell]$, we have $f_{\vec{y},t}(\vec{x}_i) = 0$. Then for all $\mathsf{tk}_j \in Q_{\vec{y},t}$, $b_j \leftarrow \mathsf{IPE.Decrypt}(\mathsf{pp}, \mathsf{tk}_j, \mathsf{ct}_i)$ and $b_j = 1$ with negligible probability. Then considering the worst case where either $\mathcal{D}(\vec{y}, \vec{x}_\ell) = t$ or for all $\vec{x}_i \in X$, $f_{\vec{y},t}(\vec{x}_i) = 0$, we have:

$$\begin{aligned}
&\Pr\big[\mathsf{PSE.Search}(\mathsf{pp}, Q_{\vec{y},t}, I_X) = J_{X,\vec{y},t}\big] \\
&\quad \geq 1 - \ell(t+1) \times \Pr\left[\begin{smallmatrix}\mathsf{IPE.Decrypt}(\mathsf{pp},\mathsf{tk}_j,\mathsf{ct}_i) \\ \neq (\mathcal{D}(\vec{x}_i,\vec{y}) \stackrel{?}{=} d_j)\end{smallmatrix}\right] \\
&\quad \geq 1 - \ell(t+1) \times \mathsf{negl}(\lambda).
\end{aligned}$$

We now prove the security of the construction. Let $\mathcal{A}_{\mathsf{PSE}}$ be a PPT adversary for the experiment $\mathsf{Exp}^{\mathsf{PSE}}_{IND}$ and $\mathcal{C}_{\mathsf{IPE}}$ be an challenger for $\mathsf{Exp}^{\mathsf{IPE}}_{IND}$. Then we can build a PPT adversary $\mathcal{A}_{\mathsf{IPE}}$ for the experiment $\mathsf{Exp}^{\mathsf{IPE}}_{IND}$ which works as follows:

1. $\mathcal{A}_{\mathsf{IPE}}$ receives pp from $\mathcal{C}_{\mathsf{IPE}}$ and forwards it to $\mathcal{A}_{\mathsf{PSE}}$.
2. $\mathcal{A}_{\mathsf{IPE}}$ receives two m-query histories $\mathsf{History}^{(0)}, \mathsf{History}^{(1)}$ from $\mathcal{A}_{\mathsf{PSE}}$ where $\mathsf{History}^{(\beta)} = (X^{(\beta)}, F^{(\beta)})$ for $\beta \in \{0,1\}$.
3. For each $\vec{x}_i^{\,(\beta)} \in X^{(\beta)}$, $i \in [1,\ell]$, $\mathcal{A}_{\mathsf{IPE}}$ encodes it as $\vec{x}_i^{\,(\beta)*} \in \{\text{-}1,1\}^n$ and creates the encryption query $S_i = (\vec{x}_i^{\,(0)*} ||\ \text{-}1,\ \vec{x}_i^{\,(1)*} ||\ \text{-}1)$.
4. $\mathcal{A}_{\mathsf{IPE}}$ sets $S = S_1, \cdots, S_\ell$.
5. For each $f_j^{(\beta)} \in F^{(\beta)}$, $j \in [1,m]$:
   (a) $\mathcal{A}_{\mathsf{IPE}}$ extracts a vector $\vec{y}_j^{\,(\beta)} \in \mathbb{Z}_q^n$ and the distance threshold $t \in \mathbb{N}$.
   (b) $\mathcal{A}_{\mathsf{IPE}}$ encodes $\vec{y}_j^{\,(\beta)}$ as $\vec{y}_j^{\,(\beta)*} \in \{\text{-}1,1\}^n$ and creates $D_j^{(0)} = (d_0, \cdots, d_t)$ such that $d_k = n - 2k$ with $0 \le k \le t$.
   (c) $\mathcal{A}_{\mathsf{IPE}}$ creates $D_j^{(0)*}$ by reordering the elements in $D_j^{(0)}$ such that for all $k \in [0,t]$ and $d_k^{(0)} \in D_j^{(0)*}, d_k^{(1)} \in D_j^{(1)}$ we have
   $$\big(\langle \vec{x}_i^{\,(0)}, \vec{y}_j^{\,(0)} \rangle \stackrel{?}{=} d_k^{(0)}\big) = \big(\langle \vec{x}_i^{\,(1)}, \vec{y}_j^{\,(1)} \rangle \stackrel{?}{=} d_k^{(1)}\big).$$
   ($\mathcal{A}_{\mathsf{IPE}}$ can always find a permutation to make this last condition by the admissibility requirement.)
   (d) $\mathcal{A}_{\mathsf{IPE}}$ samples a random permutation $\psi_j : [0,t] \to [0,t]$.
   (e) For $0 \le k \le t$, $\mathcal{A}_{\mathsf{IPE}}$ creates $\vec{y}_j^{\,(\beta)*} ||\ d_k^{(\beta)}$ with $\beta \in \{0,1\}$, $d_k^{(0)} \in D_j^{(0)*}$ and $d_k^{(1)} \in D_j^{(1)}$. Then $\mathcal{A}_{\mathsf{IPE}}$ computes
   $$R_j^{(\beta)} = \psi_j\Big(\ \vec{y}_j^{\,(\beta)*} ||\ d_0^{(\beta)}, \cdots, \vec{y}_j^{\,(\beta)*} ||\ d_t^{(\beta)}\ \Big)$$
   and sets $R_j = (R_j^{(0)}, R_j^{(1)})$.
   (f) $\mathcal{A}_{\mathsf{IPE}}$ sets $R = R_1, \cdots, R_m$.
6. $\mathcal{A}_{\mathsf{IPE}}$ sends the token generation queries R and encryption queries S to $\mathcal{C}_{\mathsf{IPE}}$ and receives back a set of tokens $T^{(\beta)} = \mathsf{tk}_{1,0}^{(\beta)}, \cdots, \mathsf{tk}_{m,t}^{(\beta)}$ and a set of encrypted keywords $C^{(\beta)} = \mathsf{ct}_1^{(\beta)}, \cdots, \mathsf{ct}_\ell^{(\beta)}$ such that
   $$\mathsf{tk}_{j,k}^{(\beta)} \leftarrow \mathsf{IPE.TokGen}(\mathsf{sk},\ \vec{y}_j^{\,(\beta)*} ||\ d_k^{(\beta)})$$
   $$\mathsf{ct}_i^{(\beta)} \leftarrow \mathsf{IPE.Encrypt}(\mathsf{sk},\ \vec{x}_i^{\,(\beta)*} ||\ \text{-}1)$$
   for $i \in [1,\ell]$, $j \in [1,m]$, $k \in [0,t]$ and $\beta \in \{0,1\}$. $\mathcal{A}_{\mathsf{MPIPE}}$ forwards $T^{(\beta)}$ and $C^{(\beta)}$ to $\mathcal{A}_{\mathsf{PSE}}$, respectively as the encrypted index $I^{(\beta)}$ and the list of queries $Q^{(\beta)}$.
7. $\mathcal{A}_{\mathsf{IPE}}$ receives $\beta' \in \{0,1\}$ from $\mathcal{A}_{\mathsf{PSE}}$ and returns it.

Since the number of token generation queries, $m \times t$, sent by $\mathcal{A}_{\mathsf{IPE}}$ remains polynomial in the security parameter, the advantage of $\mathcal{A}_{\mathsf{PSE}}$ is

$$\mathsf{Adv}^{\mathsf{Exp}^{\mathsf{PSE}}_{IND}}_{\mathcal{A}_{\mathsf{PSE}}} = \mathsf{Adv}^{\mathsf{Exp}^{\mathsf{IPE}}_{IND}}_{\mathcal{A}_{\mathsf{IPE}}}$$

This completes the proof of Theorem 1. □

Choosing an $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ scheme Table 1 presents the resulting efficiency of PSE schemes based on different $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ constructions. This table corresponds to $t+1$ tokens with all operations on dimension $n+1$. Barbosa et al. presented a concept implementation and benchmarked it across dimensions [BCSW19].

The following two sections (4 and 5) address two bottlenecks in the construction: 1) Both t and n are dependent on the dimension of the biometric, usually $t = \Theta(n)$, 2) The quadratic size (in the dimension n) of the secret key. The size of the secrete key corresponds to client storage. Barbosa et al. [BCSW19] would yield the most efficient scheme in all other aspects. However, a dimension of 1024 for a modest group size of 256 bits would translate to 500 Mb of storage.

$\underline{\mathsf{PSE.Setup}(1^\lambda) \to (\mathsf{sk}, \mathsf{pp})}$:
Run and output $(\mathsf{sk}, \mathsf{pp}) \leftarrow \mathsf{IPE.Setup}(1^\lambda)$.

$\underline{\mathsf{PSE.Trapdoor}(\mathsf{sk}, f_{\vec{y},t}) \to Q_{\vec{y},t}}$:

1. For $i = 0$ to t, compute $d_j = n - 2j$,
2. Set $D = (d_0, ..., d_t)$,
3. Sample random permutation $\pi : [0, t] \to [0, t]$,
4. Compute $D^* = \pi(D) = \{d_1^*, \cdots, d_t^*\}$,
5. Encode $\vec{y}$ as $\vec{y}^* \in \{-1, 1\}^n$,
6. For $1 \le j \le t$ call $\mathsf{tk}_j \leftarrow \mathsf{IPE.TokGen}(\mathsf{sk}, \vec{y}^* || \ d_j^*)$.,
7. Output $Q_{\vec{y},t} = (\mathsf{tk}_1, \cdots, \mathsf{tk}_t)$.

$\underline{\mathsf{PSE.BuildIndex}(\mathsf{sk}, X) \to I_X}$:

1. For each keyword $\vec{x}_i \in X$, $i \in \{1, \cdots, \ell\}$, encode $\vec{x}_i^* \in \{-1, 1\}^n$, compute $\mathsf{ct}_i \leftarrow \mathsf{IPE.Encrypt}(\mathsf{sk}, \vec{x}_i^* || \text{ -1})$.
2. Outputs $I_X = (\mathsf{ct}_1, \cdots, \mathsf{ct}_\ell)$.

$\underline{\mathsf{PSE.Search}(\mathsf{pp}, Q_{\vec{y},t}, I_X) \to J_{X,\vec{y},t}}$:

1. Initialize $J_{X,\vec{y},t} = \emptyset$.
2. For each $\mathsf{ct}_i \in I_X$ and for each $\mathsf{tk}_j \in Q_{\vec{y},t}$, call $b_j \leftarrow \mathsf{IPE.Decrypt}(\mathsf{pp}, \mathsf{tk}_j, \mathsf{ct}_i)$. If $b_j = 1$, add i to $J_{X,\vec{y},t}$, continue to ct_{i+1}.
3. Outputs $J_{X,\vec{y},t}$.

Figure 3: Construction of proximity search from $\mathsf{IPE}_{\mathsf{fh},\mathsf{sk},\mathsf{pred}}$.

	Underlying IPE scheme				
	Multi-bases ($\sigma = n$)	[BCSW19] (GGM)	[BCSW19] (SM)	[SSW09]	[KT14]
secret key	$8n + 10$	$2(n+1)^2 + 2$	$24n + 42$	$4n + 8$	$60(n+1)^2$
index	$2\ell(n+1)$	$\ell(n+1)$	$\ell(6n+12)$	$\ell(4n+8)$	$6\ell(n+1)$
query	$2(t+1)(n+1)$	$(t+1)(n+1)$	$(t+1)(6n+12)$	$(t+1)(4n+8)$	$6(t+1)(n+1)$
buildindex	$2\ell(n+1)$	$\ell(n+1)$	$\ell(12n+21)$	$\ell(32n+36)$	$6\ell(n+1)$
trapdoor	$2(t+1)(n+1)$	$(t+1)(n+1)$	$(t+1)(12n+21)$	$(t+1)(24n+40)$	$6(t+1)(n+1)$
search	$2\ell(t+1)(n+1)$	$\ell t(n+1)$	$\ell(t+1)(6n+12)$	$\ell(t+1)(4n+8)$	$6\ell(t+1)(n+1)$

Table 1: PSE scheme efficiency for keywords of size n depending on underlying $\mathsf{IPE}_{\mathsf{fh},\mathsf{sk},\mathsf{pred}}$ scheme. Upper part of the table shows number of group elements per component. Lower part of the table shows number of group or pairing operations per function. The number n is the length of the biometric template, t is the desired distance tolerance, and ℓ is the total number of records in the database.

4 Iris Statistics and Leakage

Biometrics represent a unique target for encrypted databases as their statistical properties are stable and well-understood. In this section we show that the leakage of our proximity search construction is uniquely suited to the the well spread nature of biometrics and the goals of biometric databases. We focus on the iris; iris feature extractors usually result in templates compared using the Hamming metric.[5]

Daugman [Dau05, Dau09] introduced the seminal iris processing pipeline. This pipeline assumes a near infrared camera. Note that iris images in near infrared are believed to be independent from the visible light pattern and that the iris is epigenetic, irises of identical twins are believed to be independent [Dau09, HBF10]. Traditional iris recognition consists of three phases:

Segmentation This takes the image and identifies which pixels should be included as part of the iris. This produces a $\{0, 1\}$ matrix of the same size as the input image with 1s corresponding to iris pixels.

Normalization This takes the variable size set of iris pixels and maps them to a fixed size rectangular array. This can roughly be thought of as unrolling the iris.

Feature Extraction In the final stage the fixed size rectangular array is used for feature extraction. In Daugman's original work this consisted of convolving small areas of the rectangle with a fixed 2D wavelet. Modern feature extractors are usually convolutional neural networks.

[5] In fingerprint systems, the metric is usually set difference, in facial recognition, the metric is usually the L2 distance.

In identification systems the natural tradeoff for the system is between *true accept rate* (TAR) and *false accept rate* (FAR). TAR is how frequently readings of the *same* biometric are regarded as the same. FAR is how frequently readings of *different* biometrics are regarded as the same. As described above, when one wishes to match a biometric y against a database one considers matches as the set $\{x_i | \mathcal{D}(x_i, y) \leq t\}$ for some metric $\mathcal{D}$ and distance parameter t. Selecting a small t lowers both TAR and FAR. As described in the previous section for a query y our construction leaks the following information:

Access Pattern $\{x_i | \mathcal{D}(x_i, y) \leq t\}$

Distance Equality Leakage $\{\mathcal{D}(x_i, y) \stackrel{?}{=} \mathcal{D}(x_j, y) | \mathcal{D}(x_i, y) \leq t \wedge \mathcal{D}(x_j, y) \leq t\}$.

Thus, the setting of t additionally affects the leakage of the system. There are two types of biometric databases, those which associate a single reading x_i of a biometric with each record r_i and those where multiple readings of a biometric $x_{i,1}, ..., x_{i,k}$ are associated with a single record. We separately discuss the impact of leaking information about multiple biometrics (Sec 4.1) and leaking information about multiple readings of the same biometric (Sec 4.2). We believe leaking about multiple biometrics is more harmful and consider this first. Also in the setting where at most one reading is stored, the second type of leakage does not apply.

4.1 Leakage on different irises

We now consider what can be inferred about two different biometrics $x_{i,\alpha}$ and $x_{j,\beta}$. We assume that there may be multiple readings of each biometric present in the database. By construction, information is leaked about two different biometrics when they are jointly returned by a query. The chance of this happening is bounded by the FAR of the corresponding identification system. Understanding FAR of a system requires specification of 1) the feature extractor, 2) the biometric database, and 3) the queries that will be issued by the client.

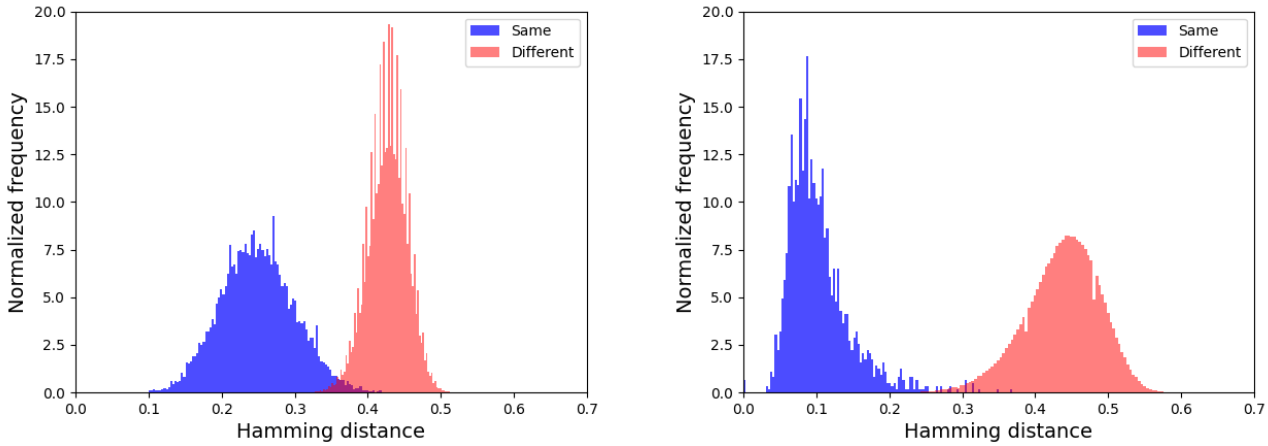
Feature Extractor For the feature extractor, we use the recent pipeline called ThirdEye [AF18, AF19] which is publicly available [Ahm20]. This feature extractor produces a 1024 dimensional real valued feature vector. We convert this to a binary vector by setting $f'_i = 1$ if $f_i > \mathsf{Exp}[f_i]$ where the expectation is for a single feature vector output, otherwise $f'_i = 0$. We train the feature extractor as specified in [AF19].

Biometric Database There are many iris datasets collected across a variety of conditions. We consider two datasets in this work. First, the NotreDame 0405 dataset [PSO+09, BF16] which is a superset of the NIST Iris Evaluation Challenge [PBF+08]. This dataset consists of images from 356 biometrics (we consider left and right eyes as separate biometrics) with 64964 images in total. Second, the IITD dataset [KP10] consists of 224 persons and 2240 images. The IITD dataset is considered "easier" than the ND0405 dataset because images are collected in more controlled environments leading to less noise and variation between images. Figure 4 shows the histograms for testing portions of both datasets. The blue histogram contains comparisons between different readings of the same biometric while the red histogram contains comparisons between different biometrics. Let $t' = t/1024$ be the fractional Hamming distance, the TAR is the fraction of the blue histogram to the left of t' and the FAR is the fraction of the red histogram to the left of t'. Note that these two datasets produce different statistics. In both cases there is overlap between the red and blue histogram indicating that there is a tradeoff between TAR and FAR.

Distribution over Queries There are two natural settings to consider for the distribution over queries: 1) arbitrary or 2) distributed identically to biometric readings.

Let t_{min} be the minimum observed distance between two different biometrics. If the query value y^* is assumed to be arbitrary (not the result of an biometric reading), one may begin to see leakage once the allowed query distance is at least $t = t_{min}/2$.

For both the ND0405 and IITD datasets, if one assumes that queries are arbitrary and wishes to ensure that multiple biometrics are not returned this produces a system with poor TAR. For ND0405, $t_{min} = .302$ if one sets a system with $t = t_{min}/2$ it only has a TAR of 1.4%. Similarly, for IITD, $t_{min} = .266$ a system with $t = t_{min}/2$ produces a system with TAR of 16.4%. For the rest of this section we assume that queries result from a biometric reading. The resulting system must have a mechanism to ensure the client only queries real biometrics not arbitrary values.



(a) ND 0405 Histogram

(b) IITD Histogram

Figure 4: Hamming distance distribution for images from the same iris in blue, and different irises in red. Histograms are produced using ThirdEye [AF19]. Resulting histograms for the ND 0405 and IITD datasets respectively.

However, if the query value y^* is assume to be drawn from the biometric distribution one can use TAR and FAR at different values of t to understand the correctness and probability of leakage between different biometrics respectively.

We now investigate the tradeoff between TAR and FAR for the two datasets. We stress that FAR maps to leakage probability between different readings when the biometric database is the same size and is drawn from the same underlying distribution as the dataset. As the number of records grows the tail of the red histogram will have more points meaning that more geometry is revealed (additional discussion in Section 7).

For different FAR levels, the first row of Table 2 shows the corresponding TAR for the ND0405 and IITD datasets. For both the ND0405 and IITD datasets one can achieve a 95% TAR with a FAR of only 1%.

4.2 Leakage on readings of an iris

We now consider the implications of leakage between readings of the same biometric. That is, $x_{i,1}, ..., x_{i,k}$ are readings from the same biometric and associated with a record r_i in the biometric database. First note that $x_{i,\alpha}$ and $x_{i,\beta}$ are likely to be close together (because readings of the same biometric are similar).

One may able to infer information about $x_{i,1}, ..., x_{i,k}$ from access pattern and distance equality leakage. One may be able to learn the relative positioning of the different readings by which values $\mathcal{I}$ are return by a query y (if it is not all values). Similarly, we expect the adversary to learn distance equality leakage for the entire set $x_{i,1}, ..., x_{i,k}$. Both of these leakage profiles allow an adversary to construct geometry of a biometric's different readings. This may allow the adversary to determine the type of noise present in that individual's biometric. It may be possible to use noise rates to draw conclusions about sensitive attributes about the corresponding person. Biometric systems frequently demonstrate systemic bias [DRD+20]. As one example most datasets draw from volunteer undergraduates students. Systems accuracy varies based on sensitive attributes such as gender, race, an age (see [DRD+20, Table 1]). Thus one may be able to infer sensitive attributes based on the relative size of $|\mathcal{I}|/k$.

Hiding this information seems to require fundamentally new techniques if one does not resort to using oblivious RAM. We leave this as an important open problem. Our recommendation is to use biometric averaging techniques [ZD08] to produce a representative value and make other values associated data that are not searchable.

4.3 Reducing Dimension

For the construction in Section 3 for a biometric of dimension n, the query size is proportional to $\Theta(tn)$ and checking if a record matches requires $\Theta(tn)$ group operations per record in the database. Since t is usually a constant fraction of n, reducing n can lead to a quadratic reduction in communication and query efficiency. We consider three different

mechanisms for dimension reduction and consider their impact on TAR/FAR. The three mechanisms we consider are: 1) random global sampling of output features, 2) selection of output features based on their relative "quality," and 3) training a feature extractor with fewer dimensions.

Random Sampling As our first dimensionality reduction technique we consider a fixed random sample of the 1024 output dimensions for varying sizes $512, 256, 192, 168, 96, 64, 32, 16$. For all experiments we computed four different samples and report the average of the TAR in Table 2 for the ND0405 and IITD datasets. Variance increases as the sample size decreases.[6]

For the IITD dataset, one can safely subsample to a much smaller output dimension with little effect on the TAR/FAR tradeoff. Setting $n = 64$ seems to have little effect on accuracy but would lead to a $(1024/64)^2 = 256$ decrease in query size and server computation. The tradeoff is more delicate for the ND0405 dataset owing to the larger overlap between the two histograms but one can still achieve moderate accuracy for 96 features (which results in a roughly 100 fold efficiency improvement).

Structured Sampling We take two approaches to structured sampling of dimensions. First, Hollingsworth et al. [HBF08] and Bolle et al. [BPCR04] propose the concept of "fragile bits" which are more likely to be susceptible to bit flips. Their work is based on the Gabor based feature extractor (described at the beginning of this section) while ThirdEye [AF19] is a convolutional neural network.

For our first approach we utilize 64 most "stable" bits which have the least probability of flipping or have the lowest variance. Results for this approach are shown in Table 2 and denoted by S-64 (for stable).

Surprisingly, this approach hurts performance when compared to all other results for 64 features. We believe this approach to be appropriate for the Gabor based feature extractor since it produces large number of noisy features due to noise in different readings of an iris. This is in contrast to our feature extractor which outputs a succinct feature vector with each individual feature being independent leaving less room for 'fragile' bits.

Our second approach uses bits which maximize the difference between the means of the intra and inter class distributions. Essentially we generate histograms as in Fig 4 using a single bit for every bit in the feature extractor output of size 1024. That is, we select the bits that maximize the following difference:

$$\max_i \left(\Pr_{x,y \leftarrow \text{Different}}[x_i \neq y_i] - \Pr_{x,y \leftarrow \text{Same}}[x_i \neq y_i] \right)$$

The intuition is that bits are the most useful as they maximize the difference in probability of error between the same and different comparisons. The hope is to overcome the weakness of the prior approach which did not consider the entropy of bits across different biometrics. The top 64 bits maximizing difference between difference and same distributions are used. Results for this approach are shown in Table 2 and denoted by E-64 (for error). This approach improves over random sampling in contrast to only considering bit fragility.

Training a new feature extractor Lastly, we train the ThirdEye architecture [AF19] from scratch to output a smaller feature vector of size $n = 64$ for both datasets. Essentially we train a new feature extractor on the same training data to reduce dimensions. The feature extractor remains the same but is now constrained to learn features of size 64. This is achieved by changing the number of neurons in the second last layer of our convolutional neural network. We can expect this to perform better than random sampling since the feature extractor is explicitly learning to classify using 64 features. Results are shown in Table 2 and denoted by T-64 (for train).

Average TAR increases for both datasets. The increase is more pronounced for ND0405 where a feature vector of size 64 performs on par with a randomly subsampled feature vector of size 128. For the IITD dataset we see TAR increase for the case with 0 FAR. A small TAR decrease is noticed for FAR at $0.01, 0.02$ and 0.03, we attribute this to slight variation between results and the stochastic nature of the training process.

Recommendation Training a new network and using those bits that maximize the difference in error rates have similar performance and outperform random subsampling. Both of these techniques with 64 output dimensions can be used on data similar to the IITD dataset with almost no performance impact (compared to the full $n = 1024$)

[6] This is consistent with previous observations that sampling from the iris red histogram behaves similarly to a binomial distribution where the number of trials is proportional the included entropy of the iris [SSF19].

TAR at Size	False Accept Rate: ND 0405 Dataset 0	.01	.02	.03	.04	.05	.06	.07	.08	.09	.10	IITD Dataset 0	.01	.02	.03	.04	.05	.06	.07	.08	.09	.10
1024	.50	.97	.98	.99	.99	.99	.99	.99	.99	1	1	.70	1	1	1	1	1	1	1	1	1	1
512	.35	.96	.97	.98	.98	.99	.99	.99	.99	.99	.99	.57	1	1	1	1	1	1	1	1	1	1
256	.25	.93	.95	.96	.97	.97	.98	.98	.98	.98	.98	.47	.99	1	1	1	1	1	1	1	1	1
192	.18	.89	.92	.94	.95	.96	.96	.97	.97	.97	.98	.48	.99	1	1	1	1	1	1	1	1	1
128	.10	.83	.88	.90	.92	.92	.93	.94	.95	.95	.96	.54	.99	.99	1	1	1	1	1	1	1	1
96	.05	.76	.81	.85	.87	.89	.90	.91	.92	.92	.94	.40	.99	.99	.99	1	1	1	1	1	1	1
64	.01	.62	.71	.76	.78	.82	.83	.84	.86	.87	.88	.27	.98	.99	.99	.99	.99	.99	1	1	1	1
32	.00	.36	.44	.52	.54	.60	.65	.67	.67	.69	.72	.04	.92	.96	.96	.97	.98	.98	.98	.99	.99	.99
16	.00	.18	.21	.24	.36	.39	.39	.39	.39	.44	.48	.00	.56	.76	.76	.81	.89	.89	.89	.89	.89	.92
64	.01	.62	.71	.76	.78	.82	.83	.84	.86	.87	.88	.27	.98	.99	.99	.99	.99	.99	1	1	1	1
S-64	.00	.39	.39	.49	.59	.59	.59	.68	.68	.68	.74	.0	.61	.72	.79	.86	.86	.91	.91	.91	.93	.95
E-64	.03	.70	.76	.82	.86	.86	.90	.90	.90	.93	.93	.31	.99	1	1	1	1	1	1	1	1	1
T-64	.04	.73	.84	.87	.87	.91	.91	.94	.94	.94	.96	.40	.96	.98	.98	.99	.99	.99	1	1	1	1

Table 2: TAR for different output sizes and probabilities of leakage for the ND0405 and IITD datasets. Summary of true accept rates for queries drawn from *Same* distribution. We vary a threshold t, report the true accept rate (TAR) when allowing for the corresponding FAR which is related to system leakage. The bottom four rows compare random sampling of 64 bits to the three structured approaches discussed in Section 4.3. The 64 random row is duplicated for comparison.

except if one desires FAR of .00. In the ND0405 dataset there is more overlap between the two distance histograms. For the ND0405 reducing dimension does harm the TAR/FAR tradeoff.

As expected, the tradeoff between correctness and security is complex, depending on the cryptographic approach, expected data, and query distribution.

5 Reducing Client Storage

As described in the introduction, we show a general technique for reducing the size of keys of a secret key, function hiding, predicate IPE scheme. The key idea of the technique is to use a different pair of matrices for each portion of the input vectors. These independent encodings are then are combined with a additive secret sharing of 0 in the encryption so that computation with ciphertexts is only useful when using all of the components. This technique modifies the scheme of Barbosa et al. [BCSW19, Section 4].[7]

Construction The construction is in Figure 5. The resulting size of each component is in Table 3. We first argue correctness and then security. For security we show the scheme satisfies a stronger simulation based definition of security, as in the work of Barbosa et al. [BCSW19].

Correctness First note that $\langle \vec{x}, \vec{y} \rangle = \sum_{\ell=1}^{\sigma} \langle \vec{x}_\ell, \vec{y}_\ell \rangle$, and thus

$$\begin{aligned}\Pi_{\ell=1}^{\sigma}\Pi_{i=1}^{N} e(\mathsf{tk}_\ell[i], \mathsf{ct}_\ell[i]) &= g_T^{\sum_{\ell=1}^{\sigma} \beta\cdot(\vec{x'}_\ell)^T\cdot\mathbb{B}_\ell^*\cdot\mathbb{B}_\ell^T\cdot\alpha\cdot(\vec{y'}_\ell)} \\ &= g_T^{\sum_{\ell=1}^{\sigma} \beta\cdot(\vec{x'}_\ell)^T\cdot\alpha\cdot(\vec{y'}_\ell)} = g_T^{\alpha\beta\sum_{\ell=1}^{\sigma}\zeta_\ell+\langle\vec{x}_\ell,\vec{y}_\ell\rangle} \\ &= g_T^{\alpha\beta\cdot\langle\vec{x},\vec{y}\rangle+\alpha\beta\cdot\sum_{\ell=1}^{\sigma}\zeta_\ell} = g_T^{\alpha\beta\cdot\langle\vec{x},\vec{y}\rangle}\end{aligned}$$

If the inner product of $\vec{x}$ and $\vec{y}$ vectors is zero then $\Pi_{\ell=1}^{\sigma}\Pi_{i=1}^{N} e(\mathsf{tk}_\ell[i], \mathsf{ct}_\ell[i]) = e(g_1, g_2)^0 = 1$, which is the identity element in $\mathbb{G}_T$ and is easily detectible and $\top \leftarrow \mathsf{Decrypt}(\mathsf{pp}, \mathsf{tk}, \mathsf{ct})$ with probability 1. However, if $\langle \vec{x}, \vec{y} \rangle \neq 0$, then the probability that $\top \leftarrow \mathsf{Decrypt}(\mathsf{pp}, \mathsf{tk}, \mathsf{ct})$ is $\Pr[\alpha\beta \cdot \langle \vec{x}, \vec{y} \rangle = 0] \leq \frac{2}{q}$.

Definition 7 (Simulation-based security)**.** *Let $\mathsf{IPE} = (\mathsf{IPE.Setup}, \mathsf{IPE.TokGen}, \mathsf{IPE.Encrypt}, \mathsf{IPE.Decrypt})$ be a predicate IPE scheme over $\mathbb{Z}_q^n$. Then IPE is SIM-secure if for all PPT adversaries $\mathcal{A}$, there exist a simulator $\mathcal{S}$ such that for the experiment $\mathsf{Exp}_{SIM}^{\mathsf{IPE}}$ described in figure 6, the advantage of $\mathcal{A}$ ($\mathsf{adv}_{\mathcal{A}}^{\mathsf{Exp}_{SIM}^{\mathsf{IPE}}}$) is*

$$\left|\ \Pr[1 \leftarrow \mathsf{Real}_{\mathsf{IPE},\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathsf{Ideal}_{\mathsf{IPE},\mathcal{A}}(1^\lambda)]\ \right|$$

which is $\mathsf{negl}(\lambda)$.

[7] Functional encryption for orthogonality (OFE) as defined by Barbosa et al. is equal to predicate inner product encryption, as defined in this work.

$\mathsf{Setup}(1^\lambda, n, \sigma)$:

1. Samples $(G_1, G_2, G_T, q, e) \leftarrow \mathcal{G}_{abg}$ and randomly samples generators $g_1 \in G_1$ and $g_2 \in G_2$.
2. For $1 \leq \ell \leq \sigma$, randomly samples an invertible square matrix $\mathbb{B}_\ell \in \mathbb{Z}_q^{N \times N}$ and sets $\mathbb{B}_\ell^* = (\mathbb{B}_\ell^{-1})^T$.
3. Outputs $\mathsf{pp} = (G_1, G_2, G_T, q, e, n, \sigma)$ as public parameters and $\mathsf{sk} = (g_1, g_2, \{\mathbb{B}_\ell, \mathbb{B}_\ell^*\}_{\ell=1}^{\sigma})$.

$\mathsf{TokGen}(\mathsf{pp}, \mathsf{sk}, \vec{y})$:

1. Samples $\alpha \xleftarrow{\$} \mathbb{Z}_q$.
2. Splits $\vec{y}$ into σ subvectors $\vec{y}_\ell$ of size n/σ.
3. For $1 \leq \ell \leq \sigma$, defines $\vec{y}'_\ell = 1 \,||\, \vec{y}_\ell$ and sets $\mathsf{tk}_\ell = [\ \alpha \cdot (\vec{y}'_\ell)^T \cdot \mathbb{B}_\ell\ ]_1$, a vector in G_1.
4. Outputs $\mathsf{tk} = (\mathsf{tk}_1, \cdots, \mathsf{tk}_\sigma)$.

$\mathsf{Encrypt}(\mathsf{pp}, \mathsf{sk}, \vec{x})$:

1. Samples $\beta \xleftarrow{\$} \mathbb{Z}_q$.
2. Splits $\vec{x}$ into σ subvectors $\vec{x}_\ell$ of size n/σ.
3. For $1 \leq \ell \leq \sigma - 1$, samples $\zeta_\ell \xleftarrow{\$} \mathbb{Z}_q$ then sets $\zeta_\sigma = -\sum_{\ell=1}^{\sigma-1} \zeta_\ell$.
4. For $1 \leq \ell \leq \sigma$ defines $\vec{x}'_\ell = \zeta_\ell \,||\, \vec{x}_\ell$ and sets $\mathsf{ct}_\ell = [\ \beta \cdot (\vec{x}'_\ell)^T \cdot \mathbb{B}_\ell^*\ ]_2$, a vector in G_2.
5. Outputs $\mathsf{ct} = (\mathsf{ct}_1, \cdots, \mathsf{ct}_\sigma)$.

$\mathsf{Decrypt}(\mathsf{pp}, \mathsf{tk}, \mathsf{ct})$:

Computes $\left(\Pi_{\ell=1}^{\sigma} \Pi_{i=1}^{N} e(\mathsf{tk}_\ell[i], \mathsf{ct}_\ell[i])\right)$ and returns $\top$ if the results is equal to $1 \in \mathbb{G}_T$, $\perp$ otherwise.

Figure 5: Construction of $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ in the Generic Group Model.

Component	Number of Group Elements
Secret Key	$2n^2/\sigma + 4n + 2\sigma + 2$
Ciphertext	$n + \sigma$
Token	$n + \sigma$

Table 3: Sizes in Group Elements of Each Component of Revised Scheme. The value σ is how many distinct matrices are used. Setting $\sigma = \Omega(n)$ makes all components a linear number of group elements. The scheme of Barbosa et al. [BCSW19] has a secret key of size $2n^2 + 2$ and token/ciphertext of size n.

From [KLM+16, Remark 2.5] they show that the simulation based security above implies to the indistinguishability based definition, so we argue that the scheme in Figure 5 satisfies Definition 7 which implies Definition 1.

Theorem 2. *In the Generic Group Model for asymmetric bilinear groups the construction in Figure 5 is a secure $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ scheme according to Definition 7 for the family of predicates $\mathcal{F} = \{f_y | y \in \mathbb{Z}_q^n\}$ such that for all vectors $x \in \mathbb{Z}_q^n$, $f_y(x) = (\langle x, y \rangle \stackrel{?}{=} 0)$.*

Proof of Theorem 2. This scheme has the security as the original $\mathsf{IPE}_{\mathsf{fh,sk,pred}}$ scheme from [BCSW19] for the simulation based security definition. We note that that scheme of Barbosa et al. [BCSW19] builds on the work Kim et al. [KLM+16] and our proof uses similar definitions of formal variables. The scheme works by having a challenger interact with a simulator S and two oracles, $\mathcal{O}'_{\mathsf{TokGen}}$ and $\mathcal{O}'_{\mathsf{Encrypt}}$, in the ideal scheme and a pair of oracles, $\mathcal{O}_{\mathsf{TokGen}}$ and $\mathcal{O}_{\mathsf{Encrypt}}$, in the real scheme.

For this proof, we will build the simulator $\mathcal{S}$ which can correctly simulate the distribution of tokens and ciphertexts only using the predicate evaluation on whether the inner product of the two vectors is 0. This information is supplied to the simulator by the oracles $\mathcal{O}'_{\mathsf{TokGen}}$ and $\mathcal{O}'_{\mathsf{Encrypt}}$ to match the functionality of the encryption scheme.

Inner-product collection: Let i, j be shared counters between the token generation and encryption oracles. Let $x^{(i)} \in \mathbb{Z}_q^n$ and $y^{(j)} \in \mathbb{Z}_q^n$ denote respectively the adversary's i^{th} query to the token generation oracle and j^{th} query to the encryption oracle. The collection of mappings $\mathcal{C}_{\mathsf{ip}}$ is defined as

$$\mathcal{C}_{\mathsf{ip}} = \begin{cases} (i,j) \to 0 & \text{if } \langle x^{(i)}, y^{(j)} \rangle = 0 \\ (i,j) \to 1 & \text{otherwise.} \end{cases}$$

Formal variables: The simulator constructs formal variables for the unknowns of the system in order to respond as correctly as possible. Let Q be the maximum number of queries made by an adversary. Let σ and N be as in the construction in Figure 5. For all $i \in [Q]$, $\ell \in [\sigma]$ and $k \in [N]$, let $\hat{\alpha}^{(i)}, \hat{\beta}^{(i)}, \hat{x}_{\ell,k}^{(i)}, \hat{y}_{\ell,k}^{(i)}$ represent the hidden variables

$\underline{\mathsf{Real}_{\mathsf{IPE},\mathcal{A}}(1^\lambda)}$	$\underline{\mathsf{Ideal}_{\mathsf{IPE},\mathcal{A}}(1^\lambda)}$
$(\mathsf{sk},\mathsf{pp}) \leftarrow \mathsf{IPE.Setup}(1^\lambda)$	$(\mathsf{sk},\mathsf{pp}) \leftarrow \mathsf{IPE.Setup}(1^\lambda)$
$b \leftarrow \mathcal{A}^{\mathsf{IPE.TokGen}(\mathsf{sk},\cdot),\mathsf{IPE.Encrypt}(\mathsf{sk},\cdot)}(1^\lambda)$	$b \leftarrow \mathcal{A}^{\mathcal{S}(\Phi(\cdot))}(1^\lambda)$
Output b	Output b

Figure 6: Definition of experiment $\mathsf{Exp}^{\mathsf{IPE}}_{SIM}$. Φ denotes the information leakage received by the simulator $\mathcal{S}$ such that $\Phi(i,j) = f_{y_j}(x_i)$ for all i, j.

$\alpha^{(i)}, \beta^{(i)}, x^{(i)}_{\ell,k}, y^{(i)}_{\ell,k}$, let $\hat{b}_{\ell,k,m}$ and $\hat{b}^*_{\ell,k,m}$ represent the entry in position (k, m) of the $\mathbb{B}_\ell$ and $\mathbb{B}^*_\ell$ matrices respectively, let $\hat{\zeta}^{(i)}_\ell$ be the formal variables for $\zeta^{(i)}_\ell$ where the simulator tracks the constraints that for each $i \in [Q]$, $\sum_{\ell=1}^{\sigma} \hat{\zeta}^{(i)}_\ell = 0$ and let $\hat{s}^{(i)}_{\ell,m}$ and $\hat{t}^{(i)}_{\ell,m}$ represent formal polynomials as constructed below,

$$\hat{s}^{(i)}_{\ell,m} = \sum_{k=1}^{N} \hat{y}'^{(i)}_{\ell,k} \cdot \hat{b}_{\ell,k,m} = \hat{b}_{\ell,1,m} + \sum_{k=2}^{N} \hat{y}^{(i)}_{\ell,k\text{-}1} \cdot \hat{b}_{\ell,k,m} \tag{1}$$

$$\hat{t}^{(i)}_{\ell,m} = \sum_{k=1}^{N} \hat{x}'^{(i)}_{\ell,k} \cdot \hat{b}^*_{\ell,k,m} = \hat{\zeta}^{(i)}_\ell \cdot \hat{b}^*_{\ell,1,m} + \sum_{k=2}^{N} \hat{x}^{(i)}_{\ell,k\text{-}1} \cdot \hat{b}^*_{\ell,k,m} \tag{2}$$

Then the universe of formal variables is $\mathcal{U} = \mathcal{R} \cup \mathcal{T}$, where

$$\mathcal{R} = \left\{\hat{\alpha}^{(i)}, \hat{\beta}^{(i)}\right\}_{i\in[Q]} \cup \left\{\hat{s}^{(i)}_{\ell,m}\ ,\ \hat{t}^{(i)}_{\ell,m}\right\}_{i\in[Q],\ \ell\in[\sigma],\ m\in[N]}$$

and

$$\mathcal{T} = \left\{\hat{\alpha}^{(i)}, \hat{\beta}^{(i)}\right\}_{i\in[Q]} \cup \left\{\hat{x}'^{(i)}_{\ell,k},\ \hat{y}'^{(i)}_{\ell,k},\ \hat{\zeta}^{(i)}_\ell\right\}_{i\in[Q],\,\ell\in[\sigma],\,k\in[N]} \cup \left\{\hat{b}_{\ell,k,m}\ ,\ \hat{b}^*_{\ell,k,m}\right\}_{\ell\in[\sigma],\,m,k\in[N]}$$

Specification of the simulator Let $\mathcal{A}$ be a PPT adversary that makes at most $Q = \mathtt{poly}(\lambda)$ queries to the oracles. The simulator $\mathcal{S}$ starts by initializing an empty set of inner products $\mathcal{C}_{\mathsf{ip}}$ and three empty tables T_1, T_2, T_T which map handles to the polynomials over the variables of $\mathcal{R}$. The state of the simulator consists of these four objects, $(\mathcal{C}_{\mathsf{ip}}, T_1, T_2, T_T)$, which are updated after each query received. The simulator $\mathcal{S}$ answers the adversary's queries as follows.

Token generation queries: On input $x^{(i)} \in Z_q^n$, $\mathcal{O}'_{\mathsf{TokGen}}$ sends the collection $\mathcal{C}'_{\mathsf{ip}}$ to the simulator. $\mathcal{S}$ updates $\mathcal{C}_{\mathsf{ip}} \leftarrow \mathcal{C}'_{\mathsf{ip}}$. For $1 \le \ell \le \sigma$, $1 \le m \le N$, $\mathcal{S}$ generates a new handle $h_{\ell,m} \xleftarrow{\$} \{0,1\}^\lambda$ and adds the mapping $h_{\ell,m} \to \hat{\alpha}^{(i)} \cdot \hat{s}^{(i)}_{\ell,m}$ to T_1. $\mathcal{S}$ then sets $\mathsf{tk}_\ell = h_{\ell,1}, \cdots, h_{\ell,N}$. Finally, $\mathcal{S}$ returns the token $\mathsf{tk} = (\mathsf{tk}_1, \cdots, \mathsf{tk}_\sigma)$.

Encryption queries: On input $y^{(i)} \in Z_q^n$, $\mathcal{O}'_{\mathsf{Encrypt}}$ sends the collection $\mathcal{C}'_{\mathsf{ip}}$ to the simulator. $\mathcal{S}$ updates $\mathcal{C}_{\mathsf{ip}} \leftarrow \mathcal{C}'_{\mathsf{ip}}$. For $1 \le \ell \le \sigma$, $1 \le m \le N$, $\mathcal{S}$ generates a new handle $h_{\ell,m} \xleftarrow{\$} \{0,1\}^\lambda$ and adds the mapping $h_{\ell,m} \to \hat{\beta}^{(i)} \cdot \hat{t}^{(i)}_{\ell,m}$ to T_2. $\mathcal{S}$ sets $\mathsf{ct}_\ell = h_{\ell,1}, \cdots, h_{\ell,N}$. Finally, $\mathcal{S}$ returns the ciphertext $\mathsf{ct} = (\mathsf{ct}_1, \cdots, \mathsf{ct}_\sigma)$.

Addition oracle queries: Given $h_1, h_2 \in \{0,1\}^\lambda$, $\mathcal{S}$ verifies that formal polynomials p_1, p_2 exist in table T_τ, $\tau \in \{1, 2, T\}$ such that $h_1 \to p_1$ and $h_2 \to p_2$. If it is not the case $\mathcal{S}$ returns $\perp$. If a handle for $(p_1 + p_2)$ already exists in T_τ $\mathcal{S}$ returns it. Otherwise, $\mathcal{S}$ generates a new handle $h \xleftarrow{\$} \{0,1\}^\lambda$, adds the mapping $h \to (p_1 + p_2)$ to T_τ and returns h.

Pairing oracle queries: Given $h_1, h_2 \in \{0,1\}^\lambda$, $\mathcal{S}$ verifies that formal polynomials p_1, p_2 exist in tables T_1 and T_2 respectively, such that $h_1 \to p_1$ in T_1 and $h_2 \to p_2$ in T_2. If it is not the case $\mathcal{S}$ returns $\perp$. If a handle for $(p_1 \cdot p_2)$ already exists in T_T $\mathcal{S}$ returns it. Otherwise, $\mathcal{S}$ generates a new handle $h \xleftarrow{\$} \{0,1\}^\lambda$, adds the mapping $h \to (p_1 \cdot p_2)$ to T_T and returns h.

Zero-testing oracle queries: Given $h \in \{0,1\}^\lambda$, $\mathcal{S}$ verifies that formal polynomials p exists in T_τ, $\tau \in \{1,2,T\}$, such that $h \to p$. If it is not the case $\mathcal{S}$ returns $\perp$. $\mathcal{S}$ then works as follows.

1. It "canonicalizes" the polynomial p by expressing it as a sum of products of formal variables in $\mathcal{T}$ with $\texttt{poly}(\lambda)$ terms.
2. If $\tau \in \{1,2\}$ and p is the zero polynomial, $\mathcal{S}$ outputs "zero". Otherwise if outputs "non-zero".
3. If $\tau = T$ the simulator decomposes p into the form

$$p = \sum_{i,j=1}^{Q} \hat{\alpha}^{(i)} \hat{\beta}^{(j)} \cdot \left(p_{i,j}\left(\{\hat{s}^{(i)}_{\ell,m},\ \hat{t}^{(j)}_{\ell,m}\}_{\ell\in[\sigma],m\in[N]} \right) + f_{i,j}\left(\{\hat{s}^{(i)}_{\ell,m},\ \hat{t}^{(j)}_{\ell,m}\}_{\ell\in[\sigma],m\in[N]} \right) \right) \tag{3}$$

   where for $1 \le i,j \le Q$, $p_{i,j}$ is defined as

$$p_{i,j} = c_{i,j} \cdot \left(\sum_{\ell,m=1}^{\sigma,N} \hat{s}^{(i)}_{\ell,m}\ \hat{t}^{(j)}_{\ell,m} \right)$$

   where $c_{i,j} \in \mathbb{Z}_q$ is the coefficient of the term $\hat{s}^{(i)}_{1,1}\hat{t}^{(j)}_{1,1}$, and $f_{i,j}$ consists of the remaining terms.
4. If for all $1 \le i,j \le Q$, $(i,j) = 0$ in $\mathcal{C}_{\mathsf{ip}}$ (corresponding to a zero inner product) and $f_{i,j}$ does not contain any non-zero term, $\mathcal{S}$ outputs "zero". Otherwise it outputs "non-zero".

Correctness of the simulator As in the original proof, the simulator's responses to token generation, encryption and group oracle queries are distributed identically as in the real experiment. We now have to show correctness of the simulator's answers to zero-testing oracle queries.

1. We first need to show that the canonicalization process in step 1 is efficient. Since the adversary can only obtain handles to new monomials using token generation and encryption queries, the monomials are all over formal variables in $\mathcal{R}$. Also, since the adversary can make Q queries at most, the polynomial p they can build and submit to the zero-testing oracle has at most $\texttt{poly}(Q)$ terms and degree 2.
   Then using Equations 1 and 2, the formal polynomial p can be expressed as a polynomial over formal variables in $\mathcal{T}$. Since p has degree at most 2 over variables in $\mathcal{R}$, it can be expressed as a sum of at most $\texttt{poly}(Q,n)$ monomials over variables in $\mathcal{T}$ and has degree at most $\texttt{poly}(n)$. Since both the polynomial over $\mathcal{R}$ and the canonical polynomial over $\mathcal{T}$ are polynomially-sized, the canonicalization process is efficient.
2. For $\tau = 1$, the only monomials the adversary can obtain are responses to token generation queries. Then the canonical polynomial is of the form

$$\begin{aligned} p &= \sum_{i=1}^{Q} \hat{\alpha}^{(i)} \left(\sum_{\ell,m=1}^{\sigma,N} c^{(i)}_{\ell,m} \cdot \hat{s}^{(i)}_{\ell,m} \right) \\ &= \sum_{i=1}^{Q} \hat{\alpha}^{(i)} \left(\sum_{\ell,m=1}^{\sigma,N} c^{(i)}_{\ell,m} \sum_{k=1}^{N} \hat{y}'^{(i)}_{\ell,k} \cdot \hat{b}_{\ell,k,m} \right) \\ &= \sum_{i=1}^{Q} \hat{\alpha}^{(i)} \left(\sum_{\ell,m=1}^{\sigma,N} c^{(i)}_{\ell,m} \left(\hat{b}_{\ell,1,m} + \sum_{k=2}^{N} \hat{y}^{(i)}_{\ell,k} \cdot \hat{b}_{\ell,k,m} \right) \right) \end{aligned}$$

   where $c^{(i)}_{1,1}, \cdots, c^{(i)}_{\sigma,N} \in \mathbb{Z}_q$.
   Notice that the sum $\hat{b}_{\ell,1,m} + \sum_{k=2}^{N} \hat{y}^{(i)}_{\ell,k} \cdot \hat{b}_{\ell,k,m}$ can never be the identically zero polynomial over the formal variables $\{\hat{b}_{\ell,k,m}\}_{\ell\in[\sigma],\ k,m\in[N]}$. This holds irrespective of the actual values of the adversary's query $x^{(i)}$. Since all $\{\hat{\alpha}^{(i)}\}_{i\in[Q]}$ and $\{\hat{b}_{\ell,k,m}\}_{\ell\in[\sigma],\ k,m\in[N]}$ are sampled uniformly and independently in the real game and the polynomial p has degree $\texttt{poly}(n) = \texttt{poly}(\lambda)$, then by the Schwartz-Zippel lemma [KLM+16, Lemma 2.9], p evaluates to non-zero with overwhelming probability. This implies that the simulator is correct with overwhelming probability.

3. For $\tau = 2$, the only monomials the adversary can obtain are responses to ciphertexts queries. Then the canonical polynomial is of the form

$$\begin{aligned} p &= \sum_{i=1}^{Q} \hat{\beta}^{(i)} \left(\sum_{\ell,m=1}^{\sigma,N} c_{\ell,m}^{(i)} \cdot \hat{t}_{\ell,m}^{(i)} \right) \\ &= \sum_{i=1}^{Q} \hat{\beta}^{(i)} \left(\sum_{\ell,m=1}^{\sigma,N} c_{\ell,m}^{(i)} \sum_{k=1}^{N} \hat{x}_{\ell,k}^{\prime(i)} \cdot \hat{b}_{\ell,k,m}^{*} \right) \\ &= \sum_{i=1}^{Q} \hat{\beta}^{(i)} \left(\sum_{\ell,m=1}^{\sigma,N} c_{\ell,m}^{(i)} \left(\zeta_{\ell}^{(i)} \cdot \hat{b}_{\ell,1,m}^{*} + \sum_{k=2}^{N} \hat{x}_{\ell,k}^{(i)} \cdot \hat{b}_{\ell,k,m}^{*} \right) \right) \end{aligned}$$

where $c_{1,1}^{(i)}, \cdots, c_{\sigma,N}^{(i)} \in \mathbb{Z}_q$. Notice that the sum $\zeta_{\ell}^{(i)} \cdot \hat{b}_{\ell,1,m}^{*} + \sum_{k=2}^{N} \hat{x}_{\ell,k}^{(i)} \cdot \hat{b}_{\ell,k,j}^{*}$ can only be the identically zero polynomial over the formal variables $\{\hat{b}_{\ell,k,m}^{*}\}_{\ell\in[\sigma],\ k,m\in[N]}$ if $\zeta_{\ell}^{(i)} = 0$ which happens with negligible probability. Again, this holds irrespective of the adversary's queries $y^{(1)}, \cdots, y^{(Q)}$ and p is not the identically zero polynomial over the formal variables $\{\hat{\beta}^{(i)}\}_{i\in[Q]}$ and $\{\hat{b}_{\ell,k,m}^{*}\}_{\ell\in[\sigma],\ k,m\in[N]}$. Since all $\hat{b}_{\ell,k,m}^{*}$ are independent from one another (since $\hat{b}_{\ell,k,m}$ was sampled uniformly and independently), then again by Schwartz-Zippel lemma p evaluates to non-zero with overwhelming probability and the simulator is correct with overwhelming probability.

4. For $\tau = T$, the only polynomials the adversary can obtain are products of two polynomials, one from each base group. Then the polynomial p can be decomposed into a sum of monomials that each contain $\alpha^{(i)}$ and $\beta^{(j)}$ for some $i, j \in [Q]$. Then $\mathcal{S}$ can regroup terms for each $i, j \in [Q]$ and obtain Equation 3.
Suppose $f_{i,j}$ does not contain any term, then p is of the form

$$\begin{aligned} p &= \sum_{i,j=1}^{Q} \hat{\alpha}^{(i)} \hat{\beta}^{(j)} \cdot c_{i,j} \cdot \left(\sum_{\ell,m=1}^{\sigma,N} \hat{s}_{\ell,m}^{(i)}\ \hat{t}_{\ell,m}^{(j)} \right) \\ &= \sum_{i,j=1}^{Q} \hat{\alpha}^{(i)} \hat{\beta}^{(j)} \cdot c_{i,j} \cdot \left(\sum_{\ell,m=1}^{\sigma,N} \left(\sum_{k=1}^{N} \hat{y}_{\ell,k}^{\prime(i)} \cdot \hat{b}_{\ell,k,m} \right) \cdot \left(\sum_{k=1}^{N} \hat{x}_{\ell,k}^{\prime(j)} \cdot \hat{b}_{\ell,k,m}^{*} \right) \right) \\ &= \sum_{i,j=1}^{Q} \hat{\alpha}^{(i)} \hat{\beta}^{(j)} \cdot c_{i,j} \cdot \left(\sum_{\ell=1}^{\sigma} (\hat{x}_{\ell}^{\prime(j)})^{T} \cdot \mathbb{B}_{\ell}^{*} \cdot \mathbb{B}_{\ell}^{T} \cdot \hat{y}_{\ell}^{\prime(i)} \right) \\ &= \sum_{i,j=1}^{Q} \hat{\alpha}^{(i)} \hat{\beta}^{(j)} \cdot c_{i,j} \cdot \left(\sum_{\ell=1}^{\sigma} \zeta_{\ell}^{(i)} + \langle \hat{x}_{\ell}^{(j)}, \hat{y}_{\ell}^{(i)} \rangle \right) \\ &= \sum_{i,j=1}^{Q} \hat{\alpha}^{(i)} \hat{\beta}^{(j)} \cdot c_{i,j} \cdot \langle \hat{x}^{(j)}, \hat{y}^{(i)} \rangle \end{aligned}$$

Then, it is obvious that p is the zero polynomial when all (i, j) inner products are zero, which can be known by checking if $(i, j) \to 0$ in $\mathcal{C}_{\mathsf{ip}}$.
Now suppose that for some $i, j \in [Q]$ the polynomial $f_{i,j}$ contains at least one term. Then we claim that $f_{i,j}$ cannot be the identically zero polynomial over the formal variables $\{\ \hat{b}_{\ell,k,m}\ \}_{\ell\in[\sigma],k,m\in[N]}$, irrespective of the adversary's choice of admissible queries. We refer the reader to the original work [KLM+16, Section 3] for a detailed proof of this claim. Then by the Schwartz-Zippel lemma, p evaluates to non-zero with overwhelming probability when $f_{i,j}$ contains at least one term.

□

6 Further Related Work

In this section we briefly review further related work separated into two categories: proximity searchable encryption schemes and leakage abuse attacks. Before starting this discussion we note that there are two primary general approaches to searchable encryption. *Property preserving encryption* retains compatibility with existing database indexing mechanisms by retaining a property of data, such as equality [BFOR08], but destroying other structure. The second is called *structured encryption* which creates a new indexing mechanism that requires server side changes. Property preserving encryption is subject to much stronger attacks than structured encryption (see [FVY+17]). We focus on structured encryption.

Approaches to Proximity Search Li et al. [LWW+10], Wang et al. [WMT+13] and Boldyreva and Chenette [BC14] reduced proximity search to keyword equality search. These works propose two complimentary approaches:

1. When adding a record x_i to a database, also inserts all close values as keywords, that is $\{x_j \mid \mathcal{D}(x_i, x_j) \leq t\}$ are added as keywords associated to x_i.
2. The second approach requires a searchable encryption scheme that supports disjunctive search. It inserts just x_i, but when searching for y it searches for the disjunction $\vee_{x_i|\mathcal{D}(x_i,y)\leq t}\ x_i$.

Either approach can be instantiated using a searchable encryption scheme that supports disjunction over keyword equality (inheriting any leakage). However, for biometrics, the number of keywords $\vee_{x_i|\mathcal{D}(x_i,y)\leq t}\{x_i\}$ usually grows exponentially in t. In existing disjunctive schemes, the size of the query grows with the size of the disjunction [FVY+17], making this approach only viable for constant values of t.

Kim et al. [KLM+18] use function-hiding inner product encryption (IPE) [BJK15] to store records on the server. Importantly, unlike our construction their scheme is designed to reveal the inner product between stored records x_i and a query y. This leads to the scheme leaking $\forall x_i,\ \mathcal{D}(x_i, y)$, the distance between the query and all records.

Kuzu et al.'s [KIK12] solution relies on *locality sensitive hashes* [IM98]. A locality sensitive hash ensures that close values have a higher probability to produce collisions than values that are far apart. For some value x_i, the client samples k locality sensitive hashes and computes $h_1(x_i), \cdots, h_k(x_i)$. The client then inserts $h_1(x_i), \cdots, h_k(x_i)$ as keywords for the record x_i in the database. When querying for value y the client computes the hashes of y, $h_1(y), \cdots, h_k(y)$. The server then returns every record for which at least one hash matches. Thus, a scheme can be built from any scheme supporting disjunctive keyword equality, inheriting any leakage. Since S learns the number of matching locality sensitive hashes for each record (which is expected to be more than 0), the number of matching locality sensitive hashes is a proxy for the distance between the query value and the records. More matching locality sensitive hashes implies smaller distance. This allows the server to establish the approximate distance between each record and the query, again enabling the server to establish geometry.

Zhou and Ren [ZR18] propose a variant of inner product encryption that reveals if the distance is less than t only. However, their security is based on $\mathbf{A}x_i$ and $y\mathbf{B}$ hiding x_i and y for secret square $\mathbf{A}$ and $\mathbf{B}$. Security is heuristic with no underlying assumption or proof of information theoretic security.

Leakage Abuse Attacks Searchable encryption achieves acceptable performance by *leaking* information to the server. See Kamara, Moataz, and Ohrimenko for an overview of leakage types in structured encryption [KMO18]. The key to attacks is combining leakage with auxiliary data, such as the frequency of values stored in the data set. Together these sources can prove catastrophic – allowing the attacker to run attacks to recover either the queries being made or the data stored in the database. We consider attacks that rely on injecting files or queries [ZKP16] to be out of scope. Common, attackable, relevant leakage profiles are:

1. *Response length leakage* [KKNO16, GLMP18] Often known as *volumetric leakage*, the attacker is given access to only the number of records returned for each query. Based on this information, attacks cross-correlate with auxiliary information about the dataset, and identify high frequency items in both the encrypted database and the auxiliary dataset.

2. *Query equality leakage* [WLD+17] the attacker is able to glean which queries are querying the same value, but not necessarily the value itself. Attacks on this profile rely on having information about the query distribution, and much like the response length leakage attacks, match with that auxiliary information based on frequency.

3. *Access pattern leakage* [IKK12, CGPR15] here the attacker is given knowledge if the same dataset element is returned for different queries. This allows the attacker to build a *co-occurrence* matrix, mapping what records are returned for pairs of queries. Based on the frequencies of the co-occurrence matrix for the encrypted dataset, and the co-occurrence matrix for the auxiliary dataset, the attack can identify records.

Recent attacks have targeted the geometry present in range search [GSB+17,LMP18,GLMP18,KPT20,FMC+20]. Building on the co-occurrence matrix (available with access patten leakage) consider the case when records a, b, c are returned by a first query and c, d are returned by a second query. One can immediately infer that the comparison relation between a and d is the same as the comparison relation between b and e. As more constraints of this type are collected one can collect an ordering of all records (up to reflection).

Recent work has shown how to attack nearest neighbor systems that reveal the k-closest values to a queried value [KPT19, MT19, KPT20]. Such information is available when the (approximate) distance is leaked between the query and all records.

7 Conclusion

Biometric databases range in scale and impact from having a billion plus records to hundreds of values. The Aadhaar system in India links a citizen's biometrics with a unique 12 digit number with over 1 billion numbers issued [Dau14]. The Tribune newspaper purchased full access to Aardhar database for $13 [Kha18]. Argentina, Kenya [Fou], and Gabon [SMHT+19] have national biometric databases. Smaller use case include criminal [Nak07] and immigration [Mer09] databases. The IAFIS system contains biometrics of 70,000 suspected terrorists [SK16]. Retailers keep facial recognition databases to record shoplifters [Tab18, Row20]. Biometrics have been proposed for identification of individuals at the clinic level for electronic health records or EHR [HTTK19]. Such settings usually have hundreds of individuals.

The construction introduced in this work requires a linear scan of the database and a high cost for each record to check if should be returned. As mentioned above, in our final scheme storage overhead is a multiplicative factor of $2\log(|\mathbb{G}|)$, query size is $2t\log(|\mathbb{G}|)$, and client storage is roughly $8n\log(|\mathbb{G}|)$ for a biometric of dimension n (see Table 3). However, query complexity is high, each query requires $O(tn)$ group operations per stored value. Thus, the scheme is not appropriate for deployment in large datasets. However, there are applications (access to EHR at a clinic) which have hundreds of users. It is not possible for an individual to use different biometrics in different applications. This means biometric compromises are universal so smaller databases that are likely to be outsourced deserve strong protections.

Our discussion of leakage in Section 4, used existing public biometric datasets that only have hundreds of unique biometrics (with hundreds of readings per biometric). In a million (or more) biometric application, the shape of the histograms (see Figure 4) will remain the same. However, having more records means a higher number of records in the tails of the distributions, sharpening the tradeoff between TAR and leakage. Thus, deployment to a million biometric application requires both efficiency improvements and more study on biometric statistics.

It is relatively simple to turn approximate distance into full database recovery. We believe it is critical to start from meaningful security and improve efficiency. We hope to learn from the strong claims made about property-preserving encryption and how little effective protection they provided (see Section 6). Biometric databases are infrequently access and frequently interactive applications (rather than enabling statistical applications), removing the need for response time in the milliseconds (see [FMC+15, Section 6.2]). For example, the average response time of the IAFIS system was 27 minutes [SK16]. Thus, our construction can provide strong protection on small scale biometric databases within existing performance constraints.

Acknowledgements The authors wish to thank Daniel Wichs for important preliminary discussion. In addition, the authors thank Sohaib Ahmad for running iris data analysis. The work of L.D. is supported by the Harriott Fellowship. C.C. is supported by NSF Award #1849904 and a fellowship from Synchrony Inc. The work of B.F. is supported by NSF Award #1849904 and ONR Grant N00014-19-1-2327. The work of A.H. is supported by NSF Award #1750795.

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via Contract No. 2019-19020700008. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is

authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[AEG+06] James Aspnes, Tolga Eren, David Kiyoshi Goldenberg, A Stephen Morse, Walter Whiteley, Yang Richard Yang, Brian DO Anderson, and Peter N Belhumeur. A theory of network localization. *IEEE Transactions on Mobile Computing*, 5(12):1663–1678, 2006.

[AF18] Sohaib Ahmad and Benjamin Fuller. Unconstrained iris segmentation using convolutional neural networks. In *Asian Conference on Computer Vision*, pages 450–466. Springer, 2018.

[AF19] Sohaib Ahmad and Benjamin Fuller. Thirdeye: Triplet-based iris recognition without normalization. In *IEEE International Conference on Biometrics: Theory, Applications and Systems*, 2019.

[AF20] Sohaib Ahmad and Benjamin Fuller. Resist: Reconstruction of irises from templates. In *2020 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–10. IEEE, 2020.

[Ahm20] Sohaib Ahmad. Sohaib ahmad github, 2020. Accessed: 2020-07-23.

[BC14] Alexandra Boldyreva and Nathan Chenette. Efficient fuzzy search on encrypted data. In *International Workshop on Fast Software Encryption*, pages 613–633. Springer, 2014.

[BCSW19] Manuel Barbosa, Dario Catalano, Azam Soleimanian, and Bogdan Warinschi. Efficient function-hiding functional encryption: From inner-products to orthogonality. In Mitsuru Matsui, editor, *Topics in Cryptology – CT-RSA 2019*, pages 127–148, Cham, 2019. Springer International Publishing.

[BF16] Kevin W Bowyer and Patrick J Flynn. The ND-IRIS-0405 iris image dataset. *arXiv preprint arXiv:1606.04853*, 2016.

[BFOR08] Mihir Bellare, Marc Fischlin, Adam O'Neill, and Thomas Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In *Advances in Cryptology – CRYPTO 2008*, pages 360–378, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[BHJP14] Christoph Bösch, Pieter Hartel, Willem Jonker, and Andreas Peter. A survey of provably secure searchable encryption. *ACM Computing Surveys (CSUR)*, 47(2):1–51, 2014.

[BJK15] Allison Bishop, Abhishek Jain, and Lucas Kowalczyk. Function-hiding inner product encryption. In *Advances in Cryptology – ASIACRYPT 2015*, pages 470–491, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[BPCR04] Ruud M Bolle, Sharath Pankanti, Jonathan H Connell, and Nalini K Ratha. Iris individuality: A partial iris model. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 2, pages 927–930. IEEE, 2004.

[CGKO11] Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: Improved definitions and efficient constructions. *Journal of Computer Security*, 19:895–934, 01 2011.

[CGPR15] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-abuse attacks against searchable encryption. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 668–679, 2015.

[Dau05] John Daugman. Results from 200 billion iris cross-comparisons. 01 2005.

[Dau09] John Daugman. How iris recognition works. In *The essential guide to image processing*, pages 715–739. Elsevier, 2009.

[Dau14] John Daugman. 600 million citizens of India are now enrolled with biometric id,". *SPIE newsroom*, 7, 2014.

[DRD+20] Pawel Drozdowski, Christian Rathgeb, Antitza Dantcheva, Naser Damer, and Christoph Busch. Demographic bias in biometrics: A survey on an emerging challenge. *IEEE Transactions on Technology and Society*, 1(2):89–103, 2020.

[EA11] Cem Evrendilek and Huseyin Akcan. On the complexity of trilateration with noisy range measurements. *IEEE Communications Letters*, 15(10):1097–1099, 2011.

[FMC+15] Benjamin Fuller, Darby Mitchell, Robert Cunningham, Uri Blumenthal, Patrick Cable, Ariel Hamlin, Lauren Milechin, Mark Rabe, Nabil Schear, Richard Shay, et al. Security and privacy assurance research (spar) pilot final report. Technical report, MIT Lincoln Laboratory Lexington United States, 2015.

[FMC+20] Francesca Falzon, Evangelia Anna Markatou, David Cash, Adam Rivkin, Jesse Stern, and Roberto Tamassia. Full database reconstruction in two dimensions. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 443–460, 2020.

[Fou] Electronic Frontier Foundation. Mandatory national ids and biometric databases.

[FVY+17] Benjamin Fuller, Mayank Varia, Arkady Yerukhimovich, Emily Shen, Ariel Hamlin, Vijay Gadepally, Richard Shay, John Darby Mitchell, and Robert K Cunningham. Sok: Cryptographically protected database search. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 172–191. IEEE, 2017.

[GKL+20] Paul Grubbs, Anurag Khandelwal, Marie-Sarah Lacharité, Lloyd Brown, Lucy Li, Rachit Agarwal, and Thomas Ristenpart. Pancake: Frequency smoothing for encrypted data stores. In *29th USENIX Security Symposium*, pages 2451–2468, 2020.

[GLMP18] Paul Grubbs, Marie-Sarah Lacharité, Brice Minaud, and Kenneth G Paterson. Pump up the volume: Practical database reconstruction from volume leakage on range queries. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 315–331, 2018.

[GRGB+12] Javier Galbally, Arun Ross, Marta Gomez-Barrero, Julian Fierrez, and Javier Ortega-Garcia. From the iriscode to the iris: A new vulnerability of iris recognition systems. *Black Hat Briefings USA*, 1, 2012.

[GSB+17] Paul Grubbs, Kevin Sekniqi, Vincent Bindschaedler, Muhammad Naveed, and Thomas Ristenpart. Leakage-abuse attacks against order-revealing encryption. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 655–672. IEEE, 2017.

[HBF08] Karen P Hollingsworth, Kevin W Bowyer, and Patrick J Flynn. The best bits in an iris code. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):964–973, 2008.

[HBF10] Karen Hollingsworth, Kevin W Bowyer, and Patrick J Flynn. Similarity of iris texture between identical twins. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pages 22–29. IEEE, 2010.

[HTTK19] Jigna J Hathaliya, Sudeep Tanwar, Sudhanshu Tyagi, and Neeraj Kumar. Securing electronics healthcare records in healthcare 4.0: a biometric-based approach. *Computers & Electrical Engineering*, 76:398–410, 2019.

[HWKL18] Yi Huang, Adams Kong Wai-Kin, and Kwok-Yan Lam. From the perspective of cnn to adversarial iris images. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–10. IEEE, 2018.

[IKK12] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. Access pattern disclosure on searchable encryption: ramification, attack and mitigation. In *Ndss*, volume 20, page 12. Citeseer, 2012.

[IM98] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, 1998.

[KE19] Evgenios M Kornaropoulos and Petros Efstathopoulos. The case of adversarial inputs for secure similarity approximation protocols. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 247–262. IEEE, 2019.

[Kha18] Rachna Khaira. Rs 500, 10 minutes, and you have access to billion aadhaar details. 2018.

[KIK12] Mehmet Kuzu, Mohammad Saiful Islam, and Murat Kantarcioglu. Efficient similarity search over encrypted data. In *2012 IEEE 28th International Conference on Data Engineering*, pages 1156–1167. IEEE, 2012.

[KKNO16] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Generic attacks on secure outsourced databases. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1329–1340, 2016.

[KLM+16] Sam Kim, Kevin Lewi, Avradip Mandal, Hart William Montgomery, Arnab Roy, and David J Wu. Function-hiding inner product encryption is practical. *IACR Cryptology ePrint Archive*, 2016:440, 2016.

[KLM+18] Sam Kim, Kevin Lewi, Avradip Mandal, Hart Montgomery, Arnab Roy, and David J Wu. Function-hiding inner product encryption is practical. In *International Conference on Security and Cryptography for Networks*, pages 544–562. Springer, 2018.

[KMO18] Seny Kamara, Tarik Moataz, and Olya Ohrimenko. Structured encryption and leakage suppression. pages 339–370, 2018.

[KP10] Ajay Kumar and Arun Passi. Comparison and combination of iris matchers for reliable personal authentication. *Pattern recognition*, 43(3):1016–1026, 2010.

[KPT19] Evgenios M Kornaropoulos, Charalampos Papamanthou, and Roberto Tamassia. Data recovery on encrypted databases with k-nearest neighbor query leakage. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1033–1050. IEEE, 2019.

[KPT20] Evgenios M Kornaropoulos, Charalampos Papamanthou, and Roberto Tamassia. The state of the uniform: attacks on encrypted databases beyond the uniform query distribution. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1223–1240. IEEE, 2020.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, pages 146–162, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[KT14] Yutaka Kawai and Katsuyuki Takashima. Predicate- and attribute-hiding inner product encryption in a public key setting. In Zhenfu Cao and Fangguo Zhang, editors, *Pairing-Based Cryptography – Pairing 2013*, pages 113–130, Cham, 2014. Springer International Publishing.

[Lai20] Lucas Laird. Metric dimension of hamming graphs and applications to computational biology. *arXiv preprint arXiv:2007.01337*, 2020.

[LMP18] M. Lacharité, B. Minaud, and K. G. Paterson. Improved reconstruction attacks on encrypted data using range query leakage. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 297–314, 2018.

[LTBL20] Lucas Laird, Richard C Tillquist, Stephen Becker, and Manuel E Lladser. Resolvability of hamming graphs. *SIAM Journal on Discrete Mathematics*, 34(4):2063–2081, 2020.

[LWW+10] Jin Li, Qian Wang, Cong Wang, Ning Cao, Kui Ren, and Wenjing Lou. Fuzzy keyword search over encrypted data in cloud computing. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–5. IEEE, 2010.

[MCYJ18] Guangcan Mai, Kai Cao, Pong C Yuen, and Anil K Jain. On the reconstruction of face images from deep face templates. *IEEE transactions on pattern analysis and machine intelligence*, 41(5):1188–1202, 2018.

[Mer09] Tony Mercer. Using biometrics to help secure uk borders. *Biometric Technology Today*, 17(5):7–8, 2009.

[MT19] Evangelia Anna Markatou and Roberto Tamassia. Full database reconstruction with access and search pattern leakage. In *International Conference on Information Security*, pages 25–43. Springer, 2019.

[Nak07] Ellen Nakashima. Fbi prepares vast database of biometrics. *Washington Post*, 22:2007, 2007.

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *Advances in Cryptology – CRYPTO 2010*, pages 191–208, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[OT12] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In *Advances in Cryptology – EUROCRYPT 2012*, pages 591–608, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[OT15] Tatsuaki Okamoto and Katsuyuki Takashima. Dual pairing vector spaces and their applications. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 98(1):3–15, 2015.

[PBDT05] Nissanka Bodhi Priyantha, Hari Balakrishnan, Erik D Demaine, and Seth Teller. Mobile-assisted localization in wireless sensor networks. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 1, pages 172–183. IEEE, 2005.

[PBF+08] P Jonathon Phillips, Kevin W Bowyer, Patrick J Flynn, Xiaomei Liu, and W Todd Scruggs. The iris challenge evaluation 2005. In *2008 IEEE Second International Conference on Biometrics: Theory, Applications and Systems*, pages 1–8. IEEE, 2008.

[PSO+09] P Jonathon Phillips, W Todd Scruggs, Alice J O'Toole, Patrick J Flynn, Kevin W Bowyer, Cathy L Schott, and Matthew Sharpe. FRVT 2006 and ICE 2006 large-scale experimental results. *IEEE transactions on pattern analysis and machine intelligence*, 32(5):831–846, 2009.

[RKV95] Nick Roussopoulos, Stephen Kelley, and Frédéric Vincent. Nearest neighbor queries. In *Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, pages 71–79, 1995.

[Row20] Elizabeth A Rowe. Regulating facial recognition technology in the private sector. *Stanford Technology Law Review*, 24(1), 2020.

[SDDN19] Sobhan Soleymani, Ali Dabouei, Jeremy Dawson, and Nasser M Nasrabadi. Adversarial examples to fool iris recognition systems. In *2019 International Conference on Biometrics (ICB)*, pages 1–8. IEEE, 2019.

[SK16] Monika Saini and A Kumar Kapoor. Biometrics in forensic identification: applications and challenges. *J Forensic Med*, 1(108):2, 2016.

[SMHT+19] Karine Sahli-Majira, Dominic S Haazen, Mahussi Hippolyte Togonou, Bahie Mary Rassekh, and Samuel Mills. *Gabon Civil Registration and Vital Statistics and Unique Identification Number Systems for Universal Health Coverage: A Case Study.* World Bank, 2019.

[SSF19] Sailesh Simhadri, James Steel, and Benjamin Fuller. Cryptographic authentication from the iris. In *International Conference on Information Security*, pages 465–485. Springer, 2019.

[SSW09] Emily Shen, Elaine Shi, and Brent Waters. Predicate privacy in encryption systems. In *Theory of Cryptography*, pages 457–473, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[SWP00] Dawn Xiaoding Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, pages 44–55. IEEE, 2000.

[Tab18] Nick Tabor. The secretive business of facial-recognition software in retail stores. 2018.

[TFL19] Richard C Tillquist, Rafael M Frongillo, and Manuel E Lladser. Metric dimension. *arXiv preprint arXiv:1910.04103*, 2019.

[VS11] Shreyas Venugopalan and Marios Savvides. How to generate spoofed irises from an iris code template. *IEEE Transactions on Information Forensics and Security*, 6(2):385–395, 2011.

[WLD+17] Guofeng Wang, Chuanyi Liu, Yingfei Dong, Hezhong Pan, Peiyi Han, and Binxing Fang. Query recovery attacks on searchable encryption based on partial knowledge. In *International Conference on Security and Privacy in Communication Systems*, pages 530–549. Springer, 2017.

[WMT+13] Jianfeng Wang, Hua Ma, Qiang Tang, Jin Li, Hui Zhu, Siqi Ma, and Xiaofeng Chen. Efficient verifiable fuzzy keyword search over encrypted data in cloud computing. *Comput. Sci. Inf. Syst.*, 10(2):667–684, 2013.

[ZD08] Sheikh Ziauddin and Matthew N Dailey. Iris recognition performance enhancement using weighted majority voting. In *2008 15th IEEE International Conference on Image Processing*, pages 277–280. IEEE, 2008.

[ZKP16] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. All your queries are belong to us: The power of file-injection attacks on searchable encryption. In *25th USENIX Security Symposium*, pages 707–720, 2016.

[ZR18] Kai Zhou and Jian Ren. Passbio: Privacy-preserving user-centric biometric authentication. *IEEE Transactions on Information Forensics and Security*, 13(12):3050–3063, 2018.

A Reducing Key Size of OT12 IPE scheme [OT12, Section 4]

To show the generality of our key reduction technique we apply it to a second IPE scheme of Okamoto and Takashima [OT12, Section 4]. We note that this scheme is a public key scheme that is adaptively attribute-hiding against chosen plaintext attacks under the (decisional linear) DLIN assumption. This corresponds to three changes to Definition 1:

1. The adversary no longer specifies pairs of functions, only a single value,
2. The adversary can adaptively query for values f_j receiving back tk_j,
3. There is only a single challenge plaintext $x^{(0)}, x^{(1)}$ because the adversary can encrypt values on either own.

Since this scheme is public key and is not function hiding it cannot be directly used to instantiate PSE. We use it as a second example of the applicability of the transform.

A.1 Additional notation and definitions

Let $\mathbb{F}_q$ denote a finite field of order q and $GL(n, \mathbb{F}_q)$ be the general linear group of degree n over $\mathbb{F}_q$. Let the vectors $\vec{e}_i$ be defined as $\vec{e}_i = (0^{i-1}, 1, 0^{n-i})$ for $1 \leq i \leq n$. Let $\mathbb{V}$ be a vector space, to differentiate its elements from other values we will use bold letters. Let $\boldsymbol{b}_i \in \mathbb{V}$, $1 \leq i \leq n$, then we denote the subspace generated by these vectors as $\mathsf{span}(\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n) \subseteq \mathbb{V}$. Consider the bases $\mathbb{B} = (\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n)$ and $\mathbb{B}^* = (\boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_n^*)$, and the vectors $\vec{x}$ and $\vec{v}$ then $(\vec{x})_{\mathbb{B}} = \sum_{i=1}^{n} x_i \boldsymbol{b}_i$ and $(\vec{v})_{\mathbb{B}^*} = \sum_{i=1}^{n} v_i \boldsymbol{b}_i^*$. Note that we will consider bases over both $\mathbb{F}_q$ and $\mathbb{G}_q$.

Definition 8 (Symmetric Bilinear Group)**.** *Suppose $\mathbb{G}, \mathbb{G}_T$ are an additive and multiplicative groups (respectively) of prime order q with generators $g \in \mathbb{G}$, and $g_T \in \mathbb{G}_T$ respectively. The group $\mathbb{G}$ uses additive notation, and the group $\mathbb{G}_T$ uses multiplicative notation. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a non-degenerate (i.e. $e(g, g) \neq 1$) bilinear pairing operation such that for all $x, y \in \mathbb{Z}_q$, $e(x(g), y(g)) = e(g, g)^{xy}$. Assume the group operations in $\mathbb{G}, \mathbb{G}_T$ and the pairing operation e are efficiently computable, then $(\mathbb{G}, \mathbb{G}_T, g, e)$ defines a symmetric bilinear group. Let $\mathcal{G}_{bpg}$ be an algorithm that takes input 1^λ and outputs a description of bilinear pairing groups $(q, \mathbb{G}, \mathbb{G}_T, g, e)$ with security parameter λ.*

We use the symmetric version of dual pairing vector spaces [OT15] where the pairing is based on symmetric bilinear groups defined in Definition 8.

Definition 9 (Dual Pairing Vector Spaces)**.** *Let $(q, \mathbb{G}, \mathbb{G}_T, g, e_{bg})$ be the symmetric bilinear pairing groups, then* Dual Pairing Vector Spaces *(DVPS) is a tuple of prime q, N-dimensional vector space $\mathbb{V} = \overbrace{\mathbb{G} \times \ldots \times \mathbb{G}}^{N}$ over $\mathbb{F}_q$, cyclic group $\mathbb{G}_T$ of order q, canonical basis $\mathbb{A}$ defined as:*

$$\mathbb{A} := (\vec{a}_1, \ldots, \vec{a}_n), \quad \vec{a}_i := \left(0^{i-1}, g, 0^{N-i}\right)$$

and pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The pairing e is defined with respect to e_{bg} from the symmetric bilinear pairing group $e(\vec{x}, \vec{y}) = \prod_{i=1}^{N} e_{bg}(g_i, h_i) \in \mathbb{G}_T$ where $\vec{x} = (g_1, \ldots, g_N) \in \mathbb{V}$ and $\vec{y} = (h_1, \ldots, h_N) \in \mathbb{V}$. This pairing is nondegenerate bilinear, i.e. $e(s\vec{x}, t\vec{y}) = e(\vec{x}, \vec{y})^{st}$ and if $e(\vec{x}, \vec{y}) = 1$ for all $\vec{y} \in \mathbb{V}$ then $\vec{x} = 0^N$. For all i and j, $e(\vec{a}_i, \vec{a}_j) = e(G, G)^{\delta_{i,j}}$ where $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise, and $e(g, g) \neq 1 \in \mathbb{G}_T$.

DPVS also has a linear transformation ("canonical maps") $\phi_{i,j}$ on $\mathbb{V}$ such that $\phi_{i,j}(\vec{a}_j) = \vec{a}_i$ and $\phi_{i,j}(\vec{a}_k) = 0$ if $k \neq j$. We define $\phi_{i,j}(\vec{x}) := \left(0^{i-1}, g_j, 0^{N-i}\right)$ where $\vec{x} = (g_1, \ldots, g_N)$. We then define the dual-pairing vector space generator *as $\mathcal{G}_{dpvs}$ which takes input 1^λ ($\lambda \in \mathbb{N}$) and $N \in \mathbb{N}$:*

1. *Runs $(q, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow \mathcal{G}_{bpg}\left(1^\lambda\right)$,*
2. *Compute $\mathbb{A}, \mathbb{V}$,*
3. *Returning $(q, \mathbb{G}, \mathbb{G}_T, g, e, \mathbb{V}, \mathbb{A})$.*

Lemma 1. *Let $(q, \mathbb{G}, \mathbb{G}_T, g, e, \mathbb{V}, \mathbb{A}) \leftarrow \mathcal{G}_{dpvs}$ be a (DPVS) generator as described above. We can efficiently sample a random linear transformation W by sampling random coefficients $\{r_{i,j}\}_{i,j=1,\cdots,n} \xleftarrow{\$} GL(n, \mathbb{F}_q)$ and setting*

$$W := \sum_{i,j=1}^{n,n} r_{i,j}\phi_{i,j}.$$

The matrix $R := (r_{i,j})$ and $R^* := ((r_{i,j})^{-1})^T$ then defines the adjoint action on $\mathbb{V}$ and we can define $(W^{-1})^T$ as

$$(W^{-1})^T := \sum_{i,j=1}^{N,N} r^*_{i,j}\phi_{i,j}$$

such that for any $x, y \in \mathbb{V}$, we have

$$e(W(x), (W^{-1})^T(y)) = e(x, y).$$

Assumption 1 (Decisional Linear Assumption)**.** *Let $\lambda \in N$ and $\beta \in \{0, 1\}$. We define a generator for the* Decisional Linear Assumption *(DLIN) problem, $\mathcal{G}_\beta^{DLIN}$, which on input 1^λ:*

1. *Samples $\mathsf{param}_{\mathbb{G}} = (q, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow \mathbb{G}_{bpg}(1^\lambda)$.*
2. *Samples $\kappa, \delta, \xi, \sigma \xleftarrow{\$} \mathbb{F}_q$.*
3. *Sets $Y^{(0)} = (\delta + \sigma)g$ and $Y^{(1)} \xleftarrow{\$} \mathbb{G}$.*
4. *Returns $(\mathsf{param}_{\mathbb{G}}, g, \xi g, \kappa g, \delta\xi g, \sigma\kappa g, Y^{(\beta)})$.*

The DLIN problem then consists in guessing β given $(\mathsf{param}_{\mathbb{G}}, g, \xi g, \kappa g, \delta\xi g, \sigma\kappa g, Y^{(\beta)}) \leftarrow \mathcal{G}_\beta^{DLIN}(1^\lambda)$. The decisional linear assumption *is that for any PPT distinguisher $\mathcal{D}$ for the DLIN problem the advantage is:*

$$\mathsf{Adv}_{\mathcal{D}}^{DLIN}(\lambda) = \left| \Pr[\mathcal{D}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_0^{DLIN}(1^\lambda)] - \Pr[\mathcal{D}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_1^{DLIN}(1^\lambda)] \right| = \mathsf{negl}(\lambda)$$

$\underline{\mathsf{Setup}(1^\lambda, n, \alpha)}$:

1. Sample $(\mathsf{param}_{\mathbb{V}}, \mathbb{B}, \mathbb{B}^*) \leftarrow \mathcal{G}_{\mathsf{ob}}^{\mathsf{IPE}}(1^\lambda, N)$,
2. For $1 \leq \ell \leq \alpha$, set $\hat{\mathbb{B}}_\ell = (\boldsymbol{b}_{\ell,0}, \cdots, \boldsymbol{b}_{\ell,n/\alpha}, \boldsymbol{b}_{\ell,N-1})$ and $\hat{\mathbb{B}}^*_\ell = (\boldsymbol{b}^*_{\ell,0}, \cdots, \boldsymbol{b}^*_{\ell,n/\alpha}, \boldsymbol{b}^*_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{b}^*_{\ell,N-1})$.
3. $\mathsf{pk} = (1^\lambda, \mathsf{param}_{\mathbb{V}}, \{\hat{\mathbb{B}}_\ell\}_{\ell=1,\cdots,\alpha})$ and $\mathsf{sk} = \{\hat{\mathbb{B}}^*_\ell\}_{\ell=1,\cdots,\alpha}$.

$\underline{\mathsf{TokGen}(\mathsf{pk}, \mathsf{sk}, \vec{v})}$:

1. Sample $\sigma \leftarrow \mathbb{F}_q$
2. Divide $\vec{v}$ in α smaller vectors of length n/α, such that $\vec{v} = (\vec{v}_1, \cdots, \vec{v}_\alpha)$.
3. For $1 \leq \ell \leq \alpha$, sample $\vec{\eta}_\ell \xleftarrow{\$} \mathbb{F}_q^{n/\alpha}$ and set $\boldsymbol{k}_\ell := (\overbrace{1}^{1}, \overbrace{\sigma\vec{v}_\ell}^{n/\alpha}, \overbrace{0, \cdots, 0}^{2n/\alpha}, \overbrace{\vec{\eta}_\ell}^{n/\alpha}, \overbrace{0}^{1})_{\mathbb{B}^*_\ell}$
4. $\mathsf{tk}_{\vec{v}} := (\boldsymbol{k}_1, \ldots, \boldsymbol{k}_\alpha)$

$\underline{\mathsf{Encrypt}(\mathsf{pk}, m, \vec{x})}$:

1. Sample $\omega \leftarrow \mathbb{F}_q$
2. Divide $\vec{x}$ in α smaller vectors of length n/α, such that $\vec{x} = (\vec{x}_1, \cdots, \vec{x}_\alpha)$.
3. For $1 \leq \ell \leq \alpha$, sample $\zeta_\ell, \varphi_\ell \xleftarrow{\$} \mathbb{F}_q$,
4. Set

$$\boldsymbol{c}_\ell = (\overbrace{\zeta_\ell}^{1}, \overbrace{\omega\vec{x}_\ell}^{n/\alpha}, \overbrace{0, \cdots, 0}^{3n/\alpha}, \overbrace{\varphi_\ell}^{1})_{\mathbb{B}_\ell}$$
$$\boldsymbol{c}_0 = m \cdot g_{\mathsf{T}}^{\left(\sum_{\ell=1}^{\alpha} \zeta_\ell\right)}$$

5. Return $\mathsf{ct}_{\vec{x}} := (c_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_\alpha)$

$\underline{\mathsf{Decrypt}(\mathsf{pk}, \mathsf{ct}_{\vec{x}}, \mathsf{sk}_{\vec{v}}) :}$
Return $m' = \prod_{\ell=1}^{\alpha} e(\boldsymbol{c}_\ell, \boldsymbol{k}_\ell) \ / \ c_0$

Figure 7: Description of modified IPE algorithms.

A.2 Construction

This construction is an adaptation of Okamoto and Takashima's IPE scheme [OT12, Section 4] (setting $\alpha = 1$ in Figure 7 yields the original scheme). As in the original construction, we first need to describe a random dual orthonormal bases generator, $\mathcal{G}_{\mathsf{ob}}^{\mathsf{IPE}^*}$, which will be called in the main construction's Setup algorithm to generate the master keys. This is different from the previous generator as it generates α sets of bases.

Construction 2 (Dual Orthonormal Bases Generator)**.** *Let $\mathcal{G}_{dpvs}$ be a symmetric dual-pairing vector space generator as described in Definition 9. Let $\lambda, N, \alpha \in \mathbb{N}$, where λ is the security parameter, N is the dimension of the vector space and α is the number of dual orthonormal bases pairs to generate. Then on inputs 1^λ, N and α, the orthonormal bases generator $\mathcal{G}_{ob}^{IPE^*}$ works as follows:*

1. *Sample $(q, \mathbb{G}, \mathbb{G}_T, g, e, \mathbb{V}, \mathbb{A}) \leftarrow \mathcal{G}_{dpvs}(1^\lambda, N)$.*
2. *Sample a non-zero element of the field, $\psi \xleftarrow{\$} \mathbb{F}_q^\times$.*
3. *Set $g_T = e(G, G)^\psi$ and $\mathsf{param}_{\mathbb{V}} = (q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e, g_T)$.*
4. *For each basis index $1 \leq \ell \leq \alpha$:*
   (a) *Sample a random map, as described in Lemma 1, $X_\ell = (\chi_{\ell,i,j}) \xleftarrow{\$} GL(N, \mathbb{F}_q)$ and set $(\vartheta_{\ell,i,j}) = \psi \cdot (X_\ell^T)^{-1}$, where $1 \leq i, j \leq N$.*
   (b) *For $1 \leq i \leq N$, set $\boldsymbol{b}_{\ell,i} = \sum_{j=1}^N \chi_{\ell,i,j} \cdot \boldsymbol{a}_j$ and $\boldsymbol{b}^*_{\ell,i} = \sum_{j=1}^N \vartheta_{\ell,i,j} \cdot \boldsymbol{a}_j$, where $(\boldsymbol{a}_1, \cdots, \boldsymbol{a}_N) = \mathbb{A}$.*
   (c) *Set $\mathbb{B}_\ell = (\boldsymbol{b}_{\ell,1}, \cdots, \boldsymbol{b}_{\ell,N})$ and $\mathbb{B}^*_\ell = (\boldsymbol{b}^*_{\ell,1}, \cdots, \boldsymbol{b}^*_{\ell,N})$.*
5. *Return $(\mathsf{param}_{\mathbb{V}}, \{\mathbb{B}_\ell, \mathbb{B}^*_\ell\}_{\ell=1,\cdots,\alpha})$.*

In this construction $\vec{x}$ will always denote the attribute, and $\vec{v}$ will denote the predicate. As in the original scheme, we assume that the first element of $\vec{x}$ is nonzero. Furthermore, note above we've used inner product encryption with no associated plaintext, here we include the value m which can be decrypted if the inner product is 0 and is hidden otherwise.

Component	Number of Group Elements
Secret Key	$8n^2/\alpha + 8n + 2\alpha$
Public Key	$4n^2/\alpha + 10n + 4\alpha$
Ciphertext	$4n + 2\alpha$
Token	$4n + 2\alpha$

Table 4: Sizes in Group Elements of Each Component of Revised Scheme. The value α is how many separate bases are used. Considering $\alpha = 1$ gives sizes for the original scheme of Okamoto and Takashima. Setting $\alpha = \Omega(n)$ makes all components a linear number of group elements.

Construction 3. *Let $\lambda \in \mathbb{N}$ be the security parameter and $n, \alpha \in \mathbb{N}$ such that $n/\alpha \in \mathbb{N}$ and define $N = 4n/\alpha + 2$. Let $\vec{x}, \vec{v} \in \mathbb{F}_q^n \setminus \{\vec{0}\}$ and such that the first element of $\vec{x}$ is nonzero. Define the algorithms as in Figure 7.*

Correctness If the inner product of our attribute vector and our predicate vector is zero (in each basis), $\langle \vec{x}, \vec{v} \rangle = \sum_{\ell=1}^{\alpha} \langle \vec{x}_\ell, \vec{v}_\ell \rangle = 0$, then by the properties of our group structures we cancel terms,

$$\prod_{\ell=1}^{\alpha} e(\boldsymbol{c}_\ell, \boldsymbol{k}_\ell) = g_T^{\left(\sum_{\ell=1}^{\alpha} \zeta_\ell + \omega\sigma\langle \vec{x}_\ell, \vec{v}_\ell \rangle\right)} = g_T^{\left(\sum_{\ell=1}^{\alpha} \zeta_\ell\right)},$$

and finally conclude $m' = m$, therefore our construction is correct when the inner product is zero.

Key Reduction The key reduction is summarized in Table 4. In the Okamoto and Takashima scheme the DPVSs are over vectors of dimension $4n + 2$ with the public key being $n + 2$ basis vectors and the secret key being $2n + 1$. Ciphertexts and tokens are a single vector. By splitting into α bases we introduce an α overhead on each object while reducing the dimension to $4n/\alpha + 2$ and also reducing the number of basis vectors released in the public and secret key to $2n/\alpha + 1$ and $n/\alpha + 2$ respectively.

Security The proposed IPE scheme achieves the same security as the original construction [OT12, Theorem 1].

Theorem 3. *The IPE construction in Figure 7 with $\alpha = 1$ is adaptively attribute-hiding against chosen plaintext attacks under the DLIN assumption, such that for any PPT adversary $\mathcal{A}$ there exists PPT distinguishers $\mathcal{D}_{0\text{-}1}, \mathcal{D}_{1\text{-}1}, \mathcal{D}_{0\text{-}2\text{-}h}, \mathcal{D}_{1\text{-}2\text{-}h\text{-}1}, \mathcal{D}_{1\text{-}2\text{-}h\text{-}2}$ such that for any security parameter $\lambda \in \mathbb{N}$*

$$\mathsf{Adv}^{\mathsf{IPE}}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}_{0\text{-}1}}(\lambda) + \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}_{1\text{-}1}}(\lambda) + \sum_{h=1}^{\nu} \left(\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}_{0\text{-}2\text{-}h}}(\lambda) + \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}_{1\text{-}2\text{-}h\text{-}1}}(\lambda) + \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}_{1\text{-}2\text{-}h\text{-}2}}(\lambda) \right) + \frac{28\nu + 11}{q}$$

where $\nu \in \mathbb{N}$ is the maximum number of key queries $\mathcal{A}$ can make.[8]

This proof (like the proofs we build from) involve a system of games where each game changes a single element of a vector and is shown to be indistinguishable from the last game. These indistinguishability statements are made from a system of problems that stem from the decision linear assumption. We modify the original problems of Okamoto and Takashima [OT12] to include multiple bases of the DPVS. We can maintain security while spreading material across bases, because the public portions are incomplete and the bases are sampled independently, making it difficult to create meaningful relationships between bases. Using the same structure for our system of games and problems (but now including security with multiple bases) we show that our scheme matches the security of Okamoto and Takashima [OT12].

Proof of Theorem 3. For this theorem's proof we refer the reader to Okamoto and Takashima's proof of Theorem 1 [OT12, Section 4.3.1]. Notice that in this version Games $0', 1, 2\text{-}h\text{-}1, \cdots, 2\text{-}h\text{-}4, 3$ are replaced by Games $0^*, 1^*, 2\text{-}h\text{-}1^*, \cdots, 2\text{-}h\text{-}4^*, 3^*$ and the dimension of the hidden subspaces is $2n/\alpha$ instead of $2n$. □

[8] In the original paper the constant was $(29\nu + 17)/q$ instead of $(28\nu + 11)/q$ but the proof still holds despite this small difference.

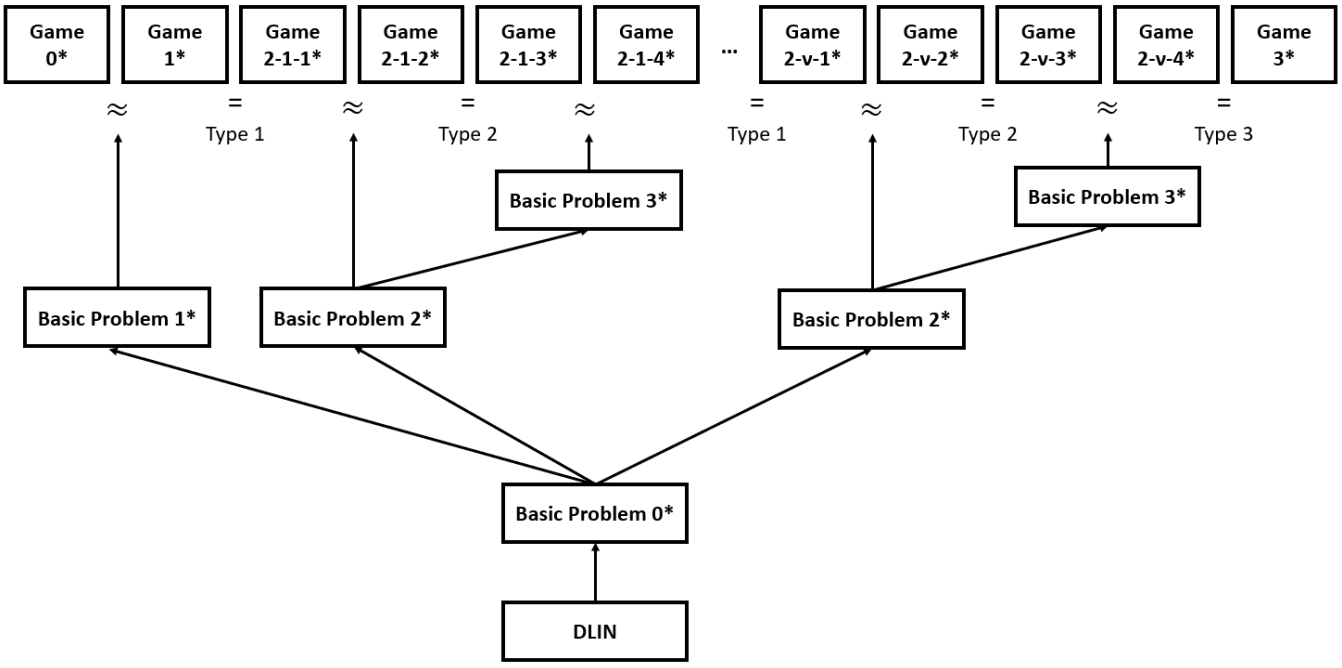


Figure 8: Structure of reductions.

Lemma 2. *For any PPT adversary $\mathcal{A}$ there exists PPT distinguishers $\mathcal{D}_1, \mathcal{D}_{2\text{-}h\text{-}1}, \mathcal{D}_{2\text{-}h\text{-}2}$ such that for any security parameter $\lambda \in \mathbb{N}$ in Game 0*,*

$$\Pr[\mathcal{A} \text{ wins} \mid t = 1] - \frac{1}{2} \leq \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}_1}(\lambda) + \sum_{h=1}^{\nu} \left(\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}_{2\text{-}h\text{-}1}}(\lambda) + \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}_{2\text{-}h\text{-}2}}(\lambda) \right) + \frac{22\nu + 6}{q}$$

where $\nu \in \mathbb{N}$ is the maximum number of key queries $\mathcal{A}$ can make.[9]

Proof of Lemma 2. For a detailed high level overview of the proof, we refer the reader to Okamoto and Takashima's work [OT12, Section 4.3.2]. The games and the problems described in their proofs had to be updated to fit our new construction, but as in the original work, the goal is to show that indistinguishably of the games reduces to the DLIN assumption through a hierarchy of Problems. In the rest of this proof, we will describe the updated version of the needed games and problems. The tree of the reductions, from the games to the DLIN assumption, can be found in Figure 8.

We define the following $4\nu + 3$ updated games. In each game we will only describe the component that changed compared to the previous game (either the keys or the ciphertexts). The boxed parts in keys and ciphertexts indicate parts that have changed compared to the previous game.

Game 0* : This game is the same as the game described in the original proof [OT12, Definition 5] except that before the setup phase the bit $t \xleftarrow{\$} \{0,1\}$ is sampled and the game is aborted when $t \neq s$, where $s = 1$ when $m^{(0)} = m^{(1)}$ and $s = 0$ otherwise. For this proof we only consider the case where $t = 1$ thus $m^{(0)} = m^{(1)}$ and c_0 is independent from β. The keys and ciphertexts are built as in our construction. The answer to a key query for some vector $\vec{v} = (\vec{v}_1, \cdots, \vec{v}_\alpha)$ is

$$\boldsymbol{k}_\ell = (1, \sigma\vec{v}_\ell, 0^{n/\alpha}, 0^{n/\alpha}, \vec{\eta}_\ell, 0)_{\mathbb{B}^*_\ell}$$

where $1 \leq \ell \leq \alpha$, $\sigma \xleftarrow{\$} \mathbb{F}_q$ and $\vec{\eta}_\ell \xleftarrow{\$} \mathbb{F}_q^{n/\alpha}$. The challenge ciphertexts for attribute $\vec{x}^{(\beta)} = (\vec{x}_1^{(\beta)}, \cdots, \vec{x}_\alpha^{(\beta)})$ and message $m^{(\beta)}$ is

$$\boldsymbol{c}_\ell = (\zeta_\ell, \omega\vec{x}_\ell^{(\beta)}, 0^{n/\alpha}, 0^{n/\alpha}, 0^{n/\alpha}, \varphi_\ell)_{\mathbb{B}_\ell}$$

[9] In the original paper the constant was $(23\nu + 12)/q$ instead of $(22\nu + 6)/q$ but the proof still holds despite this small difference.

and

$$c_0 = m^{(\beta)} g_{\mathsf{T}}^{\left(\sum_{\ell=1}^{\alpha} \zeta_\ell\right)}$$

where $1 \leq \ell \leq \alpha$, $\beta \xleftarrow{\$} \{0,1\}$ and $\omega, \zeta_\ell, \varphi_\ell \xleftarrow{\$} \mathbb{F}_q$.

Game 1* : This game is the same as Game 0* except that the challenge ciphertexts are now

$$\boldsymbol{c}_\ell = (\zeta_\ell, \omega \vec{x}_\ell^{\,(\beta)}, \boxed{z x_{\ell,1}^{(\beta)}}, \boxed{0^{(n/\alpha)\text{-}1}}, 0^{n/\alpha}, 0^{n/\alpha}, \varphi_\ell)_{\mathbb{B}_\ell}$$

where $x_{\ell,1}^{(\beta)} \neq 0$ is the first coordinate of $\vec{x}_\ell^{\,(\beta)}$, $z \xleftarrow{\$} \mathbb{F}_q$ and all other values are generated as in Game 0*.

Game 2-h-1* : For $1 \leq h \leq \nu$, each game is the same as Game 2-(h-1)-4* (here Game 2-0-4* is Game 1*), except that the challenge ciphertexts are now

$$\boldsymbol{c}_\ell = (\zeta_\ell, \omega \vec{x}_\ell^{\,(\beta)}, \boxed{\omega' \vec{x}_\ell^{\,(\beta)}}, \boxed{\omega_0'' \vec{x}_\ell^{\,(0)} + \omega_1'' \vec{x}_\ell^{\,(1)}}, 0^{n/\alpha}, \varphi_\ell)_{\mathbb{B}_\ell}$$

where $\omega', \omega_0'', \omega_1'' \xleftarrow{\$} \mathbb{F}_q$ and all other values are generated as Game 2-(h-1)-4*.

Game 2-h-2* : For $1 \leq h \leq \nu$, each game is the same as Game 2-h-1*, except that the h^{th} key query for $\vec{v}$ is now

$$\boldsymbol{k}_\ell = (1, \sigma \vec{v}_\ell, \boxed{\sigma' \vec{v}_\ell}, 0^{n/\alpha}, \vec{\eta}_\ell, 0)_{\mathbb{B}_\ell^*}$$

where $\sigma' \xleftarrow{\$} \mathbb{F}_q$ and all other values are generated as in Game 2-h-1*.

Game 2-h-3* : For $1 \leq h \leq \nu$, each game is the same as Game 2-h-2*, except that the challenge ciphertexts are now

$$\boldsymbol{c}_\ell = (\zeta_\ell, \omega \vec{x}_\ell^{\,(\beta)}, \boxed{\omega_0' \vec{x}_\ell^{\,(0)} + \omega_1' \vec{x}_\ell^{\,(1)}}, \omega_0'' \vec{x}_\ell^{\,(0)} + \omega_1'' \vec{x}_\ell^{\,(1)}, 0^{n/\alpha}, \varphi_\ell)_{\mathbb{B}_\ell}$$

where $\omega_0', \omega_1' \xleftarrow{\$} \mathbb{F}_q$ and all other values are generated as Game 2-h-2*.

Game 2-h-4* : For $1 \leq h \leq \nu$, each game is the same as Game 2-h-3*, except that the h^{th} key query for $\vec{v}$ is now

$$\boldsymbol{k}_\ell = (1, \sigma \vec{v}_\ell, \boxed{0^{n/\alpha}, \sigma'' \vec{v}_\ell}, \vec{\eta}_\ell, 0)_{\mathbb{B}_\ell^*}$$

where $\sigma'' \xleftarrow{\$} \mathbb{F}_q$ and all other values are generated as in Game 2-h-3*.

Game 3* : The game is the same as Game 2-ν-2*, except that the challenge ciphertexts are now

$$\boldsymbol{c}_\ell = \left(\zeta_\ell, \boxed{\omega_0 \vec{x}_\ell^{\,(0)} + \omega_1 \vec{x}_\ell^{\,(1)}}, \omega_0' \vec{x}_\ell^{\,(0)} + \omega_1' \vec{x}_\ell^{\,(1)}, \omega_0'' \vec{x}_\ell^{\,(0)} + \omega_1'' \vec{x}_\ell^{\,(1)}, 0^{n/\alpha}, \varphi_\ell\right)_{\mathbb{B}_\ell}$$

where $\omega_0, \omega_1 \xleftarrow{\$} \mathbb{F}_q$ and all other values are generated as Game 2-h-2*. Notice that with this modification, $\boldsymbol{c}_\ell$ becomes independent from the bit $\beta \xleftarrow{\$} \{0,1\}$.

Let $t = 1$, we define the advantage of a PPT machine $\mathcal{A}$ in Game g^* as $\mathsf{Adv}_{\mathcal{A}}^{(g*)}(\lambda)$, where $g = 0, 1, 2\text{-}h\text{-}1, \cdots, 2\text{-}h\text{-}4, 3$. In the following proofs, we will calculate the difference of advantages for each pair of neighboring games. As in the original proof [OT12, Section 4.3.2] we then obtain

$$\begin{aligned}
\mid \mathsf{Adv}_{\mathcal{A}}^{(0*)}(\lambda) \mid &\leq \mid \mathsf{Adv}_{\mathcal{A}}^{(0*)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(1*)}(\lambda) \mid + \mid \mathsf{Adv}_{\mathcal{A}}^{(2\text{-}\nu\text{-}4*)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(3*)}(\lambda) \mid + \mathsf{Adv}_{\mathcal{A}}^{(3*)}(\lambda) \\
&+ \sum_{h=1}^{\nu} \Big(\mid \mathsf{Adv}_{\mathcal{A}}^{(2\text{-}h\text{-}4*)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2\text{-}h\text{-}1*)}(\lambda) \mid + \sum_{i=2}^{4} \mid \mathcal{A}^{(2\text{-}h\text{-}(i-1)*)}(\lambda) - \mathcal{A}^{(2\text{-}h\text{-}i*)}(\lambda) \mid \Big) \\
&\leq \mathsf{Adv}_{\mathcal{D}_1}^{\mathsf{bp1}^*}(\lambda) + \sum_{h=1}^{\nu} \Big(\mathsf{Adv}_{\mathcal{D}_{2\text{-}h\text{-}1}}^{\mathsf{bp2}^*}(\lambda) + \mathsf{Adv}_{\mathcal{D}_{2\text{-}h\text{-}1}}^{\mathsf{bp3}^*}(\lambda) \Big) + \frac{10\nu + 1}{q} \\
&\leq \mathsf{Adv}_{\mathcal{D}_1}^{\mathsf{DLIN}}(\lambda) + \sum_{h=1}^{\nu} \Big(\mathsf{Adv}_{\mathcal{D}_{2\text{-}h\text{-}1}}^{\mathsf{DLIN}}(\lambda) + \mathsf{Adv}_{\mathcal{D}_{2\text{-}h\text{-}2}}^{\mathsf{DLIN}}(\lambda) \Big) + \frac{22\nu + 6}{q}
\end{aligned}$$

In the above, bounds on $\mathsf{Adv}_{\mathcal{D}_1}^{\mathsf{bp1}^*}(\lambda), \mathsf{Adv}_{\mathcal{D}_{2\text{-}h\text{-}1}}^{\mathsf{bp2}^*}(\lambda)$ and $\mathsf{Adv}_{\mathcal{D}_{2\text{-}h\text{-}1}}^{\mathsf{bp3}^*}(\lambda)$ are described in Lemmas 4, 5 and 6 respectively. This hybrid proof relies on both computational and information theoretical problems. The computational problems are the following:

Basic problem 0* embeds a DLIN instance in the smallest and simplest dual pairing vector space possible. The resulting orthonormal bases are 3x3 matrices and are built using the random elements ξ and κ from the DLIN instance. The game is then to distinguish between a vector in which the middle element is zero and a vector in which the middle element is random.

Basic problem 1* consists in distinguishing between two challenge ciphertexts. One where the third slot contains zeros, as in the actual construction, and the second where the third slot contains a randomized copy of the second slot (i.e. the vector x).

Basic problem 2* consists in distinguishing between two challenge keys. One where the third slot contains zeros, as in the actual construction, and the second where the third slot contains a randomized copy of the second slot (i.e. the vector v).

Basic problem 3* consists in distinguishing between two challenge keys. One where the randomized vector is in the third slot and the other where it is in the fourth slot. The second slot being all zeros in both cases.

The information theoretical problems are the following:

Type 1 is a linear transformation inside a hidden subspace of a ciphertext. Lemma 7 [OT12] states that the advantage of a PPT adversary $\mathcal{A}$ in a Type 1 distinguishing game is

$$\left|\mathsf{Adv}_{\mathcal{A}}^{(2\text{-}(h\text{-}1)\text{-}4)*}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2\text{-}h\text{-}1)*}(\lambda)\right| \leq \frac{2}{q}.$$

Type 2 is a linear transformation inside a hidden subspace of a ciphertext where the corresponding token is preserved. Lemma 9 [OT12] states that the advantage of a PPT adversary $\mathcal{A}$ in a Type 2 distinguishing game is

$$\left|\mathsf{Adv}_{\mathcal{A}}^{(2\text{-}h\text{-}2)*}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2\text{-}h\text{-}3)*}(\lambda)\right| \leq \frac{8}{q}.$$

Type 3 is a linear transformation across both hidden and partially public subspaces. Lemma 11 [OT12] states that the advantage of a PPT adversary $\mathcal{A}$ in a Type 3 distinguishing game is

$$\left|\mathsf{Adv}_{\mathcal{A}}^{(2\text{-}\nu\text{-}4)*}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(3)*}(\lambda)\right| \leq \frac{1}{q}.$$

We now give a detailed description of the needed computational problems and their respective proofs.

A.3 Basic Problem 0*

This is a modified version of **Basic Problem 0** [OT10, Definition 18]. Let $\lambda, \alpha \in \mathbb{N}$ and $\beta \in \{0,1\}$. We define a Basic Problem 0* generator, $\mathcal{G}_\beta^{\mathsf{bp0}^*}$, which on inputs 1^λ and α:

1. Samples $\kappa, \xi, \rho, \tau \xleftarrow{\$} \mathbb{F}_q^\times$ and $\delta, \sigma, \omega \xleftarrow{\$} \mathbb{F}_q$.
2. Samples $(q, \mathbb{G}, \mathbb{G}_T, g, e, \mathbb{V}, \mathbb{A}) \leftarrow \mathcal{G}_{\mathsf{dpvs}}$ and sets $\mathsf{pp} = (q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e, G_T)$ where $G_T = e(g,g)^{\kappa\xi}$.
3. For $1 \le \ell \le \alpha$:
   (a) Samples a random transformation, as described in Lemma 1, $X_\ell = (\chi_{\ell,1}, \chi_{\ell,2}, \chi_{\ell,3}) \xleftarrow{\$} GL(3, \mathbb{F}_q)$ and sets $(\nu_{\ell,1}, \nu_{\ell,2}, \nu_{\ell,3}) = ((X_\ell)^T)^{-1}$.
   (b) Computes $\boldsymbol{b}_{\ell,i} = \kappa \sum_{j=1}^3 \chi_{\ell,i,j} \boldsymbol{a}_j$ and sets $\hat{\mathbb{B}}_\ell = (\boldsymbol{b}_{\ell,1}, \boldsymbol{b}_{\ell,3})$.
   (c) Computes $\boldsymbol{b}^*_{\ell,i} = \xi \sum_{j=1}^3 \nu_{\ell,i,j} \boldsymbol{a}_j$ and sets $\mathbb{B}^*_\ell = (\boldsymbol{b}^*_{\ell,1}, \boldsymbol{b}^*_{\ell,2}, \boldsymbol{b}^*_{\ell,3})$.
   (d) Set $\boldsymbol{f}_\ell = (\omega, \tau, 0)_{\mathbb{B}_\ell}$.
   (e) Sets $\boldsymbol{y}_\ell^{(0)} = (\delta, 0, \sigma)_{\mathbb{B}^*_\ell}$ and $\boldsymbol{y}_\ell^{(1)} = (\delta, \rho, \sigma)_{\mathbb{B}^*_\ell}$.
4. Returns $(\mathsf{pp}, \{\hat{\mathbb{B}}_\ell, \mathbb{B}^*_\ell, \boldsymbol{y}_\ell^{(\beta)}, \boldsymbol{f}_\ell\}_{\ell=1,\cdots,\alpha}, \kappa g, \xi g, \delta\xi g)$.

Basic Problem 0* consists in guessing β given

$$(\mathsf{pp}, \{\hat{\mathbb{B}}_\ell, \mathbb{B}^*_\ell, \boldsymbol{y}_\ell^{(\beta)}, \boldsymbol{f}_\ell\}_{\ell=1,\cdots,\alpha}, \kappa g, \xi g, \delta\xi g) \leftarrow \mathcal{G}_\beta^{\mathsf{bp0}^*}(1^\lambda, \alpha).$$

We define the advantage of a PPT machine $\mathcal{A}_{\mathsf{bp0}^*}$ for Basic Problem 0* as

$$\mathsf{Adv}^{\mathsf{bp0}^*}_{\mathcal{A}_{\mathsf{bp0}^*}}(\lambda) = \Big| \Pr[\mathcal{A}_{\mathsf{bp0}^*}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_0^{\mathsf{bp0}^*}(1^\lambda, \alpha)] - \Pr[\mathcal{A}_{\mathsf{bp0}^*}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_1^{\mathsf{bp0}^*}(1^\lambda, \alpha)] \Big|$$

Lemma 3. *For any PPT adversary $\mathcal{A}_{bp0^*}$ for Basic Problem 0*, there exists a PPT distinguisher $\mathcal{D}$ for the DLIN problem such that for any security parameter $\lambda \in \mathbb{N}$,*

$$Adv^{bp0^*}_{\mathcal{A}_{bp0^*}}(\lambda) \le Adv^{DLIN}_{\mathcal{D}}(\lambda) + \frac{5}{q}.$$

Proof. Let $\mathcal{A}_{\mathsf{bp0}^*}$ be an adversary for Basic Problem 0*. We can then build $\mathcal{D}$, a distinguisher for the DLIN assumption, as follows:

1. $\mathcal{D}$ receives a DLIN instance $(\mathsf{param}_{\mathbb{G}}, g, \ \xi g, \kappa g, \delta\xi g, \sigma\kappa g, Y^{(\beta)})$, where $\mathsf{param}_{\mathbb{G}} = (q, \mathbb{G}, \mathbb{G}_T, g, e)$ and $Y^{(\beta)}$ is either $Y^{(0)} = (\delta + \sigma)g$ or $Y^{(1)} = \psi g \xleftarrow{\$} \mathbb{G}$.
2. $\mathcal{D}$ samples $(q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e) \xleftarrow{\$} \mathcal{G}_{\mathsf{dpvs}}(1^\lambda, 3, \mathsf{param}_{\mathbb{G}})$.
3. $\mathcal{D}$ computes $g_T = e(\kappa g, \xi g) = e(g,g)^{\kappa\xi}$ and sets $\mathsf{pp} = (q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e, g_T)$.
4. $\mathcal{D}$ considers[10] the following basis vectors

   $$\boldsymbol{u}_1 = (\kappa, 0, 0)_{\mathbb{A}}, \ \boldsymbol{u}_2 = (\text{-}\kappa, \text{-}\xi, \kappa\xi)_{\mathbb{A}}, \ \boldsymbol{u}_3 = (0, \xi, 0)_{\mathbb{A}}$$

   such that $\mathbb{U} = (\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3)$ is a basis of $\mathbb{V}$. Notice that from the given DLIN instance, $\mathcal{D}$ can efficiently compute $\boldsymbol{u}_1, \boldsymbol{u}_3$.

[10] In the next two steps $\mathcal{D}$ considers basis vectors of the matrices Π, Π^*,

$$\Pi = \begin{pmatrix} \kappa & & \\ \text{-}\kappa & \text{-}\xi & \kappa\xi \\ & \xi & 1 \end{pmatrix} \ \Pi^* = \begin{pmatrix} \xi & & 1 \\ & & 1 \\ & \kappa & 1 \end{pmatrix}$$

and observe that $\Pi(\Pi^*)^T = \kappa\xi I_3$. $\mathcal{D}$ cannot efficiently compute Π.

5. Similarly $\mathcal{D}$ considers
$$\boldsymbol{u}_1^* = (\xi, 0, 1)_{\mathbb{A}},\ \boldsymbol{u}_2^* = (0, 0, 1)_{\mathbb{A}},\ \boldsymbol{u}_3^* = (0, \kappa, 1)_{\mathbb{A}}$$
such that $\mathbb{U}^* = (\boldsymbol{u}_1^*, \boldsymbol{u}_2^*, \boldsymbol{u}_3^*)$ is a basis of $\mathbb{V}$. Notice that from the given DLIN instance, $\mathcal{D}$ can efficiently compute $\boldsymbol{u}_1^*, \boldsymbol{u}_2^*, \boldsymbol{u}_3^*$.

6. $\mathcal{D}$ samples $\eta, \varphi \xleftarrow{\$} \mathbb{F}_q$ such that $\eta \neq 0$ and sets
$$\boldsymbol{v} = (\varphi g, \text{-}\eta g, \eta\kappa g) = (\varphi, \text{-}\eta, \eta\kappa)_{\mathbb{A}}$$
and
$$\boldsymbol{w}^{(\beta)} = (\delta\xi g, \sigma\kappa g, Y^{(\beta)})$$

7. $\mathcal{D}$ generates α random linear transformations $W_1, \cdots, W_\alpha$ on $\mathbb{V}$, as shown in Lemma 1.

8. For $1 \leq \ell \leq \alpha$:
   (a) $\mathcal{D}$ calculates
   $$\boldsymbol{b}_{\ell,i} = W_\ell(\boldsymbol{u}_i) \text{ for } i = 1, 3,$$
   $$\boldsymbol{b}_{\ell,i}^* = (W_\ell^{-1})^T(\boldsymbol{u}_i^*) \text{ for } i = 1, 2, 3$$
   and sets $\hat{\mathbb{B}}_\ell = (\boldsymbol{b}_{\ell,1}, \boldsymbol{b}_{\ell,3})$ and $\mathbb{B}_\ell^* = (\boldsymbol{b}_{\ell,1}^*, \boldsymbol{b}_{\ell,2}^*, \boldsymbol{b}_{\ell,3}^*)$
   (b) $\mathcal{D}$ sets $\boldsymbol{f}_\ell = W_\ell(\boldsymbol{v})$ and $\boldsymbol{y}_\ell^{(\beta)} = (W_\ell^{-1})^T(\boldsymbol{w}^{(\beta)})$.

9. $\mathcal{D}$ sends $(\mathsf{pp}, \{\hat{\mathbb{B}}_\ell, \mathbb{B}_\ell^*, \boldsymbol{y}_\ell^{(\beta)}, \boldsymbol{f}_\ell\}_{\ell=1,\cdots,\alpha}, \kappa g, \xi g, \delta\xi g)$ to $\mathcal{A}_{\mathsf{bp0}^*}$ and returns whatever $\mathcal{A}_{\mathsf{bp0}^*}$ sends back.

For the moment assume that η and κ are all now zero, we will later account for the probability that each could be 0 Define $\tau \stackrel{def}{=} \xi^{-1}\eta$, since $\eta \neq 0$ it holds that $\tau \neq 0$. Similarly, define $\omega \stackrel{def}{=} \tau + \kappa^{-1}\varphi$, we have

$$\begin{aligned}\boldsymbol{f}_\ell = W_\ell(\boldsymbol{v}) &= W_\ell\big((\varphi, \text{-}\eta, \eta\kappa)_{\mathbb{A}}\big) = W_\ell\big(((\omega-\tau)\kappa, \text{-}\tau\xi, \tau\kappa\xi)_{\mathbb{A}}\big) \\ &= W_\ell\big(\omega\boldsymbol{u}_1 + \tau\boldsymbol{u}_2\big) = W_\ell\big((\omega, \tau, 0)_{\mathbb{U}}\big) = (\omega, \tau, 0)_{\mathbb{B}_\ell}\end{aligned}$$

When $\beta = 0$ and $Y^{(0)} = (\delta + \sigma)g$ we have

$$\begin{aligned}\boldsymbol{y}_\ell^{(0)} &= (W_\ell^{-1})^T\big(\delta\xi g, \sigma\kappa g, (\delta+\sigma)g\big) \\ &= (W_\ell^{-1})^T\big((\delta\xi, \sigma\kappa, \delta+\sigma)_{\mathbb{A}}\big) \\ &= (W_\ell^{-1})^T\big(\delta\boldsymbol{u}_1^* + \sigma\boldsymbol{u}_3^*\big) \\ &= (W_\ell^{-1})^T\big((\delta, 0, \sigma)_{\mathbb{U}^*}\big) \\ &= (\delta, 0, \sigma)_{\mathbb{B}_\ell^*}\end{aligned}$$

When $\beta = 1$ and $Y^{(1)} = \psi g$ where $\psi \xleftarrow{\$} \mathbb{F}_q$, if we define $\rho = \psi - \delta - \sigma$, we have

$$\begin{aligned}\boldsymbol{y}_\ell^{(1)} &= (W_\ell^{-1})^T\big(\delta\xi g, \sigma\kappa g, \psi g\big) \\ &= (W_\ell^{-1})^T\big(\delta\xi g, \sigma\kappa g, (\rho+\delta+\sigma)g\big) \\ &= (W_\ell^{-1})^T\big((\delta\xi, \sigma\kappa, \rho+\delta+\sigma)_{\mathbb{A}}\big) \\ &= (W_\ell^{-1})^T\big(\delta\boldsymbol{u}_1^* + \rho\boldsymbol{u}_2^* + \sigma\boldsymbol{u}_3^*\big) \\ &= (W_\ell^{-1})^T\big((\delta, \rho, \sigma)_{\mathbb{U}^*}\big) \\ &= (\delta, \rho, \sigma)_{\mathbb{B}_\ell^*}\end{aligned}$$

Since the k linear maps W_ℓ are sampled uniformly and independently, the distribution of the bases $\mathbb{B}_\ell$ and $\mathbb{B}_\ell^*$ is the same as if they had been generated using $\mathcal{G}_\beta^{\mathsf{bp0}^*}$. Then for the distributions of $\boldsymbol{f}_\ell, \boldsymbol{y}_\ell^{(\beta)}$ to match the ones of the inputs expected by $\mathcal{A}$, we need $\kappa, \rho, \xi \neq 0$. This is true except with probability $2/q$ when $\beta = 0$, and with probability $3/q$ when $\beta = 1$. We then have:

$$\mathsf{Adv}^{\mathsf{bp0}^*}_{\mathcal{A}_{\mathsf{bp0}^*}}(\lambda) \leq \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}}(\lambda) + \frac{5}{q}.$$

□

A.4 Basic Problem 1*

This is a modified version of **Problem 1** [OT12, Definition 8]. Let $\lambda, \alpha, n \in \mathbb{N}$, $\beta \in \{0,1\}$, and set $N = 4n/\alpha + 2$. We define a Basic Problem 1* generator, $\mathcal{G}_\beta^{\mathsf{bp1}^*}$, which on inputs 1^λ, α and n:

1. Samples $\omega, z \xleftarrow{\$} \mathbb{F}_q$.
2. Samples $(\mathsf{param}_{\mathbb{V}}, \{\mathbb{B}_\ell, \mathbb{B}_\ell^*\}_{\ell=1,\cdots,\alpha}) \leftarrow \mathcal{G}_{\mathsf{ob}}^{\mathsf{IPE}^*}(1^\lambda, N)$.
3. For $1 \leq \ell \leq \alpha$:
   (a) Sets $\hat{\mathbb{B}}_\ell^* = (\boldsymbol{b}_{\ell,0}^*, \cdots, \boldsymbol{b}_{\ell,n/\alpha}^*, \boldsymbol{b}_{\ell,3n/\alpha+1}^* \cdots, \boldsymbol{b}_{\ell,N-1}^*)$.
   (b) Samples $\gamma_\ell \xleftarrow{\$} \mathbb{F}_q$.
   (c) Sets $\boldsymbol{g}_{\ell,1}^{(0)} = (0, \omega\vec{e}_1, 0^{n/\alpha}, 0^{n/\alpha}, 0^{n/\alpha}, \gamma_\ell)_{\mathbb{B}_\ell}$ and $\boldsymbol{g}_{\ell,1}^{(1)} = (0, \omega\vec{e}_1, z\vec{e}_1, 0^{n/\alpha}, 0^{n/\alpha}, \gamma_\ell)_{\mathbb{B}_\ell}$.
   (d) For $2 \leq i \leq n/\alpha$, sets $\boldsymbol{g}_{\ell,i} = \omega \boldsymbol{b}_{\ell,i}$.
4. Return
$$(\mathsf{param}_{\mathbb{V}}, \{\mathbb{B}_\ell, \hat{\mathbb{B}}_\ell^*, \boldsymbol{g}_{\ell,1}^{(\beta)}, \{\boldsymbol{g}_{\ell,i}\}_{i=2,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha}).$$

Then Basic Problem 1* consists in guessing β given

$$(\mathsf{param}_{\mathbb{V}}, \{\mathbb{B}_\ell, \hat{\mathbb{B}}_\ell^*, \boldsymbol{g}_{\ell,1}^{(\beta)}, \{\boldsymbol{g}_{\ell,i}\}_{i=2,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha}) \leftarrow \mathcal{G}_\beta^{\mathsf{bp1}^*}(1^\lambda, n, \alpha).$$

We define the advantage of a PPT machine $\mathcal{A}_{\mathsf{bp1}^*}$ for Basic Problem 1* as

$$\mathsf{Adv}^{\mathsf{bp1}^*}_{\mathcal{A}_{\mathsf{bp1}^*}}(\lambda) = \Big| \Pr[\mathcal{A}_{\mathsf{bp1}^*}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_0^{\mathsf{bp1}^*}(1^\lambda, n, \alpha)] - \Pr[\mathcal{A}_{\mathsf{bp1}^*}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_1^{\mathsf{bp1}^*}(1^\lambda, n, \alpha)] \Big|$$

Lemma 4. *For any PPT adversary $\mathcal{A}_{bp1^*}$ for Basic Problem 1*, there exists a PPT distinguisher $\mathcal{D}$ for the DLIN problem such that for any security parameter $\lambda \in \mathbb{N}$,*

$$Adv^{bp1^*}_{\mathcal{A}_{bp1^*}}(\lambda) \leq Adv^{bp0^*}_{\mathcal{A}_{bp0^*}}(\lambda) \leq Adv^{DLIN}_{\mathcal{D}}(\lambda) + \frac{5}{q}.$$

Proof of Lemma 4. Let $\mathcal{A}_{\mathsf{bp1}^*}$ be an arbitrary adversary for Basic Problem 1*. Then we can build $\mathcal{A}_{\mathsf{bp0}^*}$, an adversary for Basic Problem 0* as follows:

1. Receive a Basic Problem 0* instance
$$(\mathsf{pp}, \{\hat{\mathbb{B}}_\ell, \mathbb{B}_\ell^*, \boldsymbol{y}_\ell^{(\beta)}, \boldsymbol{f}_\ell\}_{\ell=1,\cdots,\alpha}, \kappa g, \xi g, \delta\xi g) \leftarrow \mathcal{G}_\beta^{\mathsf{bp0}^*}(1^\lambda, \alpha).$$
2. Extract g_T and $\mathsf{param}_{\mathbb{G}}(q, \mathbb{G}, \mathbb{G}_T, g, e)$ from pp and run $(q, \mathbb{G}, \mathbb{G}_T, g, e, \mathbb{V}, \mathbb{A}) \leftarrow \mathcal{G}_{\mathsf{dpvs}}(1^\lambda, N, \mathsf{param}_{\mathbb{G}})$. Sets $\mathsf{param}_{\mathbb{V}} = (q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e, g_T)$.
3. For $1 \leq \ell \leq \alpha$:
   (a) Sample a random linear transformation W_ℓ on $\mathbb{V}$, $W_\ell = (w_{\ell,1}, \cdots, w_{\ell,N}) \xleftarrow{\$} GL(N, \mathbb{F}_q)$.

(b) Compute $\boldsymbol{g}_{\ell,1}^{(\beta)} = W_\ell(0, \boldsymbol{y}^{(\beta)}, 0^{N-4})$. (Recall that $\boldsymbol{y}^{(\beta)} \in \mathbb{G}^3$.)

(c) For $2 \leq i \leq n$, compute $\boldsymbol{g}_{\ell,i} = W_\ell(0^i, \delta\xi g, 0^{N-i-1})$.

(d) Compute:

$$\begin{aligned}
\boldsymbol{d}_{\ell,1} &= W_\ell(0, \boldsymbol{b}^*_{\ell,1}, 0^{N-4}),\\
\boldsymbol{d}_{\ell,n/\alpha+1} &= W_\ell(0, \boldsymbol{b}^*_{\ell,2}, 0^{N-4}),\\
\boldsymbol{d}_{\ell,N} &= W_\ell(0, \boldsymbol{b}^*_{\ell,3}, 0^{N-4}),\\
\{\boldsymbol{d}_{\ell,i} &= W_\ell(0^{i+1}, \xi g, 0^{N-i-2})\}_{i=0,2\leq i\leq n/\alpha}\\
\{\boldsymbol{d}_{\ell,i} &= W_\ell(0^{i}, \xi g, 0^{N-i-1})\}_{n/\alpha+2\leq i\leq N-1}.
\end{aligned}$$

(e) Consider the following vectors ($\boldsymbol{d}^*_{\ell,n/\alpha+1}$ is not efficiently computable)

$$\begin{aligned}
\boldsymbol{d}^*_{\ell,1} &= (W_\ell^{-1})^T(0, \boldsymbol{b}_{\ell,1}, 0^{N-4}),\\
\boldsymbol{d}^*_{\ell,n/\alpha+1} &= (W_\ell^{-1})^T(0, \boldsymbol{b}_{\ell,2}, 0^{N-4}),\\
\boldsymbol{d}^*_{\ell,N} &= (W_\ell^{-1})^T(0, \boldsymbol{b}_{\ell,3}, 0^{N-4}),\\
\{\boldsymbol{d}^*_{\ell,i} &= (W_\ell^{-1})^T(0^{i+1}, \kappa g, 0^{N-i-2})\}_{i=0,2\leq i\leq n/\alpha},\\
\{\boldsymbol{d}^*_{\ell,i} &= (W_\ell^{-1})^T(0^{i}, \kappa g, 0^{N-i-1})\}_{n/\alpha+2\leq i\leq N-1}.
\end{aligned}$$

(f) $\mathcal{A}_{\mathsf{bp0}^*}$ sets $\mathbb{D}_\ell = (\boldsymbol{d}_{\ell,0}, \cdots, \boldsymbol{d}_{\ell,N})$ and $\hat{\mathbb{D}}^*_\ell = (\boldsymbol{d}^*_{\ell,1}, \cdots, \boldsymbol{d}^*_{\ell,n/\alpha}, \boldsymbol{d}^*_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{d}^*_{\ell,N})$.

4. Send $(\mathsf{param}_{\mathbb{V}}, \{\mathbb{D}_\ell, \hat{\mathbb{D}}^*_\ell, \boldsymbol{g}^{(\beta)}_{\ell,1}, \{\boldsymbol{g}_{\ell,i}\}_{i=1,\cdots,n}\}_{\ell=1,\cdots,\alpha})$ to $\mathcal{A}_{\mathsf{bp1}^*}$ and output the response bit.

From $\hat{\mathbb{B}}_\ell = (\boldsymbol{b}_{\ell,1}, \boldsymbol{b}_{\ell,3})$ and ξg, $\mathcal{A}_{\mathsf{bp0}^*}$ is only able to compute $\boldsymbol{d}^*_{\ell,i}$ for $i = 0, \cdots, n/\alpha, n/\alpha+2, \cdots, N$. From $\mathbb{B}^* = (\boldsymbol{b}^*_{\ell,1}, \boldsymbol{b}^*_{\ell,2}, \boldsymbol{b}^*_{\ell,3})$ and κg, $\mathcal{A}_{\mathsf{bp0}^*}$ is able to compute $\boldsymbol{d}_{\ell,i}$ for $i = 0, \cdots, N$. Then for $1 \leq \ell \leq \alpha$, $\mathbb{D}_\ell$ and $\mathbb{D}^*_\ell$ are dual orthonormal bases. Then when we define

$$\omega \stackrel{def}{=} \delta, \gamma \stackrel{def}{=} \sigma, z \stackrel{def}{=} \rho,$$

we have

$$\begin{aligned}
\boldsymbol{g}^{(0)}_{\ell,1} &= (0, \omega\vec{e}_1, 0^{n/\alpha}, 0^{n/\alpha}, \gamma)_{\mathbb{D}_\ell}\\
\boldsymbol{g}^{(1)}_{\ell,1} &= (0, \omega\vec{e}_1, z\vec{e}_1, 0^{n/\alpha}, \gamma)_{\mathbb{D}_\ell}
\end{aligned}$$

and for $2 \leq i \leq n$,$\boldsymbol{g}_{\ell,i} = \omega\boldsymbol{d}_{\ell,i}$. We then have $\mathsf{Adv}^{\mathsf{bp1}^*}_{\mathcal{A}_{\mathsf{bp1}^*}}(\lambda) \leq \mathsf{Adv}^{\mathsf{bp0}^*}_{\mathcal{A}_{\mathsf{bp0}^*}}(\lambda) \leq \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}}(\lambda) + 5/q$.

Linear Algebra In the below we show that the linear system is properly prepared. Without loss of generality consider $\alpha = 1$. Then from BP0*, we have:

$$\begin{aligned}
\boldsymbol{u}^*_1 &= (\xi, 0, 1)_{\mathbb{A}} = (\xi g, 0, g)\\
\boldsymbol{u}^*_2 &= (0, 0, 1)_{\mathbb{A}} = (0, 0, g)\\
\boldsymbol{u}^*_3 &= (0, \kappa, 1)_{\mathbb{A}} = (0, \kappa g, g)
\end{aligned}$$

The matrix $(X^{-1})^T$ (from Basic Problem 0*) is a random linear transformation (i.e. a random 3×3 matrix):

$$(X^{-1})^T = \begin{pmatrix} x_{1,1} & x_{1,2} & x_{1,3}\\ x_{2,1} & x_{2,2} & x_{2,3}\\ x_{3,1} & x_{3,2} & x_{3,3}. \end{pmatrix}$$

As a result for $\mathbb{B}^* = (\boldsymbol{b}_1^*, \boldsymbol{b}_2^*, \boldsymbol{b}_3^*)$:

$$\begin{aligned}
\boldsymbol{b}_1^* &= (X^{-1})^T(\boldsymbol{u}_1^*) = (X^{-1})^T(\xi g, 0, g) \\
&= \Big((x_{1,1}\xi + x_{1,3})g, (x_{2,1}\xi + x_{2,3})g, (x_{3,1}\xi + x_{3,3})g\Big) \\
\boldsymbol{b}_2^* &= (X^{-1})^T(\boldsymbol{u}_2^*) \\
&= (X^{-1})^T(0, 0, g) \\
&= \Big(x_{1,3}g, x_{2,3}g, x_{3,3}g\Big) \\
\boldsymbol{b}_3^* &= (X^{-1})^T(\boldsymbol{u}_3^*) \\
&= (X^{-1})^T(0, \kappa g, g) \\
&= \Big((x_{1,2}\kappa + x_{1,3})g, (x_{2,2}\kappa + x_{2,3})g, (x_{3,2}\kappa + x_{3,3})g\Big)
\end{aligned}$$

From BP1* we have the random linear transformation (i.e. random $N \times N$ matrix) W:

$$W = \begin{pmatrix} w_{1,1} & \cdots & w_{1,N} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,N} \end{pmatrix}$$

and we obtain $\mathbb{D} = (\boldsymbol{d}_0, \cdots, \boldsymbol{d}_{N-1})$ as follows:

$$\begin{aligned}
\boldsymbol{d}_j &= W(0^{j+1}, \xi g, 0^{N-j-2}) = \Big(w_{1,j+2}\xi g, \cdots, w_{N,j+2}\xi g\Big), \text{ for } j \in \{0, 2, 3, ..., n/\alpha\}, \\
\boldsymbol{d}_1 &= W(0, \boldsymbol{b}_1^*, 0^{N-4}) = W\Big(0, (x_{1,1}\xi + x_{1,3})g, (x_{2,1}\xi + x_{2,3})g, (x_{3,1}\xi + x_{3,3})g, 0^{N-4}\Big) \\
&= \Big((w_{i,2}x_{1,1} + w_{i,3}x_{2,1} + w_{i,4}x_{3,1})\xi g + (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})g\Big)_{i=1,\cdots,N} \\
\boldsymbol{d}_{n/\alpha+1} &= W(0, \boldsymbol{b}_2^*, 0^{N-4}) = W\Big(0, x_{1,3}g, x_{2,3}g, x_{3,3}g, 0^{N-4}\Big) = \Big((w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})g\Big)_{i=1,\cdots,N} \\
\boldsymbol{d}_j &= W(0^j, \boldsymbol{b}_2^*, 0^{N-j-1}) = \Big(w_{1,j+1}\xi g, \cdots, w_{N,j+1}\xi g\Big), \\
&\quad \text{for } j \in \{n/\alpha + 2, \cdots, N-1\} \\
\boldsymbol{d}_{N-1} &= W(0, \boldsymbol{b}_3^*, 0^{N-4}) = W\Big(0, (x_{1,2}\kappa + x_{1,3})g, (x_{2,2}\kappa + x_{2,3})g, (x_{3,2}\kappa + x_{3,3})g, 0^{N-4}\Big) \\
&= \Big((w_{i,2}x_{1,2} + w_{i,3}x_{2,2} + w_{i,4}x_{3,2})\kappa g + (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})g\Big)_{i=1,\cdots,N}
\end{aligned}$$

Similarly, from BP0* we have:

$$\begin{aligned}
\boldsymbol{y}^{(0)} &= (\delta, 0, \sigma)_{\mathbb{B}^*} = \Big((x_{i,1}\xi + x_{i,3})\delta g + (x_{i,2}\kappa + x_{i,3})\sigma g\Big)_{i=1,2,3}, \\
\boldsymbol{y}^{(1)} &= (\delta, \rho, \sigma)_{\mathbb{B}^*} = \Big((x_{i,1}\xi + x_{i,3})\delta g + \rho\, x_{i,3}g + (x_{i,2}\kappa + x_{i,3})\sigma g\Big)_{i=1,2,3}
\end{aligned}$$

From BP1* we have:

$$\begin{aligned}
\boldsymbol{g}_1^{(0)} &= W(0, \boldsymbol{y}^{(0)}, 0^{N-4}) \\
&= W\Big(0, (x_{1,1}\xi + x_{1,3})\delta G + (x_{1,2}\kappa + x_{1,3})\sigma g, (x_{2,1}\xi + x_{2,3})\delta g + (x_{2,2}\kappa + x_{2,3})\sigma g, (x_{3,1}\xi + x_{3,3})\delta g + (x_{3,2}\kappa + x_{3,3})\sigma g, 0^{N-4}\Big) \\
&= \Big((w_{i,2}x_{1,1} + w_{i,3}x_{2,1} + w_{i,4}x_{3,1})\delta\xi g + (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})\delta g \\
&+ (w_{i,2}x_{1,2} + w_{i,3}x_{2,2} + w_{i,4}x_{3,2})\sigma\kappa g + (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})\sigma g\Big)_{i=1,\cdots,N}
\end{aligned}$$

and

$$\begin{aligned}
\boldsymbol{g}_1^{(1)} &= W(0, \boldsymbol{y}^{(1)}, 0^{N-4}) \\
&= W\Big(0, (x_{1,1}\xi + x_{1,3})\delta g + (x_{1,2}\kappa + x_{1,3})\sigma g, (x_{2,1}\xi + x_{2,3})\delta g + (x_{2,2}\kappa + x_{2,3})\sigma g, \\
&(x_{3,1}\xi + x_{3,3})\delta g + (x_{3,2}\kappa + x_{3,3})\sigma g, 0^{N-4}\Big) \\
&= \Big((w_{i,2}x_{1,1} + w_{i,3}x_{2,1} + w_{i,4}x_{3,1})\delta\xi g + (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})\delta g \\
&+ (w_{i,2}x_{1,2} + w_{i,3}x_{2,2} + w_{i,4}x_{3,2})\sigma\kappa g + (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})\sigma g\Big)_{i=1,\cdots,N}
\end{aligned}$$

Notice that for $\omega \stackrel{def}{=} \delta$, $z \stackrel{def}{=} \rho$ and $\gamma \stackrel{def}{=} \sigma$:

$$\begin{aligned}
(0, \omega\vec{e}_1, 0^{n/\alpha}, 0^n, \gamma)_{\mathbb{D}} &= (0, \delta, 0^{n/\alpha-1}, 0^{n/\alpha}, 0^n, \sigma)_{\mathbb{D}} \\
&= \delta\boldsymbol{d}_2 + \sigma\boldsymbol{d}_N \\
&= \Big((w_{i,2}x_{1,1} + w_{i,3}x_{2,1} + w_{i,4}x_{3,1})\delta\xi g \\
&+ (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})\delta g \\
&+ (w_{i,2}x_{1,2} + w_{i,3}x_{2,2} + w_{i,4}x_{3,2})\sigma\kappa g \\
&+ (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})\sigma g\Big)_{i=1,\cdots,N} \\
&= \boldsymbol{g}_1^{(0)} \\
(0, \omega\vec{e}_1, z\vec{e}_1, 0^n, \gamma)_{\mathbb{D}} &= (0, \delta, 0^{n/\alpha-1}, \rho, 0^{n/\alpha-1}, 0^{n/\alpha}, \sigma)_{\mathbb{D}} \\
&= \delta\boldsymbol{d}_2 + \rho\boldsymbol{d}_{n/\alpha+1} + \sigma\boldsymbol{d}_N \\
&= \Big((w_{i,2}x_{1,1} + w_{i,3}x_{2,1} + w_{i,4}x_{3,1})\delta\xi g \\
&+ (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})\delta g \\
&+ (w_{i,2}x_{1,2} + w_{i,3}x_{2,2} + w_{i,4}x_{3,2})\sigma\kappa g \\
&+ (w_{i,2}x_{1,3} + w_{i,3}x_{2,3} + w_{i,4}x_{3,3})\sigma g\Big)_{i=1,\cdots,N} \\
&= \boldsymbol{g}_1^{(1)}
\end{aligned}$$

This completes the proof of Lemma 4. □

A.5 Basic Problem 2*

This is a modified version of **Problem 2** [OT12, Definition 9]. Let $\lambda, \alpha, n \in \mathbb{N}$ and $\beta \in \{0,1\}$ and set $N = 4n/\alpha + 2$. We define a Basic Problem 2* generator, $\mathcal{G}_\beta^{\mathsf{bp2}^*}(1^\lambda, \alpha, n)$:

1. Sample $\delta, \delta_0, \tau, \omega, \sigma \xleftarrow{\$} \mathbb{F}_q$.
2. Sample $(\mathsf{param}_{\mathbb{V}}, \{\mathbb{B}_\ell, \mathbb{B}_\ell^*\}_{\ell=1,\cdots,\alpha}) \leftarrow \mathcal{G}_{\mathsf{ob}}^{\mathsf{IPE}^*}(1^\lambda, N)$.
3. For $1 \le \ell \le \alpha$ set
$$\hat{\mathbb{B}}_\ell = (\boldsymbol{b}_{\ell,0}, \cdots, \boldsymbol{b}_{\ell,n/\alpha}, \boldsymbol{b}_{\ell,3n/\alpha+1} \cdots, \boldsymbol{b}_{\ell,N}).$$
4. For $1 \le \ell \le \alpha$, for $1 \le i \le n/\alpha$:
   (a) Set $\boldsymbol{h}_{\ell,i}^{(0)} = (0, \delta\vec{e}_i, 0^{n/\alpha}, 0^{n/\alpha}, \delta_0\vec{e}_i, 0)_{\mathbb{B}_\ell}$ and $\boldsymbol{h}_{\ell,i}^{(1)} = (0, \delta\vec{e}_i, \tau\vec{e}_i, 0^{n/\alpha}, \delta_0\vec{e}_i, 0)_{\mathbb{B}_\ell}$.
   (b) Set $\boldsymbol{g}_{\ell,i} = (0, \omega\vec{e}_i, \sigma\vec{e}_i, 0^{n/\alpha}, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell}$.

5. Return
$$(\mathsf{param}_{\mathbb{V}}, \{\hat{\mathbb{B}}_\ell, \mathbb{B}^*_\ell, \{\boldsymbol{h}^{(\beta)}_{\ell,i}, \boldsymbol{g}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha}).$$

Basic Problem 2* is to guess β given $(\mathsf{param}_{\mathbb{V}}, \{\hat{\mathbb{B}}_\ell, \mathbb{B}^*_\ell, \{h^{(\beta)}_{\ell,i}, g_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha}) \leftarrow \mathcal{G}^{\mathsf{bp2}^*}_\beta(1^\lambda, n, \alpha)$. We define the advantage of a PPT machine $\mathcal{A}_{\mathsf{bp2}^*}$ for Basic Problem 2* as

$$\mathsf{Adv}^{\mathsf{bp2}^*}_{\mathcal{A}_{\mathsf{bp2}^*}}(\lambda) = \left| \Pr[\mathcal{A}_{\mathsf{bp2}^*}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}^{\mathsf{bp2}^*}_0(1^\lambda, n, \alpha)] - \Pr[\mathcal{A}_{\mathsf{bp2}^*}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}^{\mathsf{bp2}^*}_1(1^\lambda, n, \alpha)] \right|$$

Lemma 5. *Let $\lambda \in \mathbb{N}$ be a security parameter. For any PPT adversary $\mathcal{A}_{bp2^*}$ for Basic Problem 2*, there exists a PPT adversary $\mathcal{A}_{bp0^*}$ for Basic Problem 0* and a PPT distinguisher $\mathcal{D}$ for the DLIN problem such that,*

$$Adv^{bp2^*}_{\mathcal{A}_{bp2^*}}(\lambda) = Adv^{bp0^*}_{\mathcal{A}_{bp0^*}}(\lambda) \leq Adv^{DLIN}_{\mathcal{D}}(\lambda) + \frac{5}{q}.$$

Proof of Lemma 5. Let $\mathcal{A}_{\mathsf{bp2}^*}$ be an arbitrary adversary for Basic Problem 2*. Then we can build $\mathcal{A}_{\mathsf{bp0}^*}$, an adversary for Basic Problem 0* as follows:

1. Receive a Basic Problem 0* instance
$$(\mathsf{pp}, \{\hat{\mathbb{B}}_\ell, \mathbb{B}^*_\ell, \boldsymbol{y}^{(\beta)}_\ell, \boldsymbol{f}_\ell\}_{\ell=1,\cdots,\alpha}, \kappa g, \xi g, \delta\xi g) \leftarrow \mathcal{G}^{\mathsf{bp0}^*}_\beta(1^\lambda, \alpha).$$
2. Extract g_T and $\mathsf{param}_{\mathbb{G}}(q, \mathbb{G}, \mathbb{G}_T, G, e)$ from pp, run $(q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e) \leftarrow \mathcal{G}_{\mathsf{dpvs}}(1^\lambda, N, \mathsf{param}_{\mathbb{G}})$. Set $\mathsf{param}_{\mathbb{V}} = (q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e, g_T)$.
3. For $1 \leq \ell \leq \alpha$:
   (a) Sample a random linear transformation $W_\ell = (w_{\ell,1}, \cdots, w_{\ell,N}) \xleftarrow{\$} GL(N, \mathbb{F}_q)$.
   (b) For $1 \leq i \leq n/\alpha$, compute
   $$\boldsymbol{g}_{\ell,i} = W_\ell(0, 0^{3(i-1)}, \boldsymbol{f}_\ell, 0^{3(n-i)}, 0).$$
   (c) For $1 \leq i \leq n/\alpha$, compute
   $$\boldsymbol{h}^{(\beta)}_{\ell,i} = (W_\ell^{-1})^T(0, 0^{3(i-1)}, \boldsymbol{y}^{(\beta)}_\ell, 0^{3(N-i)}, 0).$$
   (d) Compute $\boldsymbol{d}_{\ell,0} = W_\ell(\kappa g, 0^{N-1})$ and $\boldsymbol{d}_{\ell,N} = W_\ell(0^{N-1}, \kappa g)$.
   (e) For $1 \leq i \leq n/\alpha$ and $1 \leq j \leq 3$, compute
   $$\boldsymbol{d}_{\ell,n(j-1)+i} = W_\ell(0, 0^{3(i-1)}, \boldsymbol{b}_{\ell,j}, 0^{3(n-i)}, 0).$$
   (f) Compute $\boldsymbol{d}^*_{\ell,0} = (W_\ell^{-1})^T(\xi g, 0^{N-1})$ and $\boldsymbol{d}^*_{\ell,N} = (W_\ell^{-1})^T(0^{N-1}, \xi g)$.
   (g) For $1 \leq i \leq n/\alpha$ and $1 \leq j \leq 3$, compute
   $$\boldsymbol{d}^*_{\ell,n(j-1)+i} = (W_\ell^{-1})^T(0, 0^{3(i-1)}, \boldsymbol{b}^*_{\ell,j}, 0^{3(n-i)}, 0).$$
   (h) Sets $\mathbb{D}^*_\ell = (\boldsymbol{d}^*_{\ell,0}, \cdots, \boldsymbol{d}^*_{\ell,N})$ and $\hat{\mathbb{D}}_\ell = (\boldsymbol{d}_{\ell,0}, \cdots, \boldsymbol{d}_{\ell,n/\alpha}, \boldsymbol{d}_{\ell,2n/\alpha+1}, \cdots, \boldsymbol{d}_{\ell,N})$.
4. Send
$$(\mathsf{param}_{\mathbb{V}}, \{\mathbb{D}^*_\ell, \hat{\mathbb{D}}_\ell, \{\boldsymbol{h}^{(\beta)}_{\ell,i}, \boldsymbol{g}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha})$$
   to $\mathcal{A}_{\mathsf{bp2}^*}$.
5. Return β' from $\mathcal{A}_{\mathsf{bp2}^*}$.

From $\hat{\mathbb{B}}_\ell = (\boldsymbol{b}_{\ell,1}, \boldsymbol{b}_{\ell,3})$ and ξg, $\mathcal{A}_{\mathsf{bp0}^*}$ is able to compute $\boldsymbol{d}_{\ell,j}$ for $j = 0, \cdots, n/\alpha, 2n/\alpha+1, \cdots, N$. Similarly, from $\mathbb{B}^* = (\boldsymbol{b}^*_{\ell,1}, \boldsymbol{b}^*_{\ell,2}, \boldsymbol{b}^*_{\ell,3})$ and κg, $\mathcal{A}_{\mathsf{bp0}^*}$ can compute $\boldsymbol{d}_{\ell,j}$ for $j = 0, \cdots, N$. Then for $1 \le \ell \le \alpha$, $\mathbb{D}_\ell$ and $\mathbb{D}^*_\ell$ are dual orthonormal bases. Then we have for $1 \le i \le n/\alpha$:

$$\begin{aligned}
\boldsymbol{h}^{(0)}_{\ell,i} &= (0, \delta\vec{e}_i, 0^{n/\alpha}, 0^{n/\alpha}, \sigma\vec{e}_i, 0)^*_{\mathbb{D}_\ell} \\
\boldsymbol{h}^{(1)}_{\ell,i} &= (0, \delta\vec{e}_i, \rho\vec{e}_i, 0^{n/\alpha}, \sigma\vec{e}_i, 0)^*_{\mathbb{D}_\ell} \\
\boldsymbol{g}_{\ell,i} &= (0, \omega\vec{e}_i, \tau\vec{e}_i, 0^{n/\alpha}, 0^{n/\alpha}, 0)_{\mathbb{D}_\ell}.
\end{aligned}$$

We then have

$$\mathsf{Adv}^{\mathsf{bp2}^*}_{\mathcal{A}_{\mathsf{bp2}^*}}(\lambda) = \mathsf{Adv}^{\mathsf{bp0}^*}_{\mathcal{A}_{\mathsf{bp0}^*}}(\lambda) \le \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{D}}(\lambda) + \frac{5}{q}.$$

This completes the proof of Lemma 5 □

A.6 Basic Problem 3*

This is a modified version of **Problem 3** [OT12, Definition 10]. Let $\lambda, \alpha, n \in \mathbb{N}$ and $\beta \in \{0,1\}$, and set $N = 4n/\alpha + 2$. We define a Basic Problem 3* generator, $\mathcal{G}^{\mathsf{bp3}^*}_\beta$, which on inputs 1^λ, α and n:

1. Samples $\tau, \delta_0, \omega', \omega'', \kappa', \kappa'' \xleftarrow{\$} \mathbb{F}_q$.
2. Samples $(\mathsf{param}_{\mathbb{V}}, \{\mathbb{B}_\ell, \mathbb{B}^*_\ell\}_{\ell=1,\cdots,\alpha}) \leftarrow \mathcal{G}^{\mathsf{IPE}^*}_{\mathsf{ob}}(1^\lambda, N, \alpha)$.
3. For $1 \le \ell \le \alpha$:
   (a) Set $\hat{\mathbb{B}}_\ell = (\boldsymbol{b}_{\ell,0}, \cdots, \boldsymbol{b}_{\ell,n/\alpha}, \boldsymbol{b}_{\ell,3n/\alpha+1} \cdots, \boldsymbol{b}_{\ell,N-1})$.
   (b) Sets $\hat{\mathbb{B}}^*_\ell = (\boldsymbol{b}_{\ell,0}, \cdots, \boldsymbol{b}_{\ell,n/\alpha}, \boldsymbol{b}_{\ell,2n/\alpha+1} \cdots, \boldsymbol{b}_{\ell,N-1})$.
4. For $1 \le \ell \le \alpha$, for $1 \le i \le n/\alpha$:
   (a) Sets $\boldsymbol{h}^{(0)}_{\ell,i} = (0, 0^{n/\alpha}, \tau\vec{e}_i, 0^{n/\alpha}, \delta_0\vec{e}_i, 0)_{\mathbb{B}^*_\ell}$ and $\boldsymbol{h}^{(1)}_{\ell,i} = (0, 0^{n/\alpha}, 0^{n/\alpha}, \tau\vec{e}_i, \delta_0\vec{e}_i, 0)_{\mathbb{B}^*_\ell}$.
   (b) Sets $\boldsymbol{g}_{\ell,i} = (0, 0^{n/\alpha}, \omega'\vec{e}_i, \omega''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell}$.
   (c) Sets $\boldsymbol{f}_{\ell,i} = (0, 0^{n/\alpha}, \kappa'\vec{e}_i, \kappa''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell}$.
5. Return $(\mathsf{param}_{\mathbb{V}},$
   $\{\hat{\mathbb{B}}_\ell, \hat{\mathbb{B}}^*_\ell, \{\boldsymbol{h}^{(\beta)}_{\ell,i}, \boldsymbol{g}_{\ell,i}, \boldsymbol{f}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha})$.

Basic Problem 3* consists in guessing β given

$$(\mathsf{param}_{\mathbb{V}}, \{\hat{\mathbb{B}}_\ell, \hat{\mathbb{B}}^*_\ell, \{\boldsymbol{h}^{(\beta)}_{\ell,i}, \boldsymbol{g}_{\ell,i}, \boldsymbol{f}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha}) \leftarrow \mathcal{G}^{\mathsf{bp3}^*}_\beta(1^\lambda, n, \alpha).$$

We define the advantage of a PPT machine $\mathcal{A}_{\mathsf{bp3}^*}$ for Basic Problem 3* as

$$\mathsf{Adv}^{\mathsf{bp3}^*}_{\mathcal{A}_{\mathsf{bp3}^*}}(\lambda) = \left| \Pr[\mathcal{A}_{\mathsf{bp3}^*}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}^{\mathsf{bp3}^*}_0(1^\lambda, n, \alpha)] - \Pr[\mathcal{A}_{\mathsf{bp3}^*}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}^{\mathsf{bp3}^*}_1(1^\lambda, n, \alpha)] \right|$$

Lemma 6. *For any PPT adversary $\mathcal{A}_{bp3^*}$ for Basic Problem 3*, there exists a PPT distinguisher $\mathcal{D}$ for the DLIN problem such that for any security parameter $\lambda \in \mathbb{N}$,*

$$Adv^{bp3^*}_{\mathcal{A}_{bp3^*}}(\lambda) \le Adv^{bp2^*}_{\mathcal{A}_{bp2^*}}(\lambda) + \frac{2}{q} \le Adv^{DLIN}_{\mathcal{D}}(\lambda) + \frac{7}{q}.$$

Proof of Lemma 6. Basic Problem 3* can be decomposed into two experiments, Experiment 3-1 and 3-2 (Definitions 10 and 11 respectively). We will show that these two games are close and then use the triangle inequality. We now define these experiments.

Definition 10 (Experiment 3-1)**.** *Let $\eta \in \{0,1\}$. We define the Experiment* 3-1 *generator* $\mathcal{G}_\eta^{\exp 3\text{-}1}(1^\lambda, n, \alpha)$*:*

1. *Samples* $(\mathsf{param}_{\mathbb{V}}, \{\mathbb{B}_\ell, \mathbb{B}_\ell^*\}_{1\le\ell\le\alpha}) \leftarrow \mathcal{G}_{ob}^{IPE^*}(1^\lambda, N, \alpha)$.
2. *For* $1 \le \ell \le \alpha$*, sets* $\hat{\mathbb{B}}_\ell = (\boldsymbol{b}_{\ell,0}, \cdots, \boldsymbol{b}_{\ell,n/\alpha}, \boldsymbol{b}_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{b}_{\ell,N})$ *and* $\hat{\mathbb{B}}_\ell^* = (\boldsymbol{b}_{\ell,0}^*, \cdots, \boldsymbol{b}_{\ell,n/\alpha}^*, \boldsymbol{b}_{\ell,2n/\alpha+1}^*, \cdots, \boldsymbol{b}_{\ell,N}^*)$.
3. *Samples* $\tau, \tau', \delta_0, \omega', \omega'', \kappa', \kappa'' \xleftarrow{\$} \mathbb{F}_q$.
4. *For* $1 \le \ell \le \alpha$*, for* $1 \le i \le n/\alpha$ *set:*

$$\begin{aligned}
\boldsymbol{h}_{\ell,i}^{(0)} &= (0, 0^{n/\alpha}, \tau\vec{e}_i, 0^{n/\alpha}, \delta_0\vec{e}_i, 0)_{\mathbb{B}_\ell^*},\\
\boldsymbol{h}_{\ell,i}^{(1)} &= (0, 0^{n/\alpha}, \tau\vec{e}_i, \tau'\vec{e}_i, \delta_0\vec{e}_i, 0)_{\mathbb{B}_\ell^*},\\
\boldsymbol{g}_{\ell,i} &= (0, 0^{n/\alpha}, \omega'\vec{e}_i, \omega''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell},\\
\boldsymbol{f}_{\ell,i} &= (0, 0^{n/\alpha}, \kappa'\vec{e}_i, \kappa''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell}.
\end{aligned}$$

5. *Return* $(\mathsf{param}_{\mathbb{V}}, \{\hat{\mathbb{B}}_\ell, \hat{\mathbb{B}}_\ell^*, \{\boldsymbol{h}_{\ell,i}^{(\eta)}, \boldsymbol{g}_{\ell,i}, \boldsymbol{f}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha})$.

Experiment 3-1 *consists in guessing* $\eta \in \{0,1\}$ *given*

$$(\mathsf{param}_{\mathbb{V}}, \{\hat{\mathbb{B}}_\ell, \hat{\mathbb{B}}_\ell^*, \{\boldsymbol{h}_{\ell,i}^{(\eta)}, \boldsymbol{g}_{\ell,i}, \boldsymbol{f}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha}) \leftarrow \mathcal{G}_\eta^{\exp 3\text{-}1}(1^\lambda, n, \alpha).$$

We define the advantage of a PPT machine $\mathcal{D}$ *for Experiment 3-1 as*

$$\mathsf{Adv}_{\mathcal{D}}^{\exp 3\text{-}1}(\lambda) = \left| \Pr[\mathcal{D}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_0^{\exp 3\text{-}1}(1^\lambda, n, \alpha)] - \Pr[\mathcal{D}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_1^{\exp 3\text{-}1}(1^\lambda, n, \alpha)] \right|$$

Definition 11 (Experiment 3-2)**.** *Let* $\eta \in \{1,2\}$*. We define the Experiment 3-2 generator* $\mathcal{G}_\eta^{\exp 3\text{-}2}(1^\lambda, n, \alpha)$*:*

1. *Samples* $(\mathsf{param}_{\mathbb{V}}, \{\mathbb{B}_\ell, \mathbb{B}_\ell^*\}_{1\le\ell\le\alpha}) \leftarrow \mathcal{G}_{ob}^{IPE^*}(1^\lambda, N, \alpha)$.
2. *For* $1 \le \ell \le \alpha$*, sets*

$$\begin{aligned}
\hat{\mathbb{B}}_\ell &= (\boldsymbol{b}_{\ell,0}, \cdots, \boldsymbol{b}_{\ell,n/\alpha}, \boldsymbol{b}_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{b}_{\ell,N})\\
\hat{\mathbb{B}}^* &= (\boldsymbol{b}_{\ell,0}^*, \cdots, \boldsymbol{b}_{\ell,n/\alpha}^*, \boldsymbol{b}_{\ell,2n/\alpha+1}^*, \cdots, \boldsymbol{b}_{\ell,N}^*).
\end{aligned}$$

3. *Samples* $\tau, \tau', \delta_0, \omega', \omega'', \kappa', \kappa'' \xleftarrow{\$} \mathbb{F}_q$.
4. *For* $1 \le \ell \le \alpha$*, for* $1 \le i \le n/\alpha$ *set:*

$$\begin{aligned}
\boldsymbol{h}_{\ell,i}^{(1)} &= (0, 0^{n/\alpha}, \tau\vec{e}_i, \tau'\vec{e}_i, \delta_0\vec{e}_i, 0)_{\mathbb{B}_\ell^*},\\
\boldsymbol{h}_{\ell,i}^{(2)} &= (0, 0^{n/\alpha}, 0^{n/\alpha}, \tau'\vec{e}_i, \delta_0\vec{e}_i, 0)_{\mathbb{B}_\ell^*},\\
\boldsymbol{g}_{\ell,i} &= (0, 0^{n/\alpha}, \omega'\vec{e}_i, \omega''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell)},\\
\boldsymbol{f}_{\ell,i} &= (0, 0^{n/\alpha}, \kappa'\vec{e}_i, \kappa''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell}.
\end{aligned}$$

5. *Return* $(\mathsf{param}_{\mathbb{V}},$
$\{\hat{\mathbb{B}}_\ell, \hat{\mathbb{B}}_\ell^*, \{\boldsymbol{h}_{\ell,i}^{(\eta)}, \boldsymbol{g}_{\ell,i}, \boldsymbol{f}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha})$.

Experiment 3-2 *consists in guessing* $\eta \in \{1,2\}$ *given*

$$(\mathsf{param}_{\mathbb{V}}, \{\hat{\mathbb{B}}_\ell, \hat{\mathbb{B}}_\ell^*, \{\boldsymbol{h}_{\ell,i}^{(\eta)}, \boldsymbol{g}_{\ell,i}, \boldsymbol{f}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha}) \leftarrow \mathcal{G}_\eta^{\exp 3\text{-}1}(1^\lambda, n, \alpha).$$

We define the advantage of a PPT machine $\mathcal{D}$ *for Experiment* 3-2 *as*

$$\mathsf{Adv}_{\mathcal{D}}^{\exp 3\text{-}2}(\lambda) = \left| \Pr[\mathcal{D}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_1^{\exp 3\text{-}2}(1^\lambda, n, \alpha)] - \Pr[\mathcal{D}(1^\lambda, X) = 1 \mid X \leftarrow \mathcal{G}_2^{\exp 3\text{-}2}(1^\lambda, n, \alpha)] \right|$$

Lemma 7. *For any PPT distinguisher $\mathcal{D}$ and for any security parameter $\lambda \in \mathbb{N}$,*

$$\mathsf{Adv}_{\mathcal{D}}^{\exp 3\text{-}1}(\lambda) \leq \frac{1}{q}$$

Proof. Sample $\theta \xleftarrow{\$} \mathbb{F}_q$. Then for $1 \leq i \leq n/\alpha$ set

$$\begin{aligned}\boldsymbol{d}_{\ell,2n/\alpha+i} &= \boldsymbol{b}_{\ell,2n/\alpha+i} - \theta \boldsymbol{b}_{\ell,n/\alpha+i},\\ \boldsymbol{d}^*_{n/\alpha+i} &= \boldsymbol{b}^*_{\ell,n/\alpha+i} - \theta \boldsymbol{b}^*_{\ell,2n/\alpha+i}.\end{aligned}$$

For $1 \leq \ell \leq \alpha$, define

$$\begin{aligned}\mathbb{D}_\ell &= (\boldsymbol{b}_{\ell,0}, \cdots, \boldsymbol{b}_{\ell,2n/\alpha}, \boldsymbol{d}_{\ell,2n/\alpha+1}, \cdots, \boldsymbol{d}_{\ell,3n/\alpha}, \boldsymbol{b}_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{b}_{\ell,N-1}),\\ \mathbb{D}^*_\ell &= (\boldsymbol{b}^*_{\ell,0}, \cdots, \boldsymbol{b}^*_{\ell,n/\alpha}, \boldsymbol{d}^*_{\ell,n/\alpha+1}, \cdots, \boldsymbol{d}^*_{\ell,2n/\alpha}, \boldsymbol{b}^*_{\ell,2n/\alpha+1}, \cdots, \boldsymbol{b}^*_{\ell,N-1})\end{aligned}$$

which form dual orthonormal bases. Then we have

$$\begin{aligned}\boldsymbol{h}^{(0)}_{\ell,i} &= (0, 0^{n/\alpha}, \tau\vec{e}_i, 0^{n/\alpha}, \delta_0\vec{e}_i, 0)_{\mathbb{B}^*_\ell}\\ &= (0, 0^{n/\alpha}, \tau\vec{e}_i, \tau'\vec{e}_i, \delta_0\vec{e}_i, 0)_{\mathbb{D}^*_\ell}\\ \boldsymbol{g}_{\ell,i} &= (0, 0^{n/\alpha}, \omega'\vec{e}_i, \omega''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell}\\ &= (0, 0^{n/\alpha}, \tilde{\omega}'\vec{e}_i, \omega''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{D}_\ell}\\ \boldsymbol{f}_{\ell,i} &= (0, 0^{n/\alpha}, \kappa'\vec{e}_i, \kappa''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell}\\ &= (0, 0^{n/\alpha}, \tilde{\kappa}'\vec{e}_i, \kappa''\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{D}_\ell}\end{aligned}$$

In the above, τ' = -$\theta\tau$, $\tilde{\omega}' = \omega' + \theta\omega''$ and $\tilde{\kappa}' = \kappa' + \theta\kappa''$. Notice that since θ, ω' and κ' are sampled independently and uniformly, then τ', $\tilde{\omega}'$ and $\tilde{\kappa}'$ are independently and uniformly distributed except when $\tau = 0$, which happens with probability $1/q$. As a result, the distributions when $\eta = 0$ and when $\eta = 1$ are equivalent, except with probability $1/q$. □

Lemma 8. *For any PPT distinguisher $\mathcal{D}$ for Experiment* 3-2*, there is a PPT adversary $\mathcal{A}_{bp2^*}$ for Basic Problem 2* such that for any security parameter $\lambda \in \mathbb{N}$,*

$$\mathsf{Adv}_{\mathcal{D}}^{\exp 3\text{-}2}(\lambda) \leq \mathsf{Adv}^{bp2^*}_{\mathcal{A}_{bp2^*}}(\lambda) + \frac{1}{q}$$

Proof. Suppose we have a PPT distinguisher $\mathcal{D}$ for Experiment 3-2, then we can build a PPT adversary $\mathcal{A}_{\mathsf{bp2}^*}$ for Basic Problem 2*. On receiving a Basic Problem 2* instance $(\mathsf{param}_V, \{\hat{\mathbb{B}}_\ell, \mathbb{B}^*_\ell, \{\boldsymbol{h}^{(\beta)}_{\ell,i}, \boldsymbol{g}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha})$, $\mathcal{A}_{\mathsf{bp2}^*}$ sets, for $1 \leq \ell \leq \alpha$,

$$\begin{aligned}\mathbb{D}_\ell &= (\boldsymbol{b}_{\ell,0}, \boldsymbol{b}_{\ell,2n/\alpha+1}, \cdots, \boldsymbol{b}_{\ell,3n/\alpha}, \boldsymbol{b}_{\ell,n/\alpha+1}, \cdots, \boldsymbol{b}_{\ell,2n/\alpha}, \boldsymbol{b}_{\ell,1}, \cdots, \boldsymbol{b}_{\ell,n/\alpha}, \boldsymbol{b}_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{b}_{\ell,N-1})\\ \hat{\mathbb{D}}_\ell &= (\boldsymbol{b}_{\ell,0}, \boldsymbol{b}_{\ell,2n/\alpha+1}, \cdots, \boldsymbol{b}_{\ell,3n/\alpha}, \boldsymbol{b}_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{b}_{\ell,N-1})\end{aligned}$$

and

$$\begin{aligned}\mathbb{D}^*_\ell &= (\boldsymbol{b}^*_{\ell,0}, \boldsymbol{b}^*_{\ell,2n/\alpha+1}, \cdots, \boldsymbol{b}^*_{\ell,3n/\alpha}, \boldsymbol{b}^*_{\ell,n/\alpha+1}, \cdots, \boldsymbol{b}^*_{\ell,2n/\alpha}, \boldsymbol{b}^*_{\ell,1}, \cdots, \boldsymbol{b}^*_{\ell,n/\alpha}, \boldsymbol{b}^*_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{b}^*_{\ell,N-1})\\ \hat{\mathbb{D}}^*_\ell &= (\boldsymbol{b}^*_{\ell,0}, \boldsymbol{b}^*_{\ell,2n/\alpha+1}, \cdots, \boldsymbol{b}^*_{\ell,3n/\alpha}, \boldsymbol{b}^*_{\ell,3n/\alpha+1}, \cdots, \boldsymbol{b}^*_{\ell,N-1})\end{aligned}$$

Then $\mathcal{A}_{\mathsf{bp2}^*}$ samples $\eta_1, \eta_2 \xleftarrow{\$} \mathbb{F}_q$ and sets

$$\boldsymbol{f}_{\ell,i} = \eta_1 \boldsymbol{b}_{\ell,i} + \eta_2 \vec{e}_i, \text{ for } 1 \leq i \leq n/\alpha$$

$\mathcal{A}_{\mathsf{bp2}^*}$ sends

$$(\mathsf{param}_{\mathbb{V}}, \{\hat{D}_\ell, \hat{D}^*_\ell, \{\boldsymbol{h}^{(\beta)}_{\ell,i}, \boldsymbol{g}_{\ell,i}, \boldsymbol{f}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha})$$

to $\mathcal{D}$ and receives back $\beta' \in \{0,1\}$. $\mathcal{A}_{\mathsf{bp2}^*}$ outputs β'. Thus,

$$\begin{aligned}
\boldsymbol{h}^{(0)}_{\ell,i} &= (0, \delta\vec{e}_i, 0^{n/\alpha}, 0^{n/\alpha}, \delta_0\vec{e}_i, 0)_{\mathbb{B}^*_\ell} \\
&= (0, 0^{n/\alpha}, 0^{n/\alpha}, \delta\vec{e}_i, \delta_0\vec{e}_i, 0)_{\mathbb{D}^*_\ell} \\
\boldsymbol{h}^{(1)}_{\ell,i} &= (0, \delta\vec{e}_i, \tau\vec{e}_i, 0^{n/\alpha}, \delta_0\vec{e}_i, 0)_{\mathbb{B}^*_\ell} \\
&= (0, 0^{n/\alpha}, \tau\vec{e}_i, \delta\vec{e}_i, \delta_0\vec{e}_i, 0)_{\mathbb{D}^*_\ell} \\
\boldsymbol{g}_{\ell,i} &= (0, \omega\vec{e}_i, \sigma\vec{e}_i, 0^{n/\alpha}, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell} \\
&= (0, 0^{n/\alpha}, \sigma\vec{e}_i, \omega\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{D}_\ell} \\
\boldsymbol{f}_{\ell,i} &= (0, (\eta_1 + \eta_2\omega)\vec{e}_i, \eta_2\sigma\vec{e}_i, 0^{n/\alpha}, 0^{n/\alpha}, 0)_{\mathbb{B}_\ell} \\
&= (0, 0^{n/\alpha}, \eta_2\sigma\vec{e}_i, (\eta_1 + \eta_2\omega)\vec{e}_i, 0^{n/\alpha}, 0)_{\mathbb{D}_\ell}.
\end{aligned}$$

Since $\delta, \tau, \omega, \sigma, \eta_1$ and η_2 are independently and uniformly sampled, then $\delta, \tau, \omega, \sigma, \eta_1 + \eta_2\omega$ and $\eta_2\sigma$ are independently and uniformly distributed in $\mathbb{F}_q$ except when $\sigma = 0$, which happens with probability $1/q$. As a result, the distributions of $(\mathsf{param}_{\mathbb{V}}, \{\hat{D}_\ell, \hat{D}^*_\ell, \{\boldsymbol{h}^{(\beta)}_{\ell,i}, \boldsymbol{g}_{\ell,i}, \boldsymbol{f}_{\ell,i}\}_{i=1,\cdots,n/\alpha}\}_{\ell=1,\cdots,\alpha})$ and of the output of $\mathcal{G}^{\exp\,3\text{-}2}_\beta$ are equivalent except with probability $1/q$. □

Then from Lemmas 7, 8 and 5, for any PPT adversary $\mathcal{A}_{\mathsf{bp3}^*}$ there exists PPT adversaries, $\mathcal{A}_{\mathsf{bp2}^*}$ and $\mathcal{A}_{\mathsf{DLIN}^*}$, such that for any security parameter $\lambda \in N$ we have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{bp3}^*}_{\mathcal{A}_{\mathsf{bp3}^*}}(\lambda) &\leq \left| \Pr[\mathcal{A}_{\mathsf{bp3}^*}(1^\lambda, \mathcal{G}^{\exp\,3\text{-}1}_0(1^\lambda, n, \alpha)) = 1] - \Pr[\mathcal{A}_{\mathsf{bp3}^*}(1^\lambda, \mathcal{G}^{\exp\,3\text{-}2}_2(1^\lambda, n, \alpha)) = 1] \right| \\
&\leq \left| \Pr[\mathcal{A}_{\mathsf{bp3}^*}(1^\lambda, \mathcal{G}^{\exp\,3\text{-}1}_0(1^\lambda, n, \alpha)) = 1] - \Pr[\mathcal{A}_{\mathsf{bp3}^*}(1^\lambda, \mathcal{G}^{\exp\,3\text{-}1}_1(1^\lambda, n, \alpha)) = 1] \right| \\
&+ \left| \Pr[\mathcal{A}_{\mathsf{bp3}^*}(1^\lambda, \mathcal{G}^{\exp\,3\text{-}2}_1(1^\lambda, n, \alpha)) = 1] - \Pr[\mathcal{A}_{\mathsf{bp3}^*}(1^\lambda, \mathcal{G}^{\exp\,3\text{-}2}_2(1^\lambda, n, \alpha)) = 1] \right| \\
&\leq \mathsf{Adv}^{\exp\,3\text{-}1}_{\mathcal{A}_{\mathsf{bp3}^*}}(\lambda) + \mathsf{Adv}^{\exp\,3\text{-}2}_{\mathcal{A}_{\mathsf{bp3}^*}}(\lambda) \leq \mathsf{Adv}^{\mathsf{bp2}^*}_{\mathcal{A}_{\mathsf{bp2}^*}(\lambda)} + \frac{2}{q} \leq \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A}_{\mathsf{DLIN}}(\lambda)} + \frac{7}{q}
\end{aligned}$$

This completes the proof of Lemma 6. □

This completes the proof of Lemma 2. □