New Lower Bounds on Predicate Entropy for Function Private Public-Key Predicate Encryption

Sikhar Patranabis and Debdeep Mukhopadhyay

Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
sikhar.patranabis@iitkgp.ac.in, debdeep@cse.iitkgp.ernet.in

Abstract. We present function private public-key predicate encryption schemes from standard cryptographic assumptions, that achieve new lower bounds on the min-entropy of underlying predicate distributions. Existing function private predicate encryption constructions in the public-key setting can be divided into two broad categories. The first category of constructions are based on standard assumptions, but impose highly stringent requirements on the min-entropy of predicate distributions, thereby limiting their applicability in the context of real-world predicates. For example, the statistically function private constructions of Boneh, Raghunathan and Segev (CRYPTO'13 and ASIACRYPT'13) are inherently restricted to predicate distributions with min-entropy roughly proportional to the security parameter λ. The second category of constructions mandate more relaxed min-entropy requirements, but are either based on non-standard assumptions (such as indistinguishability obfuscation) or are secure in the generic group model. In this paper, we affirmatively bridge the gap between these categories by presenting new public-key constructions for identity-based encryption, hidden-vector encryption, and subspace-membership encryption (a generalization of inner-product encryption) that are both data and function private under variants of the well-known DBDH, DLIN and matrix DDH assumptions, while relaxing the min-entropy requirement on the predicate distributions to $\omega(\log \lambda)$. In summary, we establish that the minimum predicate entropy necessary for any meaningful notion of function privacy in the public-key setting, is in fact, sufficient, for a fairly rich class of predicates.

Keywords: Predicate Encryption, Public-Key, Function Privacy, Computational Indistinguishability, Min-Entropy, Identity-Based Encryption, Hidden-Vector Encryption, Inner-Product Encryption, Subspace-Membership Encryption

1 Introduction

Predicate encryption schemes [1–3] in the public-key setting allow a single public-key to be associated with multiple secret-keys, where each secret-key corresponds to a Boolean predicate $f : \Sigma \longrightarrow \{0,1\}$ over a pre-defined set of attributes Σ. A plaintext message in a predicate encryption scheme is an attribute-payload message pair $(I, M) \in \Sigma \times \mathcal{M}$, with $\mathcal{M}$ being the payload message space. A secret-key sk_f

associated with a predicate f successfully decrypts a ciphertext C corresponding to a plaintext (I, M) and recovers the payload message M if and only if $f(I) = 1$. On the other hand, if $f(I) = 0$, attempting to decrypt C using sk_f returns $\perp$.

Data and Function Privacy of Predicate Encryption. Suppose a probabilistic polynomial-time adversary against a predicate encryption scheme receives a ciphertext C corresponding to an attribute I and a payload message M. In addition, suppose that the adversary also receives secret-keys $\mathsf{sk}_1, \cdots, \mathsf{sk}_n$ corresponding to predicates $f_1, \cdots, f_n$, subject to the restriction that $f_j(I) = 0$ for each $j \in [1, n]$. Informally, a predicate encryption scheme is *data private* if it guarantees that the adversary learns nothing beyond the absolute minimum about both the attribute I and the message M from the ensemble $\big(C, \{\mathsf{sk}_j\}_{j\in[1,n]}\big)$. Additionally, the predicate encryption scheme is *function private* if it also guarantees that the secret-keys $\mathsf{sk}_1, \cdots, \mathsf{sk}_n$ reveal no information beyond the absolute minimum about the underlying predicates $f_1, \cdots, f_n$. Quite evidently, the notions of data and function privacy for a predicate encryption scheme are independent in the sense that one does not necessarily imply the other.

1.1 Motivation for Function Private Predicate Encryption

Predicate encryption provides a generic framework for searchable encryption supporting a wide range of query predicates including conjunctive, disjunctive, range and subset queries [4, 1–3], which makes it a highly desirable cryptographic primitive for real-life applications. In this section, we present some real-life applications of predicate encryption that require function privacy to be treated as an essential cryptographic property in addition to data privacy:

- **Secure Information Retrieval.** To design a secure information retrieval system for an organization such as a bank, one can encrypt confidential data objects such as customer details, account information and transaction logs, that are labeled with a set of attributes such as customer age, account types, or geographical location. Members of the organization may be issued restricted secret-keys that can only decrypt data objects that satisfy a specific range of values for customer age, or conform to a particular account type, or alternatively, fall under a specific geographical location. In such a setting, issuing secret-keys that do not hide the underlying predicate could allow even an unauthorized member to infer sensitive customer information. For example, consider a secret-key that allows decrypting account information for all customers aged above X years. If the holder of the secret-key knows X, she can immediately infer some information about the age of each customer simply from decryption success/failure. Similar situations occur when designing secure information retrieval systems for law enforcement and healthcare applications.
- **Secure Mail Gateway.** Predicate encryption can be used to realize a secure mail gateway that follows some special instructions to route encrypted mails based on the sender id (e.g. if the mail is from the boss/spouse and needs to routed to the "urgent" folder). The sender id acts as the attribute, while the

routing instructions act as the payload message. The need for function privacy in such a scenario is evident: secret-keys that do not hide the underlying predicate (such as "is-urgent"), potentially leak information about both the sender id and the content of the mail simply from decryption success/failure.

- **Secure Payment Gateway.** Predicate encryption can be used to realize a secure payment gateway that flags encrypted payments if they correspond to amounts beyond some pre-defined threshold X. The payment gateway is given the secret-key corresponding to the predicate "greater-than-X", the payment amount itself serves as the attribute, while the flag signal is encoded as the payload message. To see why function privacy is desirable in such a scenario, observe that if the gateway knows the predicate "greater-than-X", it can infer some information about each transaction amount simply from the flag/no-flag outcome.

1.2 Function Private Predicate Encryption in the Public-Key Setting

As pointed out by Boneh, Raghunathan and Segev in [5, 6], formalizing a realistic notion of function privacy in the context of public-key predicate encryption is, in general, not straightforward. Consider, for example, an adversary against an IBE scheme who is given a secret-key $\mathsf{sk}_{\mathsf{id}}$ corresponding to an identity id and has access to an encryption oracle. As long as the adversary has some apriori information that the identity id belongs to a set $\mathcal{S}$ such that $|\mathcal{S}|$ is at most polynomial in the security parameter λ, it can fully recover id from $\mathsf{sk}_{\mathsf{id}}$: it can simply resort to encrypting a random message M under each identity in $\mathcal{S}$, and decrypting using $\mathsf{sk}_{\mathsf{id}}$ to check for a correct recovery. Consequently, Boneh, Raghunathan and Segev [5, 6] consider a framework for function privacy under the minimal assumption that any predicate is sampled from a distribution with min-entropy at least super logarithmic in the security parameter λ. This rules out trivial attacks and results in a meaningful notion of function privacy in the public-key setting. However, their work leaves open the following important issues, which we address in this paper:

- The predicate encryption schemes proposed in [5, 6] are inherently restricted to satisfying a *statistical* notion of function privacy against computationally unbounded adversaries. For a vast majority of applications, a *computational* notion of function privacy against probabilistic polynomial-time adversaries suffices. It is currently an open problem to design public-key predicate encryption schemes whose function privacy can be based on standard computational assumptions.

- The function private constructions in [5, 6] assume predicate distributions with min-entropy $k \geq \lambda$ (where λ is the security parameter) to ensure the statistical indistinguishability of secret-keys against unbounded adversaries. This rather stringent assumption limits the applicability of their constructions in the context of real-world predicates. An interesting open problem is to explore whether, in a computational setting, the function privacy guarantees of a public-key predicate encryption scheme can hold under the minimal assumption that the predicates are sampled from a distribution with min-entropy $k = \omega(\log \lambda)$.

Other function private predicate encryption constructions in the public-key setting that do not impose stringent requirements on the min-entropy of the underlying predicate distribution are secure under non-standard assumptions: examples of such constructions include the function private IPE construction of Agrawal et al. [7] that is secure in the generic group model, and the function private constructions proposed by Iovino et al. in [8] that assume quasi-strong indistinguishability obfuscation. Highly expressive predicate encryption schemes that can support all poly-depth bounded circuits [9–13] from standard assumptions such as LWE are trivially non-function private, since they inherently assume that the secret-key contains the predicate circuit in the clear.

1.3 Our Contributions

We address the following open problem in this paper:

Is it possible to design public-key predicate encryption schemes that are provably data and function private under standard computational assumptions, while imposing the bare-minimum min-entropy requirements on the underlying predicate distributions?

We answer these questions in the affirmative by presenting new public-key constructions for identity-based encryption, subspace-membership encryption (a generalization of inner-product encryption) and hidden vector encryption, that are both data and function private under variants of the well-known DBDH, DLIN and matrix DDH assumptions, while relaxing the min-entropy requirement on the predicate distributions to $\omega(\log \lambda)$. Our constructions concretely establish that the minimum predicate entropy necessary for any meaningful notion of function privacy in the public-key setting, is in fact, sufficient, for a fairly rich class of predicates. Our results may be summarized in the form Table 1. Note that achieving function private HVE in any setting - computational or statistical, was open prior to this work. In particular, function private constructions for inner-product encryption and subspace-membership encryption do not naturally subsume hidden-vector encryption due to the presence of a special wildcard character in the predicate vectors for the latter. Function private HVE paves the way for realizing function private searchable encryption schemes supporting conjunctive, subset and range queries over encrypted data.

Table 1. Our Contributions

Predicate Encryption Constructions	**Predicates**	**Security**	**Predicate Entropy k**
Boneh, Raghunathan, Segev[5, 6]	Equality, Subspace-membership	DBDH,DLIN,LWE	$k \geq \lambda$
Agrawal et al.[7]	Inner-products	Non-standard assumptions	$k = \omega(\log \lambda)$
Iovion et al.[8]	All circuits	Obfuscation	$k = \omega(\log \lambda)$
Our Constructions	**Equality, Subspace-membership**	**DBDH,DLIN,MDDH**	$\boldsymbol{k = \omega(\log \lambda)}$
Our Constructions	**Hidden-vector**	**MDDH**	$\boldsymbol{k = \omega(\log \lambda)}$

Core Idea and Differences with Leakage Resilience. The core idea underlying our public-key function private predicate encryption constructions is to embed a randomly sampled instance of a computationally hard problem in each secret-key. Such an approach has previously been explored in *dual system encryption*, introduced by Waters in [14], and subsequently studied in [15–17]. Lewko et al. [18] demonstrated an elegant application of dual system encryption in achieving leakage resilient functional encryption schemes with *nominally semi-functional* secret-keys. However, function privacy is inherently more complex to achieve than leakage resilience. This may be intuitively explained as follows: for leakage resilience, it is sufficient to argue that a secret-key in the *semi-functional* space retains enough leftover entropy so as to hide whether the underlying predicate is orthogonal to the challenge vector (the case of nominal semi-functionality) or is uniformly random (the case of ordinary semi-functionality). However, for function privacy, the argument must additionally be *reflected back* from the *semi-functional space* to the *normal/fully functional* space, which is not immediately evident from existing dual system encryption based techniques. Our techniques, on the other hand, achieve function privacy by directly arguing the indistinguishability of secret-keys in the fully functional space. More specifically, the fully functional secret-keys in our constructions hide (in a computational sense) whether the underlying predicate has been sampled from a (sufficiently random) aversarially chosen distribution, or a uniformly random distribution. Our techniques are further elucidated in the following discussion.

1.4 Overview of Our Techniques

We briefly summarize our techniques below. A common technique implicit in the function privacy arguments for all our constructions in this paper is the use of hash proof systems [19, 20].

Function-Private IBE from DBDH and DLIN (Section 4). Our function private IBE construction is inspired by the seminal IBE scheme of Boneh and Franklin [21]. It involves public parameters $\mathsf{pp} = (g, g^{s_1}, g^{s_2})$, where g generates a bilinear group $\mathbb{G}$ of prime order q, and the master secret key is $\mathsf{msk} = (s_1, s_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$. The scheme uses two hash functions H_1 and H_2, that are modeled as random oracles in the security proofs. The secret-key $\mathsf{sk}_{\mathsf{id}}$ corresponding to an identity id is randomized as $\left(H_1(\mathsf{id})^{s_1 \cdot z_1} \cdot H_2(\mathsf{id})^{s_2 \cdot z_2}, z_1, z_2\right)$ for $z_1, z_2 \xleftarrow{R} \mathbb{Z}_q$, while a ciphertext C corresponding to (id, M) is generated as $\left(g^r, M \cdot e\left(g^{s_1}, H_1(\mathsf{id})\right)^r, e\left(g^{s_2}, H_2(\mathsf{id})\right)^r\right)$ for $r \xleftarrow{R} \mathbb{Z}_q$. The decryption procedure is essentially similar to that in the Boneh-Franklin IBE scheme. Note that both the secret-key and the ciphertext comprise of a constant number of group elements. We establish the data privacy of this scheme from the DBDH assumption via a sequence of two hybrid experiments, each of which manipulate the random oracles in the same way as [21] to answer secret-key and challenge ciphertext queries. Additionally, we establish the function privacy of this scheme using arguments akin to those of Cramer and Shoup [19], where the reduction knows the master-secret-key at all times (allowing it to answer any number of secret key-queries), and embeds a random DLIN instance in the challenge secret-key provided to the function privacy adversary. The proof suitably manipulates the random oracles, and also exploits the min-entropy restrictions on the challenge identity-distributions to argue the negligibility of abortion-probability during the reduction.

Function-Private SME from Matrix-DDH (Section 5). Our function private subspace-membership encryption (SME) scheme is inspired by a matrix-DDH-based variant of the recently proposed adaptively data private IPE of Agrawal, Libert and Stehlé [12]. It involves public parameters of the form $\mathsf{pp} = \left(g_1, g_1^{\mathbf{A}}, \left\{g_1^{\mathbf{S}_j \cdot \mathbf{A}}\right\}_{j \in [0,2n]}\right)$, where g_1 generates an asymmetric bilinear group $\mathbb{G}_1$ of prime order q, $\mathbf{A} \in \mathbb{Z}_q^{l_1 \times l_2}$, $\mathbf{S}_j \in \mathbb{Z}_q^{l_2 \times l_1}$, and the master-secret-key is $\mathsf{msk} = \left(g_2, \{\mathbf{S}_j\}_{j \in [0,2n]}\right)$, where g_2 generates a second asymmetric bilinear group $\mathbb{G}_2$ of the same order q. Note that the parameters l_1 and l_2 may be chosen to be constant, so long as $l_1 > l_2$, while the maximum number of secret-key queries supported for either the data or the function privacy experiment is n. To generate a secret-key for a matrix $\mathbf{W} \in \mathbb{Z}_q^{m \times n}$, the key-generation algorithm first adds n randomly sampled columns to $\mathbf{W}$, and outputs $\mathsf{sk}_{\mathbf{W}} = \left(g_2^{\mathbf{y}^{\mathbf{T}} \cdot \mathbf{W}}, g_2^{\sum_{j=1}^{2n} \mathbf{y}^{\mathbf{T}} \cdot \mathbf{W}_j \cdot \mathbf{S}_j}\right)$, where $\mathbf{W}_j$ denotes the j^{th} column of the augmented matrix $\mathbf{W}$. The ciphertext for an attribute vector $\mathbf{x} = \begin{bmatrix} x_1 \ x_2 \cdots x_n \end{bmatrix}^{\mathbf{T}} \in \mathbb{Z}_q^n$ is of the form $\left(g_1^{\mathbf{A} \cdot \mathbf{r}}, \{g_1^{(x_j \cdot \mathbf{S}_0 + \mathbf{S}_j) \cdot \mathbf{A} \cdot \mathbf{r}}\}_{j \in [1,2n]}\right)$ for $\mathbf{r} \xleftarrow{R} \mathbb{Z}_q^{l_2}$ and $x_{n+1} = x_{n+2} = \cdots = x_{2n} = 0$. The analysis of data privacy relies on the MDDH assumption in the group $\mathbb{G}_1$, and uses hash-proof based arguments similar to those of Cramer and Shoup [19, 20]. Note that the additional attribute vector components $x_{n+1}, \cdots, x_{2n}$ in our SME construction are 0, implying that the final n ciphertext components are statistically independent of the vector $\mathbf{x}$ being encrypted. The analysis now exploits the following fact: so long as the adversary makes no more that n secret-key queries, the first n master-secret-key components $(\mathbf{S}_0, \cdots, \mathbf{S}_n)$ retain sufficient entropy from an adversary's point of view, even given the public parameter pp and $\mathcal{B}$'s responses to n-many secret-key queries. This in turn ensures that at some stage, if the challenge ciphertext is generated using the master-secret-key instead of the public parameter, it will perfectly hide which attribute among $\mathbf{x}_0^*$ and $\mathbf{x}_1^*$ is encrypted. Finally, the scheme is adaptively secure because the reduction knows the master-secret-key at any time, which allows it to answer all secret-key queries without knowing the challenge attributes beforehand. The proof of function privacy also follows from the MDDH assumption, albeit in the group $\mathbb{G}_2$, and relies on the fact that any predicate matrix $\mathbf{W}$ with entries sampled from mutually independent distributions with super-logarithmic min-entropy has full rank n with overwhelmingly large probability.

Function-Private HVE from Matrix-DDH (Section 6). Our function-private hidden-vector encryption (HVE) scheme uses techniques similar to our function private SME construction, with slight differences to account for the presence of the wildcard character $\star$. The core idea is as follows: given a predicate vector $\mathbf{v} = (v_1, \cdots, v_n) \in (\mathbb{Z}_q \cup \{\star\})^n$, let $\mathcal{S} = \{j \in [1, n] : v_j \neq \star\}$. Also, let $\mathbf{v}_{\mathcal{S}} = \begin{bmatrix} v_j \end{bmatrix}_{j \in \mathcal{S}} \in \mathbb{Z}_q^{|\mathcal{S}|}$. Now, one can create a matrix $\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 \mid \mathbf{W}_1 \cdot \mathbf{v}_{\mathcal{S}} \end{bmatrix} \in \mathbb{Z}_q^{|\mathcal{S}| \times (|\mathcal{S}|+1)}$, where

$\mathbf{W}_1 \xleftarrow{R} \mathbb{Z}_q^{|\mathcal{S}|\times|\mathcal{S}|}$ is a random *invertible* matrix. Next, given an attribute vector $\mathbf{x} = \begin{bmatrix} x_1 \cdots x_n \end{bmatrix}^{\mathbf{T}} \in \mathbb{Z}_q^n$, let $\mathbf{x}_{\mathcal{S}} = \begin{bmatrix} x_j \end{bmatrix}_{j\in\mathcal{S}} \in \mathbb{Z}_q^{|\mathcal{S}|}$. If $v_j = x_j$ for each $j \in \mathcal{S}$, we must have $\mathbf{v}_{\mathcal{S}} = \mathbf{x}_{\mathcal{S}}$, and hence the following holds:

$$\mathbf{W} \cdot \begin{bmatrix} \mathbf{x}_{\mathcal{S}} \\ (-1) \end{bmatrix} = \begin{bmatrix} \mathbf{W}_1 \mid \mathbf{W}_1 \cdot \mathbf{v}_{\mathcal{S}} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}_{\mathcal{S}} \\ (-1) \end{bmatrix} = \mathbf{0} \in \mathbb{Z}_q^{|\mathcal{S}|}$$

The crux of our HVE construction lies in generating a ciphertext C for the vector $\begin{bmatrix} \mathbf{x} \mid (-1) \end{bmatrix}^{\mathbf{T}}$ using the encryption algorithm of our SME scheme such that, given a subset $\mathcal{S}$, C may be manipulated to generate a new ciphertext corresponding to $\begin{bmatrix} \mathbf{x}_{\mathcal{S}} \mid (-1) \end{bmatrix}^{\mathbf{T}}$. Then, given a secret-key corresponding to the matrix $\mathbf{W}$ generated using the key-generation algorithm of our SME construction, decryption is straightforward. The data and function privacy of our HVE construction follow from arguments akin to those used for our SME construction.

1.5 Notations Used

This section summarizes the notations used throughout the rest of the paper.

General Notations. We write $x \xleftarrow{R} \chi$ to represent that an element x is sampled uniformly at random from a set $\mathcal{X}$. The output a of a deterministic algorithm $\mathcal{A}$ is denoted by $x \leftarrow \mathcal{A}$ and the output a' of a randomized algorithm $\mathcal{A}'$ is denoted by $x' \xleftarrow{R} \mathcal{A}'$. We refer to $\lambda \in \mathbb{N}$ as the security parameter, and denote by $\mathsf{exp}(\lambda)$, $\mathsf{poly}(\lambda)$ and $\mathsf{negl}(\lambda)$ any generic (unspecified) exponential function, polynomial function and negligible function in λ respectively. Note that a function $f : \mathbb{N} \to \mathbb{N}$ is said to be negligible in λ if for every positive polynomial p, $f(\lambda) < 1/p(\lambda)$ when λ is sufficiently large. For $a, b \in \mathbb{Z}$ such that $a \leq b$, we denote by $[a, b]$ the set of integers lying between a and b (both inclusive). For a finite field $\mathbb{F}_q$ (q being a λ-bit prime) and $m, n \in \mathbb{N}$, we denote by $\mathbb{F}_q^{m\times n}$ the space of all $m \times n$ matrices $\mathbf{W}$ with elements from $\mathbb{F}_q$. Further, we use the short-hand notation $\mathbb{F}_q^m$ to represent the vector space $\mathbb{F}_q^{m\times 1}$. Finally, given a matrix $\mathbf{W} \in \mathbb{F}_q^{m\times n}$, its transpose in $\mathbb{F}_q^{n\times m}$ is denoted as $\mathbf{W}^{\mathbf{T}}$.

Bilinear Group Notations. Let $\mathsf{GroupGen}(1^\lambda)$ be a probabilistic polynomial-time algorithm that takes as input a security parameter λ, and outputs a tuple of the form $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, g_1, g_2, e)$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are distinct groups of order q (q being a λ-bit prime), g_1 is a generator for $\mathbb{G}_1$, g_2 is a generator for $\mathbb{G}_2$, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ is an efficiently computable non-degenerate asymmetric bilinear map. Now, given a matrix $\mathbf{W} = \{w_{i,j}\}_{i\in[1,m],j\in[1,n]} \in \mathbb{Z}_q^{m\times n}$, we use the following notations:

- $g_1^{\mathbf{W}}$: Denotes the set of group elements $\{g_1^{w_{i,j}}\}_{i\in[1,m],j\in[1,n]} \in \mathbb{G}_1^{m\times n}$
- $g_2^{\mathbf{W}}$: Denotes the set of group elements $\{g_2^{w_{i,j}}\}_{i\in[1,m],j\in[1,n]} \in \mathbb{G}_2^{m\times n}$
- $e(g_1, g_2)^{\mathbf{W}}$: Denotes the set of group elements $\{e(g_1, g_2)^{w_{i,j}}\}_{i\in[1,m],j\in[1,n]} \in \mathbb{G}_T^{m\times n}$

For ease of representation, we use the notations $e(g_1, g_2)^{\mathbf{W}_2 \cdot \mathbf{W}_1}$ and $e\left(g_1^{\mathbf{W}_1}, g_2^{\mathbf{W}_2}\right)$, interchangeably, throughout this paper. Finally, in the *symmetric* bilinear map setting, the groups $\mathbb{G}_1$ and $\mathbb{G}_2$, along with their corresponding generators g_1 and g_2, are denoted using the unified notations $\mathbb{G}$ and g, respectively,

Min-Entropy of Random Variables. The min-entropy of a random variable Y is called $\mathbf{H}_\infty(Y)$ and is evaluated as $-\log(\max_y \Pr[Y = y])$; a random variable Y is said to be a k-source if $\mathbf{H}_\infty(Y) \geq k$. We additionally define an (m, n, k)-*matrix-source* for $m, n \in \mathbb{N}$ to be a matrix of mutually independent random variables $\mathbf{Y} = \left(\{Y_{i,j}\}_{i \in [1,m], j \in [1,n]}\right)$, such that each individual $Y_{i,j}$ is uniformly distributed over some finite field $\mathbb{F}_{q_k}$, where q_k is a k-bit prime. It is easy to see that each such $Y_{i,j}$ represents a k-source.

2 Background and Preliminaries

In this section, we recall certain standard computational assumptions in bilinear groups, and also introduce certain *min-entropy* variants of these assumptions.

2.1 Standard Computational Assumptions in Bilinear Groups

The Decisional Bilinear Diffie-Hellman assumption (DBDH). Let $\mathsf{GroupGen}(1^\lambda)$ be a probabilistic polynomial-time algorithm that takes as input a security parameter λ, and outputs a tuple of the form $(\mathbb{G}, \mathbb{G}_T, q, g, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of order q (q being a λ-bit prime), g is a generator for $\mathbb{G}$ and $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ is an efficiently computable non-degenerate bilinear map. The decisional bilinear Diffie-Hellman assumption is that the distribution ensembles:

$$\{(g, g^{a_1}, g^{a_2}, g^{a_3}, e(g,g)^{a_1 \cdot a_2 \cdot a_3})\}_{a_1,a_2,a_3 \xleftarrow{R} \mathbb{Z}_q} \text{ and } \{(g, g^{a_1}, g^{a_2}, g^{a_3}, Z)\}_{a_1,a_2,a_3 \xleftarrow{R} \mathbb{Z}_q, Z \xleftarrow{R} \mathbb{G}_T}$$

are computationally indistinguishable, where $(\mathbb{G}, \mathbb{G}_T, q, g, e) \leftarrow \mathsf{GroupGen}(1^\lambda)$.

The Decisional Linear Assumption (DLIN). Let $\mathbb{G}$ be a group of prime order q and let g_1, g_2, g_3 be arbitrary generators for $\mathbb{G}$. The decisional linear assumption [22] is that the distribution ensembles:

$$\left\{\left(g_1, g_2, g_3, g_1^{a_1}, g_2^{a_2}, g_3^{a_1+a_2}\right)\right\}_{a_1,a_2 \xleftarrow{R} \mathbb{Z}_q} \text{ and } \{(g_1, g_2, g_3, g_1^{a_1}, g_2^{a_2}, g_3^{a_3})\}_{a_1,a_2,a_3 \xleftarrow{R} \mathbb{Z}_q}$$

are computationally indistinguishable, where $g_1, g_2, g_3 \xleftarrow{R} \mathbb{G}$. A generalized version of this assumption, denoted as k-DLIN, is presented in Appendix B.

The Matrix Decisional Diffie-Hellman Assumption (MDDH). Let $\mathbb{G}$ be a group of prime order q and let g be an arbitrary generator for $\mathbb{G}$. Also, let $m, n \in \mathbb{N}$ with $m > n$, and let $\mathcal{D}_{m,n}$ denote a matrix distribution from which one can efficiently sample with overwhelmingly large probability a matrix $\mathbf{W} \in \mathbb{Z}_q^{m \times n}$, such that $\mathbf{W}$ has full rank n. The $\mathcal{D}_{m,n}$-matrix decisional Diffie-Hellman assumption [23] is that the distribution ensembles:

$$\left\{\left(g^{\mathbf{W}}, g^{\mathbf{W}\cdot\mathbf{y}}\right)\right\}_{\mathbf{W}\xleftarrow{R}\mathcal{D}_{m,n}\ ,\ \mathbf{y}\xleftarrow{R}\mathbb{Z}_q^n} \text{ and } \left\{\left(g^{\mathbf{W}}, g^{\mathbf{u}}\right)\right\}_{\mathbf{W}\xleftarrow{R}\mathcal{D}_{m,n}\ ,\ \mathbf{u}\xleftarrow{R}\mathbb{Z}_q^m}$$

are computationally indistinguishable, where $g \xleftarrow{R} \mathbb{Z}_q$ and $m, n = \mathsf{poly}(\lambda)$.

The $\mathcal{D}_{m,n}$-MDDH assumption holds even if the group $\mathbb{G}$ is bilinear, albeit subject to the additional restriction that $n \geq 2$ [23]. This restriction may, however, be relaxed if the corresponding bilinear map is asymmetric over two distinct source groups $\mathbb{G}_1$ and $\mathbb{G}_2$, and the MDDH assumption is assumed to hold in either one or both these groups (analogous to the external Diffie-Hellman (XDH) and symmetric external Diffie-Hellman assumptions (SXDH) [24], respectively).

2.2 Min-Entropy-based Sub-Families of the MDDH Assumption

In this section, we introduce two min-entropy-based sub-families of the MDDH assumption. We first state the following claims:

Claim 2.1 *Let $\mathbf{V} = \left(\{V_{i,j}\}_{i\in[1,m],j\in[1,n]}\right)$ be an (m,n,k)-matrix-source over $\mathbb{Z}_q^{m\times n}$ for $k = \omega(\log\lambda)$. Then, $\mathbf{V}$ represents a matrix distribution from which one can efficiently sample with overwhelmingly large probability a matrix $\mathbf{W} \in \mathbb{Z}_q^{m\times n}$, such that $\mathbf{W}$ has full rank n.*

Claim 2.2 *Let $\mathbf{V}_1 = \left(\{V_{i,j,1}\}_{i\in[1,n],j\in[1,n]}\right)$ be an (n,n,k)-matrix-source over $\mathbb{Z}_q^{n\times n}$, such that $k = \omega(\log\lambda)$, and let $\mathbf{V}_2 = \left(\{V_{i,2}\}_{i\in[1,n]}\right)$ be any non-zero distribution over $\mathbb{Z}_q^n$. Then $\tilde{\mathbf{V}} = \left[\mathbf{V}_1 \mid \mathbf{V}_1 \cdot \mathbf{V}_2\right]^{\mathbf{T}} \in \mathbb{Z}_q^{(n+1)\times n}$ represents a matrix distribution from which one can efficiently sample with overwhelmingly large probability a matrix $\mathbf{W} \in \mathbb{Z}_q^{(n+1)\times n}$, such that $\mathbf{W}$ has full rank n.*

The detailed proofs of these claims are presented in Appendix C. The aforementioned claims allows us to define the following min-entropy-based sub-families of the MDDH assumption.

Definition 2.1 (The (m,n,k)-Source-MDDH Assumption). *Let $\mathbb{G}$ be a group of prime order q and let g be an arbitrary generator for $\mathbb{G}$. The* (m,n,k)-source-MDDH assumption, *for $m,n \in \mathbb{N}$ such that $m > n$ and $k = \omega(\log\lambda)$, is that for any probabilistic polynomial-time adversary $\mathcal{A}$, the following holds:*

$$\mathbf{Adv}^{\mathrm{MDDH}}_{m,n,k,\mathcal{A}}(\lambda) \stackrel{\text{def}}{=} \left|\Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{MDDH},m,n,k,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{MDDH},m,n,k,\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda)$$

where for each $\lambda \in \mathbb{N}$ and $b \in \{0,1\}$, the experiment $\mathsf{Expt}^{(b)}_{\mathrm{MDDH},m,n,k\mathcal{A}}(\lambda)$ is defined as:

1. *$(\mathbf{V}^*, \mathsf{state}) \xleftarrow{R} \mathcal{A}\left(1^\lambda, m, n, k\right)$, where $\mathbf{V}^* = \left(\left\{V^*_{i,j}\right\}_{i\in[1,m],j\in[1,n]}\right)$ is an (m,n,k)-matrix-source.*
2. *Sample $\mathbf{W} \xleftarrow{R} \mathbf{V}^*$.*
3. *If $b = 1$, sample $\mathbf{y} \xleftarrow{R} \mathbb{Z}_q^n$, and set $\mathbf{u} = \mathbf{W}\cdot\mathbf{y}$. Else if $b = 0$, sample $\mathbf{u} \xleftarrow{R} \mathbb{Z}_q^m$.*
4. *$b' \xleftarrow{R} \mathcal{A}\left(\left(g^{\mathbf{W}}, g^{\mathbf{u}}\right), \mathsf{state}\right)$.*
5. *Output b'.*

The (m,n,k)-Source-MDDH Assumption holds even if the group $\mathbb{G}$ is bilinear, subject to the additional restriction that $n \geq 2$.

Definition 2.2 (The (n,k)-Source-MDDH Assumption). *Let $\mathbb{G}$ be a group of prime order q and let g be an arbitrary generator for $\mathbb{G}$. The (n,k)-*source-MDDH assumption*, for $n \in \mathbb{N}$ and $k = \omega(\log\lambda)$, is that for any probabilistic polynomial-time adversary $\mathcal{A}$, the following holds:*

$$\mathbf{Adv}^{\mathrm{MDDH}}_{n,k,\mathcal{A}}(\lambda) \stackrel{\mathrm{def}}{=} \left|\Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{MDDH},n,k,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{MDDH},n,k,\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda)$$

where for each $\lambda \in \mathbb{N}$ and $b \in \{0,1\}$, the experiment $\mathsf{Expt}^{(b)}_{\mathrm{MDDH},n,k\mathcal{A}}(\lambda)$ is defined as:

1. *$(\mathbf{V}_1^*, \mathbf{V}_2^*, \mathsf{state}) \xleftarrow{R} \mathcal{A}\left(1^\lambda, n, k\right)$, where $\mathbf{V}_1^* = \left(\left\{V^*_{i,j,1}\right\}_{i\in[1,n],j\in[1,n]}\right)$ is an (n,n,k)-matrix-source and $\mathbf{V}_2^* = \left(\left\{V^*_{i,2}\right\}_{i\in[1,n]}\right)$ is a non-zero distribution over $\mathbb{Z}_q^n$.*
2. *Let $\tilde{\mathbf{V}}^* = \left[\mathbf{V}_1^* \mid \mathbf{V}_1^* \cdot \mathbf{V}_2^*\right]^{\mathbf{T}}$. Sample $\mathbf{W} \xleftarrow{R} \tilde{\mathbf{V}}^*$.*
3. *If $b = 1$, sample $\mathbf{y} \xleftarrow{R} \mathbb{Z}_q^n$, and set $\mathbf{u} = \mathbf{W}\cdot\mathbf{y}$. Else if $b = 0$, sample $\mathbf{u} \xleftarrow{R} \mathbb{Z}_q^m$.*
4. *$b' \xleftarrow{R} \mathcal{A}\left(\left(g^{\mathbf{W}}, g^{\mathbf{u}}\right), \mathsf{state}\right)$.*
5. *Output b'.*

Once again, the (n,k)-Source-MDDH Assumption holds even if the group $\mathbb{G}$ is bilinear, subject to the additional restriction that $n \geq 2$.

3 Function Privacy of Public-Key Predicate Encryption

In this section, we formally define the indistinguishability-based framework for function privacy of predicate encryption in the public-key setting. We consider adversaries that have access to the public parameters of the scheme, as well as a secret-key generation oracle. The adversary is additionally allowed to query a left-or-right function-privacy oracle $\mathsf{LoR}^{\mathsf{FP}}$. This oracle takes as input two adversarially-chosen distributions over the class of predicates $\mathcal{F}$, subject to certain min-entropy requirements, and outputs a secret-key for a predicate sampled from one of these distributions. At the end of the interaction, the adversary should be able to distinguish between the *left* and *right* modes of operation of $\mathsf{LoR}^{\mathsf{FP}}$ with only negligible probability. The formal definitions for function privacy are presented separately for three specific sub-classes of predicate encryption considered in this paper - namely, identity-based encryption (IBE), subspace-membership encryption (SME) and hidden-vector encryption (HVE). Note that the corresponding data privacy notions for each of these predicate encryption systems can be captured within a single indistinguishability-based framework (see Definition A.1 in Appendix A).

3.1 Identity-Based Encryption and its Function Privacy

An identity-based encryption scheme Π^{IBE} over an identity space $\mathcal{ID}$ and a message space $\mathcal{M}$ is a public-key predicate encryption scheme supporting the set of equality predicates $f_{\mathsf{id}} : \mathcal{ID} \longrightarrow \{0,1\}$ defined as $f_{\mathsf{id}}(\mathsf{id}') = 1$ if and only if $\mathsf{id}' = \mathsf{id}$. The secret-key associated with an identity $\mathsf{id} \in \mathcal{ID}$ is denoted as $\mathsf{sk}_{\mathsf{id}}$. We now define the function privacy notions for an IBE scheme.

Definition 3.1 (Left-or-Right Function Privacy Oracle for IBE). *A left-or-right function privacy oracle* $\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{IBE}}$ *takes as input a quadruplet* $(\mathsf{mode}, msk, \mathbf{ID}_0, \mathbf{ID}_1)$*, where* $\mathsf{mode} \in \{\mathsf{left}, \mathsf{right}\}$*, msk is the master-secret-key of the IBE scheme, and* $(\mathbf{ID}_0, \mathbf{ID}_1)$ *are circuits representing distributions over the identity space* $\mathcal{ID}$*. If* $\mathsf{mode} = \mathsf{left}$*, the oracle samples* $\mathsf{id} \xleftarrow{R} \mathbf{ID}_0$*, while if* $\mathsf{mode} = \mathsf{right}$*, it samples* $\mathsf{id} \xleftarrow{R} \mathbf{ID}_1$*. It then responds with* $\mathsf{sk}_{\mathsf{id}} \xleftarrow{R} \mathsf{KeyGen}\,(msk, \mathsf{id})$*.*

Definition 3.2 (Computationally Function Private IBE). *An IBE scheme* $\Pi_{\mathrm{IBE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be* computationally function private *if for any probabilistic polynomial-time adversary* $\mathcal{A}$*, the following holds:*

$$\mathbf{Adv}^{\mathsf{FP}}_{\Pi_{\mathrm{IBE}},\mathcal{A}}(\lambda) \overset{\mathrm{def}}{=} \left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi_{\mathrm{IBE}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi_{\mathrm{IBE}},\mathcal{A}}(\lambda) = 1\right] \right| \leq \mathsf{negl}(\lambda)$$

where for each $\lambda \in \mathbb{N}$ *and* $\mathsf{mode} \in \{\mathsf{left}, \mathsf{right}\}$*, the experiment* $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP},\Pi_{\mathrm{IBE}},\mathcal{A}}(\lambda)$ *is defined as follows:*

1. $(\mathsf{pp}, \mathsf{msk}) \xleftarrow{R} \mathsf{Setup}\left(1^\lambda\right)$.
2. $b \xleftarrow{R} \mathcal{A}^{\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{IBE}}(\mathsf{mode},\mathsf{msk},\cdot,\cdot),\mathsf{KeyGen}(\mathsf{msk},\cdot)}\left(1^\lambda, \mathsf{pp}\right)$.
3. *Output* b.

subject to the restriction that each query issued to $\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{IBE}}(\mathsf{mode}, \mathsf{msk}, \cdot, \cdot)$ *is of the form* $(\mathbf{ID}^*_0, \mathbf{ID}^*_1)$*, where both* $\mathbf{ID}^*_0$ *and* $\mathbf{ID}^*_1$ *represent* k*-sources such that* $k = \omega(\log \lambda)$*.*

3.2 Subspace-Membership Encryption and its Function Privacy

A subspace-membership encryption scheme Π^{SME} over an attribute space $\Sigma = \mathbb{F}_q^n$ (q being a λ-bit prime) and a payload message space $\mathcal{M}$ is a public-key predicate encryption scheme supporting the set of matrix predicates $f_{\mathbf{W}} : \mathbb{F}_q^n \longrightarrow \{0,1\}$, such that for $\mathbf{W} \in \mathbb{F}_q^{m\times n}$ and $\mathbf{x} \in \mathbb{F}_q^n$, we have $f_{\mathbf{W}}(\mathbf{x}) = 1$ if and only if $\mathbf{W} \cdot \mathbf{x} = \mathbf{0} \in \mathbb{F}_q^m$. The secret-key associated with a matrix $\mathbf{W}$ is denoted as $\mathsf{sk}_{\mathbf{W}}$. We now define the function privacy notions for an SME scheme.

Definition 3.3 (Left-or-Right Function Privacy Oracle for SME). *A left-or-right function privacy oracle* $\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{SME}}$ *takes as input a quadruplet* $(\mathsf{mode}, msk, \mathbf{V}_0, \mathbf{V}_1)$*, where* $\mathsf{mode} \in \{\mathsf{left}, \mathsf{right}\}$*, msk is the master-secret-key of the SME scheme, and* $(\mathbf{V}_0, \mathbf{V}_1)$ *are circuits representing joint distributions over* $\mathbb{F}_q^{m \times n}$*. If* $\mathsf{mode} = \mathsf{left}$*, the oracle samples* $\mathbf{W} \xleftarrow{R} \mathbf{V}_0$*, while if* $\mathsf{mode} = \mathsf{right}$*, it samples* $\mathbf{W} \xleftarrow{R} \mathbf{V}_1$*. It then responds with* $\mathsf{sk}_{\mathbf{W}} \xleftarrow{R} \mathsf{KeyGen}(msk, \mathbf{W})$*.*

Definition 3.4 (Computationally Function Private SME). *An SME scheme* $\Pi_{\mathrm{SME}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be* computationally function private *if for any probabilistic polynomial-time adversary* $\mathcal{A}$*, the following holds:*

$$\mathbf{Adv}^{\mathsf{FP}}_{\Pi_{\mathrm{SME}},\mathcal{A}}(\lambda) \overset{\mathrm{def}}{=} \left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi_{\mathrm{SME}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi_{\mathrm{SME}},\mathcal{A}}(\lambda) = 1\right] \right| \leq \mathsf{negl}(\lambda)$$

where for each $\lambda \in \mathbb{N}$ *and* $\mathsf{mode} \in \{\mathsf{left}, \mathsf{right}\}$*, the experiment* $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP},\Pi_{\mathrm{SME}},\mathcal{A}}(\lambda)$ *is defined as follows:*

1. $(\mathsf{pp}, \mathsf{msk}) \xleftarrow{R} \mathsf{Setup}\left(1^\lambda\right)$.
2. $b \xleftarrow{R} \mathcal{A}^{\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{SME}}(\mathsf{mode},\mathsf{msk},\cdot,\cdot),\mathsf{KeyGen}(\mathsf{msk},\cdot)}\left(1^\lambda, \mathsf{pp}\right)$.
3. *Output* b.

subject to the restriction that each query issued to $\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{SME}}(\mathsf{mode}, \mathsf{msk}, \cdot, \cdot)$ *is of the form* $(\mathbf{V}_0^*, \mathbf{V}_1^*)$*, where both* $\mathbf{V}_0^* = \left(\left\{V^*_{i,j,0}\right\}_{i\in[1,m],j\in[1,n]}\right)$ *and* $\mathbf{V}_1^* = \left(\left\{V^*_{i,j,1}\right\}_{i\in[1,m],j\in[1,n]}\right)$ *represent* (m,n,k)*-matrix-sources for* $k = \omega(\log \lambda)$*.*

Avoiding Arbitrary Correlations among the Elements of W. Note that Definition 3.4 essentially requires that a secret-key $\mathsf{sk}_{\mathbf{W}}$ reveals no unnecessary information about the predicate matrix $\mathbf{W}$ as long as the elements of $\mathbf{W}$ are sampled from mutually independent and sufficiently unpredictable distributions. This restriction is imposed to rule out arbitrary correlations among the elements of $\mathbf{W}$. Indeed, it is impossible to achieve any realistic notion of function privacy for subspace-membership encryption that allows such arbitrary correlations among the elements of a predicate matrix (see [6] for a more detailed explanation of this impossibility).

3.3 Hidden-Vector Encryption and its Function Privacy

A hidden-vector encryption scheme Π^{HVE} over an attribute space Σ, a special wildcard symbol $\star$, and a payload message space $\mathcal{M}$, is a public-key predicate encryption scheme supporting predicates of the form $f_{\mathbf{v}} : \Sigma \longrightarrow \{0,1\}$, such that for each $\mathbf{v} = (v_1, \cdots, v_n) \in (\Sigma \cup \{\star\})^n$, and each $\mathbf{x} = (x_1, \cdots, x_n) \in \Sigma^n$, we have $f_{\mathbf{v}}(\mathbf{x}) = 1$ if and only if for each $j \in [1,n]$, either $v_j = x_j$ or $v_j = \star$. The secret-key associated with a predicate vector $\mathbf{v}$ is denoted as $\mathsf{sk}_{\mathbf{v}}$. Although SME subsumes HVE, the presence of the wildcard character in the predicate vector implies that the function privacy definitions for SME do not naturally apply to HVE [3]. This necessitates separate definitions for the function privacy of an HVE scheme.

Definition 3.5 (Left-or-Right Function Privacy Oracle for HVE). *A left-or-right function privacy oracle* $\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{HVE}}$ *takes as input a quintuplet* $(\mathsf{mode}, msk, \mathbf{V}_0, \mathbf{V}_1, \mathcal{S})$, *where* $\mathsf{mode} \in \{\mathsf{left}, \mathsf{right}\}$, *msk is the master-secret-key of the HVE scheme,* $(\mathbf{V}_0, \mathbf{V}_1)$ *are circuits representing joint distributions over* Σ^n, *and* $\mathcal{S} \subseteq [1, n]$. *If* $\mathsf{mode} = \mathsf{left}$, *the oracle samples* $\mathbf{v} \overset{R}{\longleftarrow} \mathbf{V}_0$, *while if* $\mathsf{mode} = \mathsf{right}$, *it samples* $\mathbf{v} \overset{R}{\longleftarrow} \mathbf{V}_1$. *For such a* $\mathbf{v} = (v_1, \cdots, v_n)$, *the oracle constructs a second* $\mathbf{v}' = (v'_1, \cdots, v'_n)$ *such that* $v'_j = v_j$ *if* $j \in \mathcal{S}$, *and* $v'_j = \star$, *otherwise. It then responds with* $\mathsf{sk}_{\mathbf{v}'} \overset{R}{\longleftarrow} \mathsf{KeyGen}(msk, \mathbf{v}')$.

Definition 3.6 (Computationally Function Private HVE). *An HVE scheme* $\Pi_{\mathrm{HVE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be* computationally function private *if for any probabilistic polynomial-time adversary* $\mathcal{A}$, *the following holds:*

$$\mathbf{Adv}^{\mathsf{FP}}_{\Pi_{\mathrm{HVE}},\mathcal{A}}(\lambda) \overset{\mathrm{def}}{=} \left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi_{\mathrm{HVE}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi_{\mathrm{HVE}},\mathcal{A}}(\lambda) = 1\right] \right| \le \mathsf{negl}(\lambda)$$

where for each $\lambda \in \mathbb{N}$ *and* $\mathsf{mode} \in \{\mathsf{left}, \mathsf{right}\}$, *the experiment* $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP},\Pi_{\mathrm{HVE}},\mathcal{A}}(\lambda)$ *is defined as follows:*

1. $(\mathsf{pp}, \mathsf{msk}) \overset{R}{\longleftarrow} \mathsf{Setup}\left(1^\lambda\right)$.
2. $b \overset{R}{\longleftarrow} \mathcal{A}^{\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{HVE}}(\mathsf{mode},\mathsf{msk},\cdot,\cdot,\cdot),\mathsf{KeyGen}(\mathsf{msk},\cdot)}\left(1^\lambda, \mathsf{pp}\right)$.
3. *Output* b.

subject to the restriction that each query issued to $\mathsf{LoR}^{\mathsf{FP}}_{\mathrm{HVE}}(\mathsf{mode}, \mathsf{msk}, \cdot, \cdot, \cdot)$ *is of the form* $(\mathbf{V}^*_0, \mathbf{V}^*_1, \mathcal{S}^*)$, *where both* $\mathbf{V}^*_0 = \left(\left\{V^*_{j,0}\right\}_{j\in[1,n]}\right)$ *and* $\mathbf{V}^*_1 = \left(\left\{V^*_{j,1}\right\}_{j\in[1,n]}\right)$ *represent* $(1, n, k)$*-matrix-sources for* $k = \omega(\log\lambda)$, *and* $\mathcal{S}^* \subseteq [1, n]$.

Note that our definitions for function private HVE essentially require that a secret-key $\mathsf{sk}_{\mathbf{v}}$ reveals no more information about $\mathbf{v}$ than the location of the wildcard characters, subject to the restriction that the remaining components of $\mathbf{v}$ are sampled from sufficiently unpredictable distributions. This *trivial leakage* induced by the presence of wildcard characters is not captured by our function privacy definitions for SME in general, and necessitates separate function privacy definitions for HVE.

3.4 Multi-Shot vs. Single-Shot Function Privacy Adversaries

Note that Definitions 3.2 and 3.4 consider function privacy adversaries that query the left-or-right function privacy oracle for any polynomial number of times. In fact, as adversaries are also given access to the key-generation oracle, this *multi-shot* definition is polynomially equivalent to its *single-shot* variant in which adversaries query the function privacy oracle at most once. This equivalence may be proved by a hybrid argument (originally proposed by Boneh, Raghunathan and Segev [5, 6]), where the hybrids are constructed such that only one query is forwarded to the function privacy oracle, and all other queries are answered using the key-generation oracle.

4 Computationally Function Private Identity-Based Encryption

In this section we present an IBE scheme based on the DBDH and DLIN assumptions in the random-oracle model. The scheme is inspired by the IBE of Boneh and Franklin [21]. The scheme is described below, and its proofs of data privacy and function privacy are presented subsequently.

4.1 The Scheme

Let $\mathsf{GroupGen}(1^\lambda)$ be a probabilistic polynomial-time algorithm that takes as input a security parameter λ, and outputs the tuple $(\mathbb{G}, \mathbb{G}_T, q, g, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of of order q (q being a λ-bit prime), g is a generator for $\mathbb{G}$ and $e : \mathbb{G}\times\mathbb{G} \longrightarrow \mathbb{G}_T$ is an efficiently computable non-degenerate *symmetric* bilinear map. The scheme $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}}$ is parameterized by the security parameter $\lambda \in \mathbb{N}$. For any such λ, we denote by $\mathcal{ID}_\lambda$ and $\mathcal{M}_\lambda$ the identity space and the message space, respectively. The scheme uses two hash functions $H_1 : \mathcal{ID}_\lambda \longrightarrow \mathbb{G}$ and $H_2 : \mathcal{ID}_\lambda \longrightarrow \mathbb{G}$ (modeled as random oracles).

- **Setup:** The setup algorithm samples $(\mathbb{G}, \mathbb{G}_T, q, g, e) \overset{R}{\longleftarrow} \mathsf{GroupGen}(1^\lambda)$ on input the security parameter 1^λ. It also samples $s_1, s_2 \overset{R}{\longleftarrow} \mathbb{Z}_q$, and outputs the public parameter pp and the master-secret-key msk as:
$$\mathsf{pp} = (g, g^{s_1}, g^{s_2}) \ , \ \mathsf{msk} = (s_1, s_2)$$
- **KeyGen:** On input the public parameter pp, the master-secret-key msk and an identity $\mathsf{id} \in \mathcal{ID}_\lambda$, the key generation algorithm samples $z_1, z_2 \overset{R}{\longleftarrow} \mathbb{Z}_q$ and outputs the secret-key $\mathsf{sk}_{\mathsf{id}} = (d_1, d_2, d_3)$ where:
$$d_1 = H_1\left(\mathsf{id}\right)^{s_1\cdot z_1} \cdot H_2\left(\mathsf{id}\right)^{s_2\cdot z_2} \ , \ d_2 = z_1 \ , \ d_3 = z_2$$
- **Enc:** On input the public parameter pp, an identity $\mathsf{id} \in \mathcal{ID}_\lambda$ and a message $M \in \mathcal{M}_\lambda$, the encryption algorithm samples $r \overset{R}{\longleftarrow} \mathbb{Z}_q$ and outputs the ciphertext $C = (c_1, c_2, c_3)$ where:
$$c_1 = g^r \ , \ c_2 = M \cdot e\left(g^{s_1}, H_1\left(\mathsf{id}\right)\right)^r \ , \ c_3 = e\left(g^{s_2}, H_2\left(\mathsf{id}\right)\right)^r$$
- **Dec:** On input a ciphertext $C = (c_1, c_2, c_3)$ and a secret-key $\mathsf{sk}_{\mathsf{id}} = (d_1, d_2, d_3)$, the decryption algorithm outputs:
$$M' = \left(\frac{c_2^{d_2} \cdot c_3^{d_3}}{e\left(d_1, c_1\right)}\right)^{1/d_2}$$

Correctness. Consider a ciphertext $C = (c_1, c_2, c_3)$ corresponding to a message M under an identity id, and a secret-key $\mathsf{sk}_{\mathsf{id}} = (d_1, d_2, d_3)$ corresponding to the same identity id. Then, we have:

$$\begin{aligned}
M' &= \left(\frac{c_2^{d_2} \cdot c_3^{d_3}}{e\left(d_1, c_1\right)} \right)^{1/d_2} \\
&= \left(\frac{M^{z_1} \cdot e\left(g^{s_1}, H_1\left(\mathsf{id}\right)\right)^{r \cdot z_1} \cdot e\left(g^{s_2}, H_2\left(\mathsf{id}\right)\right)^{r \cdot z_2}}{e\left(H_1\left(\mathsf{id}\right)^{s_1 \cdot z_1} \cdot H_2\left(\mathsf{id}\right)^{s_2 \cdot z_2}, g^r\right)} \right)^{1/z_1} \\
&= M \cdot \left(\frac{e\left(g^{s_1}, H_1\left(\mathsf{id}\right)\right)^{r \cdot z_1} \cdot e\left(g^{s_2}, H_2\left(\mathsf{id}\right)\right)^{r \cdot z_2}}{e\left(g^{r}, H_1\left(\mathsf{id}\right)\right)^{s_1 \cdot z_1} \cdot e\left(g^{r}, H_2\left(\mathsf{id}\right)\right)^{s_2 \cdot z_2}} \right)^{1/z_1} \\
&= M
\end{aligned}$$

Therefore as long as $z_1 \neq 0 \pmod q$ (an event which occurs with probability $1 - 1/q$ over the randomness of KeyGen), the message is recovered correctly.

4.2 Security of the Scheme

Adaptive Data Privacy. We state the following theorem for the adaptive data privacy of $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}}$:

Theorem 4.1 *Our IBE scheme $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}}$ is adaptively data private under the* DBDH *assumption in the random oracle model.*

Proof. The analysis of data privacy uses techniques very similar to those of Boneh and Franklin [21]. We define a sequence of three experiments, the first of which is identical to the standard data privacy experiment, the second experiment randomizes the second component of the challenge ciphertext provided to the adversary, while the final experiment randomizes both the second and third challenge ciphertext components. The indistinguishability of the first and second experiments is argued via a simulation where a simulator $\mathcal{B}$ embeds a DBDH instance in the second challenge ciphertext component, such that the component is well-formed with respect to the public parameters and a challenge identity-message pair if and only if the DBDH instance is valid. The indistinguishability of the second and third experiments follows from a similar simulation-based argument. The analysis models both the hash functions H_1 and H_2 as random oracles, and argues that the probability that either simulation aborts due to potential inconsistencies in either the secret-key-generation phase or the challenge ciphertext generation phase is negligible in the security parameter λ.

Computational Function Privacy. We state the following theorem for the computational function privacy of $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}}$:

Theorem 4.2 *Our IBE scheme $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}}$ is function private under the* DLIN *assumption for identities sampled uniformly from k-sources with $k = \omega\left(\log \lambda\right)$.*

Proof. The proof follows directly from the following claim:

Claim 4.1 *For any probabilistic polynomial-time adversary $\mathcal{A}$, the following holds:*

$$\left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] \right| \leq \mathsf{negl}(\lambda)$$

To prove this claim, we assume the contrary. Let $\mathcal{A}$ be a probabilistic polynomial-time adversary such that:

$$\left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] \right| = \epsilon$$

where $\epsilon > \mathsf{negl}(\lambda)$. We construct an algorithm $\mathcal{B}$ that solves an instance of the DLIN problem with advantage ϵ' negligibly close to ϵ. $\mathcal{B}$ receives as input a DLIN instance $(g_1, g_2, g_3, g_1^{a_1}, g_2^{a_2}, g_3^{a_3})$ over a bilinear group $\mathbb{G}$ with prime order q and generator g, and interacts with $\mathcal{A}$ as follows:

- ***Setup:*** $\mathcal{B}$ samples $x_1, x_2, x_3 \xleftarrow{R} \mathbb{Z}_q$ and provides $\mathcal{A}$ with the public parameter pp as:

$$\mathsf{pp} = (g, g_1^{x_1} \cdot g_3^{x_3}, g_2^{x_2} \cdot g_3^{x_3})$$

  where g_1,g_2 and g_3 are part of its input DLIN instance. Let $\alpha_j = \log_g g_j$ for $j \in \{1, 2, 3\}$. Then observe that $\mathcal{B}$'s choice of public parameters *formally* fixes the master secret key to be:

$$\mathsf{msk} = (s_1 = \alpha_1 \cdot x_1 + \alpha_3 \cdot x_3, s_2 = \alpha_2 \cdot x_2 + \alpha_3 \cdot x_3)$$

- H_1, H_2 ***Query Phase-1***: $\mathcal{A}$ is allowed to issue H_1 and H_2 queries. $\mathcal{B}$ maintains a list of three-tuples $\left(\mathsf{id}_j, y_j, y'_j\right) \in \mathcal{ID}_\lambda \times \mathbb{Z}_q \times \mathbb{Z}_q$. Upon receiving a query for an identity id_i it first looks up the list for a matching $(\mathsf{id}_i, y_i, y'_i)$ entry. If not found, it samples $y_i, y'_i \xleftarrow{R} \mathbb{Z}_q$, and adds the tuple $(\mathsf{id}_i, y_i, y'_i)$ to the list. Finally, it responds with $H_1(\mathsf{id}_i) = g^{y_i}$ and $H_2(\mathsf{id}_i) = g^{y'_i}$.

- ***Secret-Key Query Phase-1:*** When $\mathcal{A}$ issues a secret-key query for some identity id_i, $\mathcal{B}$ looks up $H_1(\mathsf{id}_i)$ and $H_2(\mathsf{id}_i)$, as described above. Let $y_{1,i}$ and $y_{2,i}$ be the corresponding tuple entries. $\mathcal{B}$ samples $z_{1,i}, z_{2,i} \xleftarrow{R} \mathbb{Z}_q$, and responds with:

$$\mathsf{sk}_{\mathsf{id}_i} = \left((g_1^{x_1} \cdot g_3^{x_3})^{y_{1,i} \cdot z_{1,i}} \cdot (g_2^{x_2} \cdot g_3^{x_3})^{y_{2,i} \cdot z_{2,i}}, z_{1,i}, z_{2,i}\right)$$

  It is easy to see that this simulation of the key-generation oracle by $\mathcal{B}$ is computationally indistinguishable from the real oracle the experiment $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda)$. In particular, we have:

$$\begin{aligned} \mathsf{sk}_{\mathsf{id}_i} &= \left((g_1^{x_1} \cdot g_3^{x_3})^{y_{1,i} \cdot z_{1,i}} \cdot (g_2^{x_2} \cdot g_3^{x_3})^{y_{2,i} \cdot z_{2,i}}, z_{1,i}, z_{2,i}\right) \\ &= \left((g^{y_{1,i}})^{s_1 \cdot z_{1,i}} \cdot (g^{y_{2,i}})^{s_2 \cdot z_{2,i}}, z_{1,i}, z_{2,i}\right) \\ &= \left(H_1(\mathsf{id}_i)^{s_1 \cdot z_{1,i}} \cdot H_2(\mathsf{id}_i)^{s_2 \cdot z_{2,i}}, z_{1,i}, z_{2,i}\right) \end{aligned}$$

- ***Left-or-Right Query:*** Suppose $\mathcal{A}$ queries the left-or-right oracle with $(\mathbf{ID}_0^*, \mathbf{ID}_1^*)$ - a two-tuple of circuits representing k-sources over the identity space $\mathcal{ID}_\lambda$ such that $k = \omega(\log \lambda)$. $\mathcal{B}$ uniformly samples $\mathsf{mode} \xleftarrow{R} \{\mathsf{left}, \mathsf{right}\}$. If $\mathsf{mode} = \mathsf{left}$, it samples $\mathsf{id}^* \xleftarrow{R} \mathbf{ID}_0^*$; otherwise, it samples $\mathsf{id}^* \xleftarrow{R} \mathbf{ID}_1^*$. If either $H_1(\mathsf{id}^*)$ or $H_2(\mathsf{id}^*)$ have already been looked up, $\mathcal{B}$ *outputs a random bit and aborts.* Otherwise, it samples $z_1^*, z_2^* \xleftarrow{R} \mathbb{Z}_q$, and responds with:

$$\mathsf{sk}_{\mathsf{id}^*} = (g_1^{a_1 \cdot x_1} \cdot g_2^{a_2 \cdot x_2} \cdot g_3^{a_3 \cdot x_3}, z_1^*, z_2^*)$$

  where $(g_1^{a_1}, g_2^{a_2}, g_3^{a_3})$ is part of its input DLIN instance. It also formally sets $H_1(\mathsf{id}^*) = g^{a_1/z_1^*}$ and $H_2(\mathsf{id}^*) = g^{a_2/z_2^*}$.

- H_1, H_2 ***Query Phase-2***: $\mathcal{A}$ continues to issue queries to the random oracles H_1 and H_2. $\mathcal{B}$ responds as in Phase-1, except if a query for id^* arrives. In this case, $\mathcal{B}$ *outputs* 1 *and aborts.*

- ***Secret-Key Query Phase-2:*** $\mathcal{A}$ continues to issue secret-key queries, and $\mathcal{B}$ continues to respond as in Phase-1, except if a query for id^* arrives. In this case, $\mathcal{B}$ *outputs* 1 *and aborts.*

- ***Output:*** Finally, $\mathcal{A}$ outputs a bit $b \in \{0, 1\}$. $\mathcal{B}$ outputs 1 if the challenge identity id^* was sampled from $\mathbf{ID}_b^*$, and 0 otherwise.

Note that we considered the single-shot variant of the function privacy adversary making a single left-or-right oracle query. Such an adversary is polynomially equivalent to its multi-shot variant (see Section 3.4). We now state and prove the following claims:

Claim 4.2 *When $a_3 = a_1 + a_2$, the joint distribution of* mode *and the challenge secret-key* $\mathsf{sk}_{\mathsf{id}^*}$ *in the simulation of the left-or-right oracle by $\mathcal{B}$ is computationally indistinguishable from that in the experiment* $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP}, \Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}}, \mathcal{A}}(\lambda)$.

Proof. Note that the sampling of id^* from either $\mathbf{ID}_1^*$ or $\mathbf{ID}_1^*$ by $\mathcal{B}$ is consistent with its random choice of mode. Additionally, when $a_3 = a_1 + a_2$, the secret-key $\mathsf{sk}_{\mathsf{id}^*}$ takes the form:

$$\begin{aligned}
\mathsf{sk}_{\mathsf{id}^*} &= \left(g_1^{a_1 \cdot x_1} \cdot g_2^{a_2 \cdot x_2} \cdot g_3^{(a_1 + a_2) \cdot x_3}, z_1^*, z_2^*\right) \\
&= ((g_1^{x_1} \cdot g_3^{x_3})^{a_1} \cdot (g_2^{x_2} \cdot g_3^{x_3})^{a_2}, z_1^*, z_2^*) \\
&= \left((g_1^{x_1} \cdot g_3^{x_3})^{(a_1/z_1^*) \cdot z_1^*} \cdot (g_2^{x_2} \cdot g_3^{x_3})^{(a_2/z_2^*) \cdot z_2^*}, z_1^*, z_2^*\right) \\
&= \left(H_1(\mathsf{id}^*)^{s_1 \cdot z_1^*} \cdot H_2(\mathsf{id}^*)^{s_2 \cdot z_2^*}, z_1^*, z_2^*\right)
\end{aligned}$$

which is identically distributed to the response of the left-or-right oracle in the experiment $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP}, \Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}}, \mathcal{A}}(\lambda)$. This completes the proof of Claim 4.2.

Claim 4.3 *When a_3 is uniformly random in $\mathbb{Z}_q$, the distribution of the challenge secret-key $\mathsf{sk}_{\mathsf{id}^*}$ is statistically independent of $\mathcal{B}$'s choice of* mode *with overwhelmingly large probability.*

Proof. Recall that $\alpha_j = \log_g g_j$ for $j \in \{1, 2, 3\}$. Consider the following system of equations, determined by the public parameters pp and the secret-key $\mathsf{sk}_{\mathsf{id}^*}$:

$$\log_g \left(g_1^{x_1} \cdot g_3^{x_3}\right) = \alpha_1 \cdot x_1 + \alpha_3 \cdot x_3$$
$$\log_g \left(g_2^{x_2} \cdot g_3^{x_3}\right) = \alpha_2 \cdot x_2 + \alpha_3 \cdot x_3$$
$$\log_g \left(g_1^{a_1 \cdot x_1} \cdot g_2^{a_2 \cdot x_2} \cdot g_3^{a_3 \cdot x_3}\right) = a_1 \cdot \alpha_1 \cdot x_1 + a_2 \cdot \alpha_2 \cdot x_2 + a_3 \cdot \alpha_3 \cdot x_3$$

Since a_3 is uniformly random in $\mathbb{Z}_q$, with all but negligible probability, we have that $a_3 \neq a_1 + a_2$, *which makes the aforementioned system of equations linearly independent*. Hence, the *conditional distribution* of $\mathsf{sk}_{\mathsf{id}^*}$ (where the conditioning is on $\mathcal{B}$'s choice of mode and everything else in $\mathcal{A}$'s view) is uniform. This completes the proof of Claim 4.3.
Note that while z_1^* and z_2^* are not directly used in the reduction, they are sampled to be uniformly random to ensure the indistinguishability of the adversary's views in the real and simulated experiments as per Claim 4.2. It now follows from Claims 4.2 and 4.3 that the advantage ϵ' of $\mathcal{B}$ in solving the DLIN instance may be quantified as $\epsilon' = \epsilon \cdot (1 - \Pr[\mathsf{abort}_1]) - \Pr[\mathsf{abort}_2]$, where ϵ is the advantage of the function privacy adversary $\mathcal{A}$, while abort_1 and abort_2 denote the events that aborting the simulation during the left-or-right query phase and the second hash/secret-key query phases, respectively, leads to a loss of advantage for $\mathcal{B}$. We bound the probability of this event as follows:

- **Abortion during Left-or-Right Query.** Suppose that the adversary $\mathcal{A}$ makes Q_1 and Q_2 queries to the random oracles and secret-key generation oracle, respectively, prior to the left-or-right query. Recall that both the circuits $\mathbf{ID}_0^*$ and $\mathbf{ID}_1^*$ generated by $\mathcal{A}$ represent k-sources over the identity space $\mathcal{ID}_\lambda$, such that $k = \omega(\log \lambda)$. Hence, the probability that a uniformly randomly sampled id^* from either distribution has already been queried by $\mathcal{A}$ may be upper bounded as $\frac{(Q_1+Q_2)}{2^{\omega(\log \lambda)}} \leq \mathsf{negl}(\lambda)$.

- **Abortion during Query Phase-2.** Note that when $\mathcal{B}$ aborts during a random oracle/secret-key generation oracle query in phase-2, it always outputs 1, thereby indicating that its input DLIN instance is valid. Hence, a loss of advantage for $\mathcal{B}$ occurs only when it has to abort even if the DLIN instance is invalid. Now, as per Claim 4.3, when the DLIN instance is invalid, the distribution of the challenge secret-key $\mathsf{sk}_{\mathsf{id}^*}$ is uniformly random and independent of mode. Given the min-entropy bounds on the circuits represented by $\mathbf{ID}_0^*$ and $\mathbf{ID}_1^*$, the probability that $\mathcal{A}$ still correctly guesses id^* and issues oracle queries for the same is $\mathcal{O}\left(2^{-\omega(\log \lambda)}\right) \leq \mathsf{negl}(\lambda)$.

In summary, we have $\Pr[\mathsf{abort}_\beta] \leq \mathsf{negl}(\lambda)$ for $\beta \in \{1, 2\}$, implying that $\mathcal{B}$'s advantage in solving the DLIN instance is negligibly close to the advantage of the function privacy adversary $\mathcal{A}$. This completes the proof of Claim 4.1.

We demonstrate in Appendix E that the $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}}$ scheme can be readily extended to a sequence of IBE schemes that share the same data privacy guarantees, while enjoying progressively stronger function privacy guarantees under weaker variants of the DLIN assumption, albeit at the cost of a constant growth in ciphertext size. Finally, while the aforementioned IBE scheme is secure in the random oracle model, our subsequent constructions for subspace-membership encryption and hidden-vector encryption naturally subsume IBE constructions that are secure in the standard model.

5 Computationally Function Private Subspace-Membership Encryption

In this section we present an adaptively data private and computationally function private SME scheme based on the matrix DDH assumption in the standard model. The scheme is described below, and its proofs of data privacy and function privacy are presented subsequently.

The Scheme. Let $\mathsf{GroupGen}(1^\lambda)$ be a probabilistic polynomial-time algorithm that takes as input a security parameter $\lambda \in \mathbb{N}$, and outputs the tuple $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, g_1, g_2, e)$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are distinct groups of prime order q (q being a λ-bit prime), g_1 is a generator for $\mathbb{G}_1$, g_2 is a generator for $\mathbb{G}_2$, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ is an efficiently computable non-degenerate *asymmetric* bilinear map. Our scheme $\Pi^{\mathrm{SME}}_{\mathrm{MDDH}}$ is parameterized by $m, n, l_1, l_2 = \mathsf{poly}(\lambda)$ (for $l_1 > l_2$), in the sense that it supports predicate matrices of the form $\mathbf{W} \in \mathbb{Z}_q^{m \times n}$, and attribute vectors of the form $\mathbf{x} \in \mathbb{Z}_q^n$.

- **Setup:** The setup algorithm samples $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, g_1, g_2, e) \xleftarrow{R} \mathsf{GroupGen}(1^\lambda)$ on input the security parameter 1^λ. It also randomly samples $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{l_1 \times l_2}$, and $\mathbf{S}_0, \mathbf{S}_1, \cdots, \mathbf{S}_n, \mathbf{S}_{n+1} \cdots, \mathbf{S}_{2n} \xleftarrow{R} \mathbb{Z}_q^{l_2 \times l_1}$. It outputs the public parameter pp and the master-secret-key msk as:
$$\mathsf{pp} = \left(g_1, g_1^{\mathbf{A}}, \left\{ g_1^{\mathbf{S}_j \cdot \mathbf{A}} \right\}_{j \in [0,2n]} \right), \ \mathsf{msk} = \left(g_2, \{\mathbf{S}_j\}_{j \in [0,2n]} \right)$$
- **KeyGen:** On input the public parameter pp, the master-secret-key msk and a predicate matrix $\mathbf{W} = \left[w_{i,j} \right]_{i \in [1,m], j \in [1,n]} \in \mathbb{Z}_q^{m \times n}$, the key-generation algorithm uniformly samples n additional columns of values $\{w_{i,n+j}\}_{i \in [1,m], j \in [1,n]} \xleftarrow{R} \mathbb{Z}_q$. Next, it samples $\mathbf{y} = \left[y_1 \ y_2 \cdots y_m \right]^{\mathbf{T}} \xleftarrow{R} \mathbb{Z}_q^m$ and computes the following for each $i \in [1, m]$:
$$d_{i,j} = g_2^{y_i \cdot w_{i,j}} \text{ for } j \in [1, 2n] \ , \ d_{i,0} = g_2^{y_i \cdot \left(\sum_{j=1}^{2n} w_{i,j} \cdot \mathbf{S}_j \right)}$$
It then outputs the secret-key:
$$\mathsf{sk}_{\mathbf{W}} = \left(\left\{ d_j = \prod_{i=1}^{m} d_{i,j} \right\}_{j \in [0,2n]} \right)$$

- **Enc:** On input pp and an attribute vector $\mathbf{x} = \begin{bmatrix} x_1 \ x_2 \cdots x_n \end{bmatrix}^{\mathbf{T}} \in \mathbb{Z}_q^n$, the encryption algorithm sets $x_{n+1} = x_{n+2} = \cdots = x_{2n} = 0$, samples $\mathbf{r} \xleftarrow{R} \mathbb{Z}_q^{l_2}$, and outputs the ciphertext $C = \left(\{c_j\}_{j\in[0,2n]}\right)$ where:

$$c_0 = g_1^{\mathbf{A}\cdot\mathbf{r}} \ , \ c_j = g_1^{(x_j\cdot\mathbf{S}_0+\mathbf{S}_j)\cdot\mathbf{A}\cdot\mathbf{r}} \text{ for } j \in [1, 2n]$$

- **Dec:** On input a ciphertext $C = \left(\{c_j\}_{j\in[0,2n]}\right)$ and a secret-key $\mathsf{sk}_{\mathbf{W}} = \left(\{d_j\}_{j\in[0,2n]}\right)$, the decryption algorithm computes the following :

$$T_j = e\left(c_j, d_j\right) \text{ for } j \in [0, 2n]$$

Finally, the decryption algorithm checks if the following holds:

$$\prod_{j=1}^{2n} T_j = T_0$$

If yes, it returns 0 else it returns 1.

Correctness. Consider a ciphertext $C = \left(\{c_j\}_{j\in[0,2n]}\right)$ corresponding to an attribute vector $\mathbf{x}$, and a secret-key $\mathsf{sk}_{\mathbf{W}} = \left(\{d_j\}_{j\in[0,2n]}\right)$ corresponding to a predicate matrix $\mathbf{W}$. Then we have the following (recall that we use the notations $e(g_1, g_2)^{\mathbf{W}_2\cdot\mathbf{W}_1}$ and $e\left(g_1^{\mathbf{W}_1}, g_2^{\mathbf{W}_2}\right)$, interchangeably):

$$\begin{aligned}\prod_{j=1}^{2n} T_j = \prod_{j=1}^{2n} e\left(c_j, d_j\right) \\ &= e\left(g_1, g_2\right)^{\sum_{j=1}^{2n}\left(\sum_{i=1}^{m} y_i\cdot w_{i,j}\right)\cdot(x_j\cdot\mathbf{S}_0+\mathbf{S}_j)\cdot\mathbf{A}\cdot\mathbf{r}} \\ &= e\left(g_1, g_2\right)^{\sum_{i=1}^{m} y_i\cdot\left(\sum_{j=1}^{2n} w_{i,j}\cdot x_j\right)\cdot\mathbf{S}_0\cdot\mathbf{A}\cdot\mathbf{r}} \cdot e\left(g_1, g_2\right)^{\sum_{i=1}^{m} y_i\cdot\left(\sum_{j=1}^{2n} w_{i,j}\cdot\mathbf{S}_j\right)\cdot\mathbf{A}\cdot\mathbf{r}} \\ &= e\left(g_1, g_2\right)^{\sum_{i=1}^{m} y_i\cdot\left(\sum_{j=1}^{2n} w_{i,j}\cdot x_j\right)\cdot\mathbf{S}_0\cdot\mathbf{A}\cdot\mathbf{r}} \cdot e\left(c_0, d_0\right) \\ &= e\left(g_1, g_2\right)^{\sum_{i=1}^{m} y_i\cdot\left(\sum_{j=1}^{2n} w_{i,j}\cdot x_j\right)\cdot\mathbf{S}_0\cdot\mathbf{A}\cdot\mathbf{r}} \cdot T_0\end{aligned}$$

Now, if $\mathbf{W}\cdot\mathbf{x} = \mathbf{0}$, we have $\sum_{j=1}^{2n} w_{i,j}x_j = 0$ for each $i \in [1, m]$ (recall that $x_{n+1} = x_{n+2} = \cdots = x_{2n} = 0$), which in turn implies $\prod_{j=1}^{2n} T_j = T_0$. On the other hand, if $\mathbf{W}\cdot\mathbf{x} \neq \mathbf{0}$, there exists at least one $i \in [1, m]$ such that $\sum_{j=1}^{2n} w_{i,j}x_j \neq 0$. Hence, with probability $1 - 1/q$ over the randomness of KeyGen, we have $\prod_{j=1}^{2n} T_j \neq T_0$.

5.1 Security of the Scheme

Adaptive Data Privacy. We state the following theorem for the adaptive data privacy of $\Pi_{\mathrm{MDDH}}^{\mathrm{SME}}$:

Theorem 5.1 *Our SME scheme* $\Pi^{\text{SME}}_{\text{MDDH}}$ *is adaptively data private for parameters* $m, n, l_1, l_2 = \mathsf{poly}(\lambda)$ *under the* (l_1, l_2, k)-Source-MDDH *assumption for* $k = \log_2 q$*, subject to the restrictions that:*

- $l_1 > l_2$.
- *The maximum number of secret-key-generation queries made by the adversary in the data privacy experiment is* n.

Proof. The analysis of data privacy relies on hash proof systems, and uses arguments similar to those of Cramer and Shoup [19,20]. Note that the additional attribute vector components $x_{n+1}, \cdots, x_{2n}$ in our SME construction are 0, implying that the final n ciphertext components are statistically independent of the vector $\mathbf{x}$ being encrypted. The analysis now exploits the following fact: the restriction on the number of secret-key queries ensures that any n master-secret-key components $(\mathbf{S}_0, \cdots, \mathbf{S}_n)$ retain sufficient entropy from an adversary's point of view, even given the public parameter pp and $\mathcal{B}$'s responses to the secret-key queries. This in turn ensures that at some stage, if the challenge ciphertext is generated using the master-secret-key instead of the public parameter, it will perfectly hide which attribute among $\mathbf{x}_0^*$ and $\mathbf{x}_1^*$ is encrypted. Finally, the scheme is adaptively secure because the reduction knows the master-secret-key at any time, which allows it to answer all secret-key queries without knowing the challenge attributes beforehand. This feature is common to nearly all security proofs relying on hash proof systems [19,20]. The detailed analysis is presented in Appendix F. The analysis uses indistinguishability-based arguments, similar to those presented by Agrawal et al. in [12] for their adaptively data private IPE construction. We note that the arguments could equivalently be presented in a simulation-based framework, using the elegant techniques proposed by Wee in [13]. We leave this as an interesting future exercise.

Computational Function Privacy. We state the following theorem for the computational function privacy of $\Pi^{\text{SME}}_{\text{MDDH}}$:

Theorem 5.2 *Our SME scheme* $\Pi^{\text{SME}}_{\text{MDDH}}$ *is function private for parameters* $m, n, l_1, l_2 = \mathsf{poly}(\lambda)$ *under the* $(2n, m, k)$-Source-MDDH *assumption for* $k = \omega(\log \lambda)$*, subject to the restrictions that:*

- *Each predicate matrix* $\mathbf{W} \in \mathbb{Z}_q^{m \times n}$ *is sampled uniformly from an* (m, n, k)*-matrix source.*
- $n > m$.
- *The maximum number of secret-key-generation queries made by the adversary during the function privacy experiment is* n.

Proof. The proof follows directly from the following claim:

Claim 5.1 *For any probabilistic polynomial-time adversary* $\mathcal{A}$*, the following holds:*

$$\left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\text{SME}}_{\text{MDDH}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\text{SME}}_{\text{MDDH}},\mathcal{A}}(\lambda) = 1\right] \right| \leq \mathsf{negl}(\lambda)$$

To prove this claim, we assume the contrary. Let $\mathcal{A}$ be a probabilistic polynomial-time adversary such that:

$$\left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] \right| = \epsilon$$

We construct a probabilistic polynomial-time algorithm $\mathcal{B}$ such that:

$$\mathbf{Adv}^{\mathrm{MDDH}}_{2n,m,k,\mathcal{B}}(\lambda) = \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{MDDH},2n,m,k,\mathcal{B}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{MDDH},2n,m,k,\mathcal{B}}(\lambda) = 1\right] \right| = \epsilon$$

with respect to the group $\mathbb{G}_2$, for $k = \omega(\log \lambda)$. $\mathcal{B}$ poses as the challenger for $\mathcal{A}$ in the function privacy experiment, and delays outputting its own choice of matrix distribution in the min-entropy MDDH game until $\mathcal{A}$ queries the left-or-right function privacy oracle. More concretely, $\mathcal{B}$ proceeds as follows:

- ***Setup:*** $\mathcal{B}$ randomly samples $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{l_1 \times l_2}$, and $\mathbf{S}_0, \mathbf{S}_1, \cdots, \mathbf{S}_n, \mathbf{S}_{n+1}, \cdots, \mathbf{S}_{2n} \xleftarrow{R} \mathbb{Z}_q^{l_2 \times l_1}$. It sets the public parameter pp and the master-secret-key msk as:

  $$\mathsf{pp} = \left(g_1, g_1^{\mathbf{A}}, \left\{g_1^{\mathbf{S}_j \cdot \mathbf{A}}\right\}_{j \in [0,2n]}\right) , \ \mathsf{msk} = \left(g_2, \{\mathbf{S}_j\}_{j \in [0,2n]}\right)$$

  It provides pp to $\mathcal{A}$.

- ***Secret-Key Queries:*** Suppose $\mathcal{A}$ issues a key-generation query for a predicate matrix $\mathbf{W} = \left[w_{i,j}\right]_{i \in [1,m], j \in [1,n]} \in \mathbb{Z}_q^{m \times n}$. Since $\mathcal{B}$ has full knowledge of the master-secret-key, it responds with the desired secret-key $\mathsf{sk}_{\mathbf{W}}$ exactly in the real function privacy experiment.

- ***Left-or-Right Query:*** Suppose $\mathcal{A}$ queries the left-or-right oracle with the tuple $(\mathbf{V}_0^*, \mathbf{V}_1^*)$, where both $\mathbf{V}_0^* = \left(\left\{V_{i,j,0}^*\right\}_{i \in [1,m], j \in [1,n]}\right)$ and $\mathbf{V}_1^* = \left(\left\{V_{i,j,1}^*\right\}_{i \in [1,m], j \in [1,n]}\right)$ represent (m,n,k)-matrix-sources over $\mathbb{Z}_q^{m \times n}$ for $k = \omega(\log \lambda)$. $\mathcal{B}$ uniformly samples $\mathsf{mode} \xleftarrow{R} \{\mathsf{left}, \mathsf{right}\}$. If $\mathsf{mode} = \mathsf{left}$, it sets $b = 0$, else it sets $b = 1$.

  At this point, $\mathcal{B}$ outputs the distribution $V'^*_b = \left[\mathbf{V}_b^* \ \mathbf{U}_{m \times n}\right]^{\mathbf{T}}$ where $\mathbf{U}_{m \times n}$ represents a circuit for the uniform distribution over $\mathbb{Z}_q^{m \times n}$, and receives in response a tuple $\left(g_2^{(\mathbf{W}^*)^{\mathbf{T}}}, g_2^{\mathbf{u}^*}\right) \in \left(\mathbb{G}_2^{m \times (2n)} \times \mathbb{G}_2^{2n}\right)$. Note that $\mathbf{u}^* = \left[u_1^* \ u_2^* \cdots u_{2n}^*\right]^{\mathbf{T}}$ is either of the form $(\mathbf{W}^*)^{\mathbf{T}} \cdot \mathbf{y}^*$ for some uniformly random $y^* \in \mathbb{Z}_q^m$, or $\mathbf{u}^*$ is uniformly random in $\mathbb{Z}_q^{2n}$.

  $\mathcal{B}$ now sets:

  $$d_j^* = g_2^{u_j^*} \text{ for } j \in [1,2n] \ , \ d_0^* = g_2^{\sum_{j \in [1,2n]} u_j^* \cdot \mathbf{S}_j} \tag{1}$$

  Finally, $\mathcal{B}$ responds with the challenge secret-key:

  $$\mathsf{sk}_{\mathbf{W}^*} = \left(\left\{d_j^*\right\}_{j \in [0,2n]}\right)$$

- ***Output:*** $\mathcal{A}$ outputs a bit b'. If $b' = b$ (recall that $b = 0$ if $\mathsf{mode} = \mathsf{left}$ and $b = 1$ if $\mathsf{mode} = \mathsf{right}$), $\mathcal{B}$ outputs 1; else, it outputs 0.

Note that we considered the single-shot variant of the function privacy adversary making a single left-or-right oracle query. Such an adversary is polynomially equivalent to its multi-shot variant (see Section 3.4). The following claims now follow:

Claim 5.2 *When $(\mathbf{W}^*)^{\mathbf{T}} \cdot \mathbf{y}^*$ for some uniformly random $y^* \in \mathbb{Z}_q^m$, the joint distribution of mode and the challenge secret-key $\mathsf{sk}_{\mathbf{W}^*}$ in the simulation of the left-or-right oracle by $\mathcal{B}$ is computationally indistinguishable from that in the real function privacy experiment.*

Claim 5.3 *When $\mathbf{u}^*$ is uniformly random in $\mathbb{Z}_q^{2n}$, the distribution of the challenge secret-key $\mathsf{sk}_{\mathbf{W}^*}$ is statistically independent of $\mathcal{B}$'s choice of mode with overwhelmingly large probability.*

Proof. The first claim is obvious. To prove the second claim, it is sufficient to prove that the conditional distribution of $\sum_{j=1}^{2n} u_j^* \cdot \mathbf{S}_j$ (where the conditioning is on $\mathcal{B}$'s choice of mode and everything else in $\mathcal{A}$'s view) is independent of mode, given that $\mathbf{u}^*$ is a uniformly random vector in $\mathbb{Z}_q^{2n}$.

Suppose that during the function privacy experiment, the adversary $\mathcal{A}$ makes n secret-key-generation queries on predicate matrices $\mathbf{W}_1, \cdots, \mathbf{W}_n \in \mathbb{Z}_q^{m \times n}$, and $\mathcal{B}$ responds with $\mathsf{sk}_{\mathbf{W}_l} = \left(\{d_{j,l}\}_{j \in [0,2n]}\right)$ for $l \in [1, n]$. Consider the following system of $(l_2)^2$ equations in $\mathbf{S}_0$ and $\left((2n) \times (l_2)^2 + (n \times l_1 \times l_2)\right)$ equations in $\{\mathbf{S}_j\}_{j \in [1,2n]}$, determined by the public parameters and the responses of $\mathcal{B}$ to the aforementioned key-generation queries:

$$\log_{g_1}\left(g^{\mathbf{S}_0 \cdot \mathbf{A}}\right) = \mathbf{S}_0 \cdot \mathbf{A}$$
$$\log_{g_1}\left(g^{\mathbf{S}_j \cdot \mathbf{A}}\right) = \mathbf{S}_j \cdot \mathbf{A} \text{ for } j \in [1, 2n]$$
$$\log_{g_2}(d_{0,l}) = y_i \cdot \left(\sum_{j=1}^{2n} w_{i,j} \cdot \mathbf{S}_j\right) \text{ for } l \in [1, n]$$

Quite evidently, the aforementioned system of equations has exponentially many solutions for $(\mathbf{S}_0, \mathbf{S}_1, \cdots, \mathbf{S}_{2n})$ so long as $l_1 > l_2$. The details of the resulting distributions for these matrices are presented in Appendix F. For the proof of function privacy, we stress the fact that that each of these distributions must be independent of $\mathcal{B}$'s choice of mode. This follows intuitively from the fact that the master-secret-key components were generated even before $\mathcal{B}$ saw the challenge predicate matrix distributions and made its choice of mode.

Additionally, when $\mathbf{u}^*$ is a uniformly random vector in $\mathbb{Z}_q^{2n}$, each component u_j^* for $j \in [1, 2n]$ is also independent of mode with overwhelmingly large probability. Hence, the conditional distribution of $\sum_{j=1}^{2n} u_j^* \cdot \mathbf{S}_j$ (where the conditioning is on $\mathcal{B}$'s choice of mode and everything else in $\mathcal{A}$'s view) is independent of mode with overwhelmingly large probability. This completes the proof of Claim 5.3.

It now follows from Claims 5.2, and 5.3, that:

$$\begin{aligned}\mathbf{Adv}^{\mathrm{MDDH}}_{2n,m,k,\mathcal{B}}(\lambda) &= \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{MDDH},2n,m,k,\mathcal{B}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{MDDH},2n,m,k,\mathcal{B}}(\lambda) = 1\right] \right| \\ &= \left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] \right| = \epsilon\end{aligned}$$

This completes the proof of Claim 5.1.

Relaxing the Parameter Restriction for Function Privacy. Our SME scheme, parameterized by $m, n = \mathsf{poly}(\lambda)$, is computationally function private subject to the restriction that $n > m$. However, we demonstrate here any instance π of our SME scheme that is parameterized by m, n such that $n \leq m$ can be transformed into an equivalent instance π' that is parameterized by m, n', such that $n' > m$. In particular, the transformation works on the predicate matrices and the attribute vectors as follows:

- Given a predicate matrix $\mathbf{W} \in \mathbb{Z}_q^{m\times n}$ for the instance π, the instance π' uses a corresponding predicate matrix $\mathbf{W}' = \left[\mathbf{W} \mid \mathbf{R}\right] \in \mathbb{Z}_q^{m\times n'}$, where $\mathbf{R} \in \mathbb{Z}_q^{m\times (n'-n)}$ is sampled uniformly from an $(m, (n'-n), k)$-matrix-source for $k = \omega(\log \lambda)$.
- Given an attribute vector $\mathbf{x} \in \mathbb{Z}_q^n$ for the instance π, the instance π' uses a corresponding attribute vector $\mathbf{x}' = \left[\mathbf{x}^{\mathbf{T}} \mid 0\,0\cdots 0\right]^{\mathbf{T}} \in \mathbb{Z}_q^{n'}$.

The following straightforward observations establish the validity of this transformation:

- $\mathbf{W}'$ is an (m, n', k)-matrix-source if and only if $\mathbf{W}$ is an (m, n, k)-matrix-source.
- $\mathbf{W}' \cdot \mathbf{x}' = \mathbf{0}$ if and only if $\mathbf{W} \cdot \mathbf{x} = \mathbf{0}$, where $\mathbf{0} \in \mathbb{Z}_q^m$.

6 Computationally Function Private Hidden-Vector Encryption

In this section we present an adaptively data private and computationally function private HVE scheme based on the matrix DDH assumption in the standard model. The scheme uses techniques similar to the function private HVE scheme presented in Section 5, with subtle differences to account for the presence of the wildcard characters. The scheme is described below, and its proofs of data privacy and function privacy are presented subsequently.

The Scheme. Let $\mathsf{GroupGen}(1^\lambda)$ be a probabilistic polynomial-time algorithm that takes as input a security parameter $\lambda \in \mathbb{N}$, and outputs the tuple $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, g_1, g_2, e)$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are distinct groups of prime order q (q being a λ-bit prime), g_1 is a generator for $\mathbb{G}_1$, g_2 is a generator for $\mathbb{G}_2$, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ is an efficiently computable non-degenerate *asymmetric* bilinear map. Our scheme $\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}}$ is parameterized by $n, l_1, l_2 = \mathsf{poly}(\lambda)$ (for $l_1 > l_2$), in the sense that it supports predicate vectors of the form $\mathbf{v} \in (\mathbb{Z} \cup \{0,1\})^n_q$, and attribute vectors of the form $\mathbf{x} \in \mathbb{Z}^n_q$.

- **Setup:** The setup algorithm samples $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, g_1, g_2, e) \xleftarrow{R} \mathsf{GroupGen}(1^\lambda)$ on input the security parameter 1^λ. It also randomly samples $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{l_1 \times l_2}$, and $\mathbf{S}_0, \mathbf{S}_1, \cdots, \mathbf{S}_n, \mathbf{S}_{2n} \cdots, \mathbf{S}_{2n} \xleftarrow{R} \mathbb{Z}_q^{l_2 \times l_1}$. It outputs the public parameter pp and the master-secret-key msk as:

$$\mathsf{pp} = \left(g_1, g_1^{\mathbf{A}}, \left\{ g_1^{\mathbf{S}_j \cdot \mathbf{A}} \right\}_{j \in [0,2n]} \right) \ , \ \mathsf{msk} = \left(g_2, \{\mathbf{S}_j\}_{j \in [0,2n]} \right)$$

- **KeyGen:** On input the public parameter pp, the master-secret-key msk and a predicate vector $\mathbf{v} = \begin{bmatrix} v_1 \cdots v_n \end{bmatrix}^{\mathbf{T}} \in (\mathbb{Z}_q \cup \{\star\})^n$, the key-generation algorithm first sets:

$$\mathcal{S} = \{j \in [1,n] : v_j \neq \star\} = \left\{ j_1, j_2, \cdots, j_{|\mathcal{S}|} \right\}$$
$$\mathbf{v}_{\mathcal{S}} = \begin{bmatrix} v_j \end{bmatrix}_{j \in \mathcal{S}} \in \mathbb{Z}_q^{|\mathcal{S}|}$$

  It then samples a random *invertible* matrix $\mathbf{W}_1 \xleftarrow{R} \mathbb{Z}_q^{|\mathcal{S}| \times |\mathcal{S}|}$ and sets:

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 \mid \mathbf{W}_1 \cdot \mathbf{v}_{\mathcal{S}} \end{bmatrix} \in \mathbb{Z}_q^{|\mathcal{S}| \times (|\mathcal{S}|+1)}$$

  Let $\mathbf{W} = \begin{bmatrix} w_{i,k} \end{bmatrix}_{i \in [1,|\mathcal{S}|], k \in [1,|\mathcal{S}|+1]}$. At this point, the key-generation algorithm samples $\mathbf{y} = \begin{bmatrix} y_1 \ y_2 \cdots y_{|\mathcal{S}|} \end{bmatrix}^{\mathbf{T}} \xleftarrow{R} \mathbb{Z}_q^{|\mathcal{S}|}$. It also samples $n-1$ additional columns of values $\{w_{i,n+1+j}\}_{i \in [1,m], j \in [1,n-1]} \xleftarrow{R} \mathbb{Z}_q$, and computes the following for each $i \in [1, |\mathcal{S}|]$:

$$d_{i,k} = g_2^{y_i \cdot w_{i,k}} \text{ for } k \in [1, |\mathcal{S}| + n] \ , \ d_{i,0} = g_2^{y_i \cdot \left(\left(\sum_{k=1}^{|\mathcal{S}|} w_{i,k} \cdot \mathbf{S}_{j_k} \right) + \left(\sum_{k=1}^{n} w_{i,|\mathcal{S}|+k} \cdot \mathbf{S}_{n+k} \right) \right)}$$

  It then outputs the secret-key:

$$\mathsf{sk}_{\mathbf{v}} = \left(\mathcal{S}, \left\{ d_k = \prod_{i=1}^{|\mathcal{S}|} d_{i,k} \right\}_{k \in [0,|\mathcal{S}|+n]} \right)$$

- **Enc:** On input pp and an attribute vector $\mathbf{x} = \begin{bmatrix} x_1 \ x_2 \cdots x_n \end{bmatrix}^{\mathbf{T}} \in \mathbb{Z}_q^n$, the encryption algorithm sets $x_{n+1} = -1$ and $x_{n+2} = \cdots = x_{2n} = 0$, samples $\mathbf{r} \xleftarrow{R} \mathbb{Z}_q^{l_2}$, and outputs the ciphertext $C = \left(\{c_j\}_{j \in [0,2n]} \right)$ where:

$$c_0 = g_1^{\mathbf{A} \cdot \mathbf{r}} \ , \ c_j = g_1^{(x_j \cdot \mathbf{S}_0 + \mathbf{S}_j) \cdot \mathbf{A} \cdot \mathbf{r}} \text{ for } j \in [1, 2n]$$

- **Dec:** On input a ciphertext $C = \left(\{c_j\}_{j\in[0,2n]}\right)$ and a secret-key $\mathsf{sk}_{\mathbf{v}} = \left(\mathcal{S}, \{d_k\}_{k\in[0,|\mathcal{S}|+n]}\right)$, the decryption algorithm parses $\mathcal{S}$ as:

$$\mathcal{S} = \left\{j_1, j_2, \cdots, j_{|\mathcal{S}|}\right\}$$

  It then computes the following :

$$T_0 = e\left(c_0, d_0\right)\ , T_k = e\left(c_{j_k}, d_k\right) \text{ for } k \in [1, |\mathcal{S}|]\ , T_{|\mathcal{S}|+k} = e\left(c_{n+k}, d_{|\mathcal{S}|+k}\right) \text{ for } k \in [1, n]$$

  Finally, the decryption algorithm checks if the following holds:

$$\prod_{k=1}^{|\mathcal{S}|+n} T_k = T_0$$

  If yes, it returns 0 else it returns 1.

Correctness. Consider a predicate vector $\mathbf{v} = \left[v_1 \cdots v_n\right]^{\mathbf{T}} \in (\mathbb{Z}_q \cup \{\star\})^n$ and an attribute vector $\mathbf{x} = \left[x_1 \cdots x_n\right]^{\mathbf{T}} \in \mathbb{Z}_q^n$. Also, let $\mathcal{S} = \{j \in [1, n] : v_j \neq \star\}$. Also, let $\mathbf{v}_{\mathcal{S}}$ and $\mathbf{x}_{\mathcal{S}}$ be defined as:

$$\mathbf{v}_{\mathcal{S}} = \left[v_j\right]_{j\in\mathcal{S}} \in \mathbb{Z}_q^{|\mathcal{S}|}\ , \mathbf{x}_{\mathcal{S}} = \left[x_j\right]_{j\in\mathcal{S}} \in \mathbb{Z}_q^{|\mathcal{S}|}$$

If $v_j = x_j$ for each $j \in \mathcal{S}$, we must have $\mathbf{v}_{\mathcal{S}} = \mathbf{x}_{\mathcal{S}}$, and hence the following holds:

$$\mathbf{W} \cdot \begin{bmatrix} \mathbf{x}_{\mathcal{S}} \\ (-1) \end{bmatrix} = \left[\mathbf{W}_1 \mid \mathbf{W}_1 \cdot \mathbf{v}_{\mathcal{S}}\right] \cdot \begin{bmatrix} \mathbf{v}_{\mathcal{S}} \\ (-1) \end{bmatrix} = \mathbf{0} \in \mathbb{Z}_q^{|\mathcal{S}|}$$

The correctness of the HVE scheme now follows directly from the correctness of our SME scheme presented in Section 5.

6.1 Security of the Scheme

Adaptive Data Privacy. We state the following theorem for the adaptive data privacy of $\Pi_{\mathrm{MDDH}}^{\mathrm{HVE}}$:

Theorem 6.1 *Our HVE scheme $\Pi_{\mathrm{MDDH}}^{\mathrm{HVE}}$ is adaptively data private for parameters $n, l_1, l_2 = \mathsf{poly}(\lambda)$ under the (l_1, l_2, k)-*source-matrix decisional Diffie Hellman *assumption for $k = \log_2 q$ subject to the restrictions that:*

- *$l_1 > l_2$*
- *The maximum number of secret-key-generation queries made by the adversary in the data privacy experiment is n.*

Proof. The analysis of data privacy again uses arguments based on hash proof systems, akin to those in the proof of data privacy for our SME construction in Section 6. Note that the encryption algorithms for our HVE and SME constructions are practically identical, except for the fact that the attribute vector component x_{n+1} is -1 (or alternatively $q-1$) in the HVE construction, as opposed to 0 in the SME construction. But the crux remains the same: the final n ciphertext components in our HVE construction are still statistically independent of the vector $\mathbf{x}$ being encrypted. The analysis of data privacy again exploits the fact that the restriction on the number of secret-key queries ensures that any n master-secret-key components $(\mathbf{S}_0, \cdots, \mathbf{S}_n)$ retain sufficient entropy from an adversary's point of view, even given the public parameter pp and $\mathcal{B}$'s responses to n-many secret-key queries. This in turn ensures that at some stage, if the challenge ciphertext is generated using the master-secret-key instead of the public parameter, it will perfectly hide which attribute among $\mathbf{x}_0^*$ and $\mathbf{x}_1^*$ is encrypted. Finally, the scheme is adaptively secure because the reduction knows the master-secret-key at any time, which allows it to answer all secret-key queries without knowing the challenge attributes beforehand.

Computational Function Privacy. We state the following theorem for the computational function privacy of $\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}}$:

Theorem 6.2 *Our HVE scheme $\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}}$ is function private for parameters $n, l_1, l_2 = \mathsf{poly}(\lambda)$ under the $(m, \log_2 q)$-Source-MDDH assumption for some $m \leq 2n$, subject to the restrictions that:*

- *The non-wildcard entries of each predicate vector $\mathbf{v} \in \mathbb{Z}_q^n$ are sampled uniformly from independent k-sources, for $k = \omega(\log \lambda)$.*
- *$n > 1$.*
- *The maximum number of secret-key-generation queries made by the adversary during the function privacy experiment is n.*

Proof. The proof follows directly from the following claim:

Claim 6.1 *For any probabilistic polynomial-time adversary $\mathcal{A}$, the following holds:*

$$\left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] \right| \leq \mathsf{negl}(\lambda)$$

Proof. Let $\mathcal{A}$ be a probabilistic polynomial-time adversary such that:

$$\left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] \right| = \epsilon$$

We construct a probabilistic polynomial-time algorithm $\mathcal{B}$ such that:

$$\mathbf{Adv}^{\mathrm{MDDH}}_{m,k,\mathcal{B}}(\lambda) = \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{MDDH},n,k,\mathcal{B}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{MDDH},m,k,\mathcal{B}}(\lambda) = 1\right] \right| = \epsilon$$

with respect to the group $\mathbb{G}_2$, for some $m \leq n$ and $k = \log_2 q$. $\mathcal{B}$ poses as the challenger for $\mathcal{A}$ in the function privacy experiment, and delays outputting its own choice of matrix distribution in the min-entropy MDDH game until $\mathcal{A}$ queries the left-or-right function privacy oracle. More concretely, $\mathcal{B}$ proceeds as follows:

- ***Setup:*** $\mathcal{B}$ randomly samples $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{l_1 \times l_2}$, and $\mathbf{S}_0, \mathbf{S}_1, \cdots, \mathbf{S}_n, \mathbf{S}_{2n} \cdots, \mathbf{S}_{2n} \xleftarrow{R} \mathbb{Z}_q^{l_2 \times l_1}$. It sets the public parameter pp and the master-secret-key msk as:

$$\mathsf{pp} = \left(g_1, g_1^{\mathbf{A}}, \left\{g_1^{\mathbf{S}_j \cdot \mathbf{A}}\right\}_{j \in [0,2n]}\right), \ \mathsf{msk} = \left(g_2, \{\mathbf{S}_j\}_{j \in [0,2n]}\right)$$

  It provides pp to $\mathcal{A}$.

- ***Secret-Key Queries:*** Suppose $\mathcal{A}$ issues a secret-key query for a vector $\mathbf{v} = \begin{bmatrix} v_1 \cdots v_n \end{bmatrix}^{\mathbf{T}} \in (\mathbb{Z}_q \cup \{\star\})^n$. Since $\mathcal{B}$ has full knowledge of the master-secret-key, it responds with the desired secret-key $\mathsf{sk}_{\mathbf{v}}$ exactly in the real function privacy experiment.

- ***Left-or-Right Query:*** Suppose $\mathcal{A}$ queries the left-or-right oracle with $(\mathbf{V}_0^*, \mathbf{V}_1^*, \mathcal{S}^*)$, where $\mathbf{V}_0^* = \left(\left\{V_{j,0}^*\right\}_{j \in [1,n]}\right)$ and $\mathbf{V}_1^* = \left(\left\{V_{j,1}^*\right\}_{j \in [1,n]}\right)$ represent $(1, n, k)$-matrix-sources for $k = \omega(\log \lambda)$, and $\mathcal{S}^* \subseteq [1, n]$. $\mathcal{B}$ uniformly samples $\mathsf{mode} \xleftarrow{R} \{\mathsf{left}, \mathsf{right}\}$. If $\mathsf{mode} = \mathsf{left}$, it sets $b = 0$, else it sets $b = 1$.

  Now, $\mathcal{B}$ parses/sets the following:

$$\mathcal{S}^* = \left\{j_1^*, j_2^*, \cdots, j_{|\mathcal{S}^*|}^*\right\}$$
$$(\mathbf{V}_{\mathcal{S}^*}^*)_b = \left[V_{j,b}^*\right]_{j \in \mathcal{S}^*}$$

  Note that $(\mathbf{V}_{\mathcal{S}^*}^*)_b$ is a non-zero distribution so long as $\mathcal{S}^* \neq \phi$. At this point, $\mathcal{B}$ outputs the distribution:

$$\tilde{\mathbf{V}}^* = \left[\mathbf{U}_{|\mathcal{S}^*| \times |\mathcal{S}^*|} \mid \mathbf{U}_{|\mathcal{S}^*| \times |\mathcal{S}^*|} \cdot (\mathbf{V}_{\mathcal{S}^*}^*)_b \mid \mathbf{U}_1\right]^{\mathbf{T}}$$

  where $\mathbf{U}_{|\mathcal{S}^*| \times |\mathcal{S}^*|}$ is a circuit representing the uniform distribution over $\mathbb{Z}_q^{|\mathcal{S}^*| \times |\mathcal{S}^*|}$ and $\mathbf{U}_1$ is a circuit represent a uniform distribution over $\mathbb{Z}_q^{|\mathcal{S}^*| \times (n-1)}$.

  $\mathcal{B}$ receives in response a tuple:

$$\left(g^{(\mathbf{W}^*)^{\mathbf{T}}}, g^{\mathbf{u}^*}\right) \in \left((\mathbb{G}_2)^{(|\mathcal{S}^*|+n) \times |\mathcal{S}^*|} \times (\mathbb{G}_2)^{|\mathcal{S}^*|+n}\right)$$

  Note that $\mathbf{u}^*$ is either of the form $(\mathbf{W}^*)^{\mathbf{T}} \cdot \mathbf{y}^*$ for some uniformly random $\mathbf{y}^* \in \mathbb{Z}_q^{|\mathcal{S}^*|}$, or $\mathbf{u}^*$ is uniformly random in $\mathbb{Z}_q^{|\mathcal{S}^*|+n}$.

  Let $\mathbf{u}^* = \left[u_k^*\right]_{k \in [1, |\mathcal{S}^*|+n]}$. Also recall that $\mathcal{S}^*$ has been parsed as $\left\{j_1^*, j_2^*, \cdots, j_{|\mathcal{S}^*|}^*\right\}$. $\mathcal{B}$ responds with the secret-key $\mathsf{sk}_{\mathbf{v}^*} = \left(\{d_k^*\}_{k \in [0, |\mathcal{S}^* + n|]}\right)$ where:

$$d_k^* = g_2^{u_k^*} \text{ for } k \in [1, |\mathcal{S}^* + n|]$$
$$d_0^* = g_2^{\sum_{k=1}^{|\mathcal{S}^*|} u_k^* \cdot \mathbf{S}_{j_k^*} + \sum_{k=1}^{n} u_{|\mathcal{S}^*|+k}^* \cdot \mathbf{S}_{n+k}}$$

- ***Output:*** $\mathcal{A}$ outputs a bit b'. If $b' = b$ (where $b = 0$ if $\mathsf{mode} = \mathsf{left}$ and $b = 1$ if $\mathsf{mode} = \mathsf{right}$), $\mathcal{B}$ outputs 1; else, it outputs 0.

The following claims now follow:

Claim 6.2 *When $\mathbf{u}^*$ is of the form $(\mathbf{W}^*)^{\mathbf{T}} \cdot \mathbf{y}^*$ for some uniformly random $\mathbf{y}^* \in \mathbb{Z}_q^{|\mathcal{S}^*|}$, the joint distribution of mode and the challenge secret-key $\mathsf{sk}_{\mathbf{v}^*}$ in the simulation of the left-or-right oracle by $\mathcal{B}$ is computationally indistinguishable from that in the experiment $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP},\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda)$.*

Claim 6.3 *When $\mathbf{u}^*$ is uniformly random in $\mathbb{Z}_q^{|\mathcal{S}^*|+n}$, the distribution of the challenge secret-key $\mathsf{sk}_{\mathbf{v}^*}$ is statistically independent of $\mathcal{B}$'s choice of mode with overwhelmingly large probability.*

Proof. Claim 6.2 is obvious, while the proof of Claim 6.3 is intuitively identical to the proof of Claim 5.3 in the proof of function privacy for our SME scheme. Note that the the master-secret-key components $(\mathbf{S}_0, \mathbf{S}_1, \cdots, \mathbf{S}_{2n})$ were generated even before $\mathcal{B}$ saw the challenge distributions, and are hence statistically independent of $\mathcal{B}$'s choice of mode with overwhelmingly large probability. Hence, when $\mathbf{u}^*$ is uniformly random in $\mathbb{Z}_q^{|\mathcal{S}^*|+1}$, the conditional distribution of the challenge secret-key $\mathsf{sk}_{\mathbf{v}^*}$ (where the conditioning is on $\mathcal{B}$'s choice of mode and everything else in $\mathcal{A}$'s view) is statistically independent of $\mathcal{B}$'s choice of mode with overwhelmingly large probability. In other words, $\mathsf{sk}_{\mathbf{v}^*}$ perfectly hides the distribution $\mathbf{V}_b^*$ from which the predicate vector $\mathbf{v}^*$ was sampled. The following observations complete the proof of Claim 6.1, and hence the proof of Theorem 6.2:

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{MDDH}}_{|\mathcal{S}^*|+n,k,\mathcal{B}}(\lambda) &= \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{MDDH},|\mathcal{S}^*|+n,k,\mathcal{B}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{MDDH},|\mathcal{S}^*|+n,k,\mathcal{B}}(\lambda) = 1\right] \right| \\
&= \left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{HVE}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] \right| = \epsilon
\end{aligned}$$

References

1. Dan Boneh and Brent Waters. Conjunctive, Subset, and Range Queries on Encrypted Data. In *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, pages 535–554, 2007.
2. Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional Encryption for Inner Product Predicates from Learning with Errors. In *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, pages 21–40, 2011.
3. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. *J. Cryptology*, 26(2):191–224, 2013.

4. Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public Key Encryption with Keyword Search. In *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, pages 506–522, 2004.
5. Dan Boneh, Ananth Raghunathan, and Gil Segev. Function-Private Identity-Based Encryption: Hiding the Function in Functional Encryption. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 461–478, 2013.
6. Dan Boneh, Ananth Raghunathan, and Gil Segev. Function-Private Subspace-Membership Encryption and Its Applications. In *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*, pages 255–275, 2013.
7. Shashank Agrawal, Shweta Agrawal, Saikrishna Badrinarayanan, Abishek Kumarasubramanian, Manoj Prabhakaran, and Amit Sahai. On the practical security of inner product functional encryption. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, pages 777–798, 2015.
8. Vincenzo Iovino, Qiang Tang, and Karol Zebrowski. On the power of public-key function-private functional encryption. In *Cryptology and Network Security - 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings*, pages 585–593, 2016.
9. Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 533–556, 2014.
10. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 503–523, 2015.
11. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. *J. ACM*, 62(6):45:1–45:33, 2015.
12. Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 333–362, 2016.
13. Hoeteck Wee. Attribute-hiding predicate encryption in bilinear groups, revisited. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 206–233. Springer, 2017.
14. Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.
15. Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland,*

February 9-11, 2010. Proceedings, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer, 2010.

16. Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 557–577, 2014.
17. Hoeteck Wee. Dual system encryption via predicate encodings. In Yehuda Lindell, editor, *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 616–637. Springer, 2014.
18. Allison B. Lewko, Yannis Rouselakis, and Brent Waters. Achieving leakage resilience through dual system encryption. In Yuval Ishai, editor, *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, volume 6597 of *Lecture Notes in Computer Science*, pages 70–88. Springer, 2011.
19. Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, pages 13–25, 1998.
20. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *Advances in CryptologyEurocrypt 2002*, pages 45–64. Springer, 2002.
21. Dan Boneh and Matthew Franklin. Identity-based encryption from the weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.
22. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, pages 41–55, 2004.
23. Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge L. Villar. An algebraic framework for diffie-hellman assumptions. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 129–147, 2013.
24. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, pages 415–432, 2008.

A Definitions for Public-Key Predicate Encryption

A public-key predicate encryption scheme for a class of predicates $\mathcal{F}$ over an attribute space Σ and a payload-message space $\mathcal{M}$ is a quadruple of probabilistic polynomial time algorithms $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$. The Setup algorithm takes as input the security parameter λ, and generates the public parameter pp and the master-secret-key msk for the system. The key-generation algorithm, KeyGen takes as input the public parameter pp, the master-secret-key msk and a predicate $f \in \mathcal{F}$, and generates a secret-key sk_f corresponding to f. The Enc algorithm takes as input the public parameter pp, an attribute $I \in \Sigma$ and a payload-message $M \in \mathcal{M}$, and outputs the ciphertext $C = \mathsf{Enc}(\mathsf{pp}, I, M)$. The Dec algorithm takes as input the public parameter pp, a ciphertext C and a secret-key sk_f, and outputs either a payload-message $M \in \mathcal{M}$ or the symbol $\perp$.

Correctness. A predicate encryption scheme Π is said to be functionally correct if for any security parameter λ, for any predicate $f \in \mathcal{F}$, for any attribute $I \in \Sigma$ and any payload-message $M \in \mathcal{M}$, the following hold:

1. If $f(I) = 1$, we have $\mathsf{Dec}(\mathsf{pp}, \mathsf{Enc}(\mathsf{pp}, I, M), \mathsf{KeyGen}(\mathsf{pp}, \mathsf{msk}, f)) = M$.
2. If $f(I) = 0$, we have $\mathsf{Dec}(\mathsf{pp}, \mathsf{Enc}(\mathsf{pp}, I, M), \mathsf{KeyGen}(\mathsf{pp}, \mathsf{msk}, f)) = \perp$ with probability at least $1 - \mathsf{negl}(\lambda)$.

where the probability is taken over the internal randomness of the Setup, KeyGen, Enc, and Dec algorithms.

A.1 Data Privacy of Public-Key Predicate Encryption

We recall the standard notion of *attribute hiding* data privacy for a predicate encryption scheme against an adaptive probabilistic polynomial-time adversary.

Definition A.1 (Adaptively Data Private Predicate Encryption). *A predicate encryption scheme* $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be* adaptively data private *if for any probabilistic polynomial-time adversary* $\mathcal{A}$*, the following holds:*

$$\mathbf{Adv}^{\mathsf{DP}}_{\Pi,\mathcal{A}}(\lambda) \overset{\text{def}}{=} \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathsf{DP},\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathsf{DP},\Pi,\mathcal{A}}(\lambda) = 1\right] \right| \leq \mathsf{negl}(\lambda)$$

where for each $\lambda \in \mathbb{N}$ *and* $b \in \{0,1\}$*, the experiment* $\mathsf{Expt}^{(b)}_{\mathsf{DP},\Pi,\mathcal{A}}(\lambda)$ *is defined as:*

1. $(\mathsf{pp}, \mathsf{msk}) \overset{R}{\leftarrow} \mathsf{Setup}\left(1^\lambda\right)$.
2. $((I_0^*, M_0^*), (I_1^*, M_1^*), \mathsf{state}) \overset{R}{\leftarrow} \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}\left(1^\lambda, \mathsf{pp}\right)$*, where* $I_0^*, I_1^* \in \Sigma$ *and* $M_0^*, M_1^* \in \mathcal{M}$.
3. $C^* \overset{R}{\leftarrow} \mathsf{Enc}(\mathsf{pp}, I_b^*, M_b^*)$.
4. $b' \overset{R}{\leftarrow} \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(C^*, \mathsf{state})$.
5. *Output* b'.

subject to the following restrictions:

1. *For each predicate f_i with which $\mathcal{A}$ queries* $\mathsf{KeyGen}(\mathsf{msk}, \cdot)$*, we have* $f_i(I_0^*) = f_i(I_1^*)$.
2. *If $\mathcal{A}$ queries* $\mathsf{KeyGen}(\mathsf{msk}, \cdot)$ *with a predicate f_i such that* $f_i(I_0^*) = f_i(I_1^*) = 1$*, we have* $M_0^* = M_1^*$.

The above notion of adaptive data privacy is referred to as DP throughout the rest of the paper. There also exists a *selective* variant of the above security notion, referred to as sDP throughout the rest of the paper, that requires the adversary to commit to the challenge pair of attributes (I_0^*, I_1^*) before seeing the public parameters of the scheme.

B The Generalized Decisional *k*-Linear Assumption (*k*-DLIN)

Let $\mathbb{G}$ be a group of prime order q and let $g_1, \cdots, g_k, g_{k+1}$ be arbitrary generators for $\mathbb{G}$, for $k \geq 2$. The generalized decisional k-linear assumption is that the distribution ensembles:

$$\left\{ \left(g_1, \cdots, g_k, g_{k+1}, g_1^{a_1}, \cdots, g_k^{a_k}, g_{k+1}^{\sum_{j=1}^{k} a_j} \right) \right\}_{a_1, \cdots, a_k \xleftarrow{R} \mathbb{Z}_q} \quad \text{and}$$
$$\left\{ \left(g_1, \cdots, g_k, g_{k+1}, g_1^{a_1}, \cdots, g_k^{a_k}, g_{k+1}^{a_{k+1}} \right) \right\}_{a_1, \cdots, a_k, a_{k+1} \xleftarrow{R} \mathbb{Z}_q}$$

are computationally indistinguishable, where $g_1, \cdots, g_k, g_{k+1} \xleftarrow{R} \mathbb{G}$.

Note that the k-DLIN assumption implies the $(k+1)$-DLIN assumption for all $k \geq 1$, but the reverse is not necessarily true. In other words, the k-DLIN assumption family is a family of progressively weaker assumptions.

C Proofs of Claims 2.1 and 2.2

Let $\mathbf{V} = \left(\{V_{i,j}\}_{i \in [1,m], j \in [1,n]} \right)$ be an (m,n,k)-matrix-source over $\mathbb{Z}_q^{m \times n}$ for $k = \omega(\log \lambda)$. Sample uniformly at random $\mathbf{W} \xleftarrow{R} \mathbf{V}$ and choose any n rows of $\mathbf{W}$. By definition of a matrix-source, the elements of the resulting $n \times n$ sub-matrix of $\mathbf{W}$ have been sampled from mutually independent distributions, where each distribution is uniformly random over a sub-field $\mathbb{Z}_{q_k}$, where $q_k \leq q$ is a k-bit prime. Consequently, the probability that this sub-matrix has a zero determinant is given by $\left(1 - \prod_{j=1}^{n} \left(1 - \frac{1}{2^{j \cdot k}} \right) \right)$. It is easy to see that this quantity is upper bounded by $\frac{1}{2^k} = \frac{1}{2^{\omega(\log \lambda)}} \leq \mathsf{negl}(\lambda)$. In other words, the matrix $\mathbf{W}$ has full rank n with overwhelmingly large probability. This completes the proof of Claim 2.1.

Again, let $\mathbf{V}_1 = \left(\{V_{i,j,1}\}_{i \in [1,n], j \in [1,n]} \right)$ be an (n,n,k)-matrix-source over $\mathbb{Z}_q^{n \times n}$, such that $k = \omega(\log \lambda)$, and let $\mathbf{V}_2 = \left(\{V_{i,2}\}_{i \in [1,n]} \right)$ be any non-zero distribution over over

$\mathbb{Z}_q^n$. Let $\tilde{\mathbf{V}} = \left[\mathbf{V}_1 \mid \mathbf{V}_1 \cdot \mathbf{V}_2\right]^{\mathbf{T}} \in \mathbb{Z}_q^{(n+1)\times n}$. Sample uniformly at random $\mathbf{W} \xleftarrow{R} \tilde{\mathbf{V}}$ and choose the first n rows of $\mathbf{W}$. By definition of a matrix-source, the elements of the resulting $n \times n$ sub-matrix of $\mathbf{W}$ have been sampled from mutually independent distributions, where each distribution is uniformly random over a sub-field $\mathbb{Z}_{q_k}$, where $q_k \leq q$ is a k-bit prime. Consequently, the probability that this sub-matrix has a zero determinant is given by $\left(1 - \prod_{j=1}^{n}\left(1 - \frac{1}{2^{j\cdot k}}\right)\right)$. It is easy to see that this quantity is upper bounded by $\frac{1}{2^k} = \frac{1}{2^{\omega(\log\lambda)}} \leq \mathsf{negl}(\lambda)$. In other words, the matrix $\mathbf{W}$ has full rank n with overwhelmingly large probability. This completes the proof of Claim 2.2.

D Proof of Theorem 4.1

We define a series of experiments $\left\{\mathsf{Expt}^{(b,k)}_{\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\},k\in\{0,1,2\}}$ as follows:

- The experiments $\left\{\mathsf{Expt}^{(b,0)}_{\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\}}$ are identical to the data privacy experiments $\left\{\mathsf{Expt}^{(b)}_{\mathsf{DP},\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\}}$.
- For each $k \in \{1,2\}$, the experiments $\left\{\mathsf{Expt}^{(b,k)}_{\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\}}$ are identical to their predecessor experiments $\left\{\mathsf{Expt}^{(b,k-1)}_{\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\}}$ except that, in the challenge ciphertext $C^* = (c_1^*, c_2^*, c_3^*)$, the component c_{k+1}^* is additionally statistically independent of b.

Quite obviously, the following holds for any probabilistic polynomial-time adversary $\mathcal{A}$:

$$\left|\Pr\left[\mathsf{Expt}^{(0,2)}_{\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1,2)}_{\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right]\right| = 0$$

Now, for $b \in \{0,1\}$, let $\mathcal{A}$ be any probabilistic polynomial-time adversary such that:

$$\left|\Pr\left[\mathsf{Expt}^{(b,0)}_{\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(b,1)}_{\Pi^{\text{IBE}}_{\text{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right]\right| = \epsilon$$

where $\epsilon > \mathsf{negl}(\lambda)$. We construct a polynomial-time algorithm $\mathcal{B}$ that solves an instance of the DBDH problem with advantage $\epsilon' \geq \epsilon/(e(Q+1))$, where Q is the maximum number of oracle queries made by $\mathcal{A}$ during the data privacy experiment. $\mathcal{B}$ receives as input a DBDH instance $(g, g^{a_1}, g^{a_2}, g^{a_3}, Z)$ over a bilinear group $\mathbb{G}$ with prime order q and generator g, and interacts with $\mathcal{A}$ as follows:

- ***Setup:*** $\mathcal{B}$ randomly samples $a_4 \xleftarrow{R} \mathbb{Z}_q$, and provides $\mathcal{A}$ with the public parameter pp as:

  $$\mathsf{pp} = (g, g^{a_1}, g^{a_4})$$

  where g and g^{a_1} are part of its input DBDH instance. Then observe that $\mathcal{B}$'s choice of public parameters *formally* fixes the master secret key to be:

  $$\mathsf{msk} = (s_1 = a_1, s_2 = a_4)$$

- H_1, H_2 ***Query Phase-1***: $\mathcal{A}$ is allowed to issue H_1 and H_2 queries. $\mathcal{B}$ maintains a list of four-tuples $\left(\mathsf{id}_j, y_j, y'_j, \delta_j\right) \in \mathcal{ID}_\lambda \times \mathbb{Z}_q \times \mathbb{Z}_q \times \{0,1\}$. Upon receiving a query for an identity id_i it first looks up the list for a matching $(\mathsf{id}_i, y_i, y'_i, \delta_i)$ entry. If not found, it uniformly samples $y_i, y'_i \xleftarrow{R} \mathbb{Z}_q$, and samples δ_i from a distribution over $\{0,1\}$ such that $\Pr[\delta_i = 1] = \gamma$. It then adds the tuple $(\mathsf{id}_i, y_i, y'_i, \delta_i)$ to the list and proceeds as follows:
  - If $\delta_i = 0$, $\mathcal{B}$ responds with $H_1(\mathsf{id}_i) = g^{a_2 \cdot y_i}$ and $H_2(\mathsf{id}_i) = g^{y'_i}$, where g^{a_2} is part of its input DBDH instance.
  - If $\delta_i = 1$, $\mathcal{B}$ responds with $H_1(\mathsf{id}_i) = g^{y_i}$ and $H_2(\mathsf{id}_i) = g^{y'_i}$.

  Note that only the output of the random oracle H_1 depends on δ.

- ***Secret-Key Queries:*** When $\mathcal{A}$ issues a secret-key query for some identity id_i, $\mathcal{B}$ looks up $H_1(\mathsf{id}_i)$ and $H_2(\mathsf{id}_i)$, as described above. Let y_i, y'_i and δ_i be the corresponding tuple entries. $\mathcal{B}$ proceeds as follows:
  - If $\delta_i = 0$, $\mathcal{B}$ *outputs a random bit and aborts.*
  - If $\delta_i = 1$, $\mathcal{B}$ samples $z_{1,i}, z_{2,i} \xleftarrow{R} \mathbb{Z}_q$ and responds with:

$$\mathsf{sk}_{\mathsf{id}_i} = \left(g^{a_1 \cdot y_1 \cdot z_{1,i}} \cdot g^{a_4 \cdot y'_1 \cdot z_{2,i}}, z_{1,i}, z_{2,i}\right)$$

  Observe that if $\delta_i = 1$, we have $H_1(\mathsf{id}_i) = g^{y_i}$ and $H_2(\mathsf{id}_i) = g^{y'_i}$; consequently, the secret-key $\mathsf{sk}_{\mathsf{id}_i}$ is well-formed with respect to the public parameter pp.

- ***Challenge:*** $\mathcal{A}$ outputs the challenge pair $((\mathsf{id}^*_0, M^*_0), (\mathsf{id}^*_1, M^*_1))$. $\mathcal{B}$ samples a random bit $b \xleftarrow{R} \{0,1\}$ and looks up $H_1(\mathsf{id}^*_b)$ and $H_2(\mathsf{id}^*_b)$, as described above. Let y^*_1, y^*_2 and δ^* be the corresponding tuple entries. $\mathcal{B}$ proceeds as follows:
  - If $\delta^* = 1$, $\mathcal{B}$ *outputs a random bit and aborts.*
  - If $\delta^* = 0$, we must have we have $H_1(\mathsf{id}^*_b) = g^{a_2 \cdot y^*_1}$ and $H_2(\mathsf{id}^*_b) = g^{y^*_2}$. In this case, $\mathcal{B}$ outputs the challenge ciphertext as:

$$C^* = \left(g^{a_3}, M_b \cdot Z^{y^*_1}, e\left(g^{a_3}, g^{a_4}\right)^{y^*_2}\right)$$

- ***Output:*** Finally, $\mathcal{A}$ outputs a guess b' for b. If $b' = b$, $\mathcal{B}$ outputs 1, else it outputs 0.

We now state and prove the following claims:

Claim D.1 *When $Z = e(g,g)^{a_1 \cdot a_2 \cdot a_3}$, the joint distribution of b and the challenge ciphertext C^* in the simulation by $\mathcal{B}$ is computationally indistinguishable from that in the experiment $\mathsf{Expt}^{(b,0)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda)$.*

Proof. Let $Z = e(g,g)^{a_1 \cdot a_2 \cdot a_3}$. As already discussed above, if $\mathcal{B}$ does not abort, we must have $H_1(\mathsf{id}^*_b) = g^{a_2 \cdot y^*_1}$ and $H_2(\mathsf{id}^*_b) = g^{y^*_2}$. Then the challenge ciphertext C^* takes the form:

$$\begin{aligned} C^* &= \left(g^{a_3}, M_b \cdot e(g,g)^{a_1 \cdot a_2 \cdot a_3 \cdot y^*_1}, e\left(g^{a_3}, g^{a_4}\right)^{y^*_2}\right) \\ &= \left(g^{a_3}, M_b \cdot e\left(g^{s_1}, H_1\left(\mathsf{id}^*_b\right)\right)^{a_3}, e\left(g^{s_2}, H_2\left(\mathsf{id}^*_b\right)\right)^{a_3}\right) \end{aligned}$$

for $s_1 = a_1$ and $s_2 = a_4$. Quite evidently, C^* is identically distributed to the challenge ciphertext in the experiment $\mathsf{Expt}^{(b,0)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda)$. This completes the proof of Claim D.1.

Claim D.2 *When Z is uniformly random in $\mathbb{G}_T$, the joint distribution of b and the challenge ciphertext C^* in the simulation by $\mathcal{B}$ is computationally indistinguishable from that in the experiment $\mathsf{Expt}^{(b,1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda)$.*

Proof. The claim follows trivially from the fact that if Z is uniformly random in $\mathbb{G}_T$, the message M_b is perfectly one-time padded in the second component of the challenge ciphertext C^*, while the remaining components of C^* are well-formed with respect to b.

It now follows from Claims D.1 and D.2 that the advantage ϵ' of $\mathcal{B}$ in solving the DBDH instance may be quantified as $\epsilon' = \epsilon \cdot (1 - \Pr[\mathsf{abort}])$, where ϵ is the advantage of $\mathcal{A}$, and abort denotes the event of $\mathcal{B}$ aborting the simulation. We bound the probability of this event as follows:

- Suppose that the adversary $\mathcal{A}$ makes a maximum of Q_1, and Q_2 queries to the random oracle and the secret-key generation oracle, respectively. The probability that $\mathcal{A}$ does not abort is lower-bounded by $\gamma^{Q_2} \cdot (1-\gamma)$.
- Now, $\mathcal{B}$ can set $\gamma = 1 - 1/(Q_2+1)$. Then, we have:

$$\Pr[\mathsf{abort}] \leq 1 - (1 - 1/(Q_2+1))^{Q_2} / (Q_2+1) \leq 1 - 1/\mathsf{e} \cdot (Q_2+1)$$

In other words, we have $\epsilon' \geq \epsilon/\mathsf{e} \cdot (Q_2+1)$. Hence, we must have $\epsilon \leq \mathsf{negl}(\lambda)$. Following a similar proof technique, one can also show that for any probabilistic polynomial-time adversary $\mathcal{A}$ and for $b \in \{0,1\}$, the following also holds:

$$\left|\Pr\left[\mathsf{Expt}^{(b,1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(b,2)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathrm{negl}(\lambda)$$

The proof uses a simulation similar to the one described above, except that the algorithm $\mathcal{B}$ embeds its input DBDH instance in the third component of the challenge ciphertext C^*, while the second component is anyways set to a uniformly random element in $\mathbb{G}_T$. It is now easy to see the following:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{DP}}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) &= \left|\Pr\left[\mathsf{Expt}^{(0)}_{\mathsf{DP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathsf{DP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right]\right| \\ &\leq \sum_{k\in\{1,2\}} \sum_{b\in\{0,1\}} \left|\Pr\left[\mathsf{Expt}^{(b,k-1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(b,k)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+DLIN}},\mathcal{A}}(\lambda) = 1\right]\right| \\ &\leq \mathsf{negl}(\lambda)\end{aligned}$$

This completes the proof of Theorem 4.1.

E A DBDH+k-DLIN-based IBE Scheme

In this section, we demonstrate that our $\Pi^{\text{IBE}}_{\text{DBDH+DLIN}}$ scheme () is that it can be readily extended to a sequence of IBE schemes that share the same data privacy guarantees, while enjoying progressively stronger function privacy guarantees. For $k > 1$, the $(k-1)^{\text{th}}$ IBE scheme in this sequence, denoted as $\Pi^{\text{IBE}}_{\text{DBDH+k-DLIN}}$, is adaptively data private under the DBDH assumption, and computationally function private under the k-DLIN assumption, in the random oracle model. We present the construction for $\Pi^{\text{IBE}}_{\text{DBDH+k-DLIN}}$ below. The detailed proofs for data and function privacy are presented subsequently.

The Scheme. Let $\mathsf{GroupGen}(1^\lambda)$ be a probabilistic polynomial-time algorithm that takes as input a security parameter λ, and outputs the tuple $(\mathbb{G}, \mathbb{G}_T, q, g, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of of order q (q being a λ-bit prime), g is a generator for $\mathbb{G}$ and $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ is an efficiently computable non-degenerate bilinear map. The scheme $\Pi^{\text{IBE}}_{\text{DBDH+k-DLIN}}$ is parameterized by the security parameter $\lambda \in \mathbb{N}$. For any such λ, we denote by $\mathcal{ID}_\lambda$ and $\mathcal{M}_\lambda$ the identity space and the message space, respectively. The scheme uses k hash functions $\{H_j : \mathcal{ID}_\lambda \longrightarrow \mathbb{G}\}_{j\in[1,k]}$ (modeled as random oracles).

- **Setup:** The setup algorithm samples $(\mathbb{G}, \mathbb{G}_T, q, g, e) \overset{R}{\longleftarrow} \mathsf{GroupGen}(1^\lambda)$ on input the security parameter 1^λ. It also samples $s_1, \cdots, s_k \overset{R}{\longleftarrow} \mathbb{Z}_q$, and outputs the public parameter pp and the master-secret-key msk as:

$$\mathsf{pp} = \left(g, \{g^{s_j}\}_{j\in[1,k]}\right) \ , \ \mathsf{msk} = \left(\{s_j\}_{j\in[1,k]}\right)$$

- **KeyGen:** On input the public parameter pp, the master-secret-key msk and an identity $\mathsf{id} \in \mathcal{ID}_\lambda$, the key generation algorithm samples $z_1, \cdots, z_k \overset{R}{\longleftarrow} \mathbb{Z}_q$ and outputs the secret-key $\mathsf{sk}_{\mathsf{id}} = \left(\{d_j\}_{j\in[1,k+1]}\right)$ where:

$$d_1 = \prod_{j=1}^{k} H_j(\mathsf{id})^{s_j \cdot z_j} \ , \ d_{j+1} = z_j \text{ for } j \in [1,k]$$

- **Enc:** On input the public parameter pp, an identity $\mathsf{id} \in \mathcal{ID}_\lambda$ and a message $M \in \mathcal{M}_\lambda$, the encryption algorithm samples $r \overset{R}{\longleftarrow} \mathbb{Z}_q$ and outputs the ciphertext $C = \left(\{c_j\}_{j\in[1,k+1]}\right)$ where:

$$c_1 = g^r \ , \ c_2 = M \cdot e\left(g^{s_1}, H_1(\mathsf{id})\right)^r \ , \ c_{j+1} = e\left(g^{s_j}, H_j(\mathsf{id})\right)^r \text{ for } j \in [2,k]$$

- **Dec:** On input a ciphertext $C = \left(\{c_j\}_{j\in[1,k+1]}\right)$ and a secret-key $\mathsf{sk}_{\mathsf{id}} = \left(\{d_j\}_{j\in[1,k+1]}\right)$, the decryption algorithm outputs:

$$M' = \left(\frac{\prod_{j=1}^{k} c_{j+1}^{d_{j+1}}}{e(d_1, c_1)}\right)^{1/d_2}$$

Correctness. Consider a ciphertext $C = \left(\{c_j\}_{j\in[1,k+1]}\right)$ corresponding to a message M under an identity id, and a secret-key $\mathsf{sk}_{\mathsf{id}} = \left(\{d_j\}_{j\in[1,k+1]}\right)$ corresponding to the same identity id. Then, we have:

$$\begin{aligned} M' &= \left(\frac{\prod_{j=1}^{k} c_{j+1}^{d_{j+1}}}{e\left(d_1, c_1\right)}\right)^{1/d_2} \\ &= \left(\frac{M^{z_1}\cdot\prod_{j=1}^{k} e\left(g^{s_j}, H_j\left(\mathsf{id}\right)\right)^{r\cdot z_j}}{e\left(\prod_{j=1}^{k} H_j\left(\mathsf{id}\right)^{s_j\cdot z_j}, g^r\right)}\right)^{1/z_1} \\ &= M\cdot\left(\frac{\prod_{j=1}^{k} e\left(g^{s_j}, H_j\left(\mathsf{id}\right)\right)^{r\cdot z_j}}{\prod_{j=1}^{k} e\left(g^{s_j}, H_j\left(\mathsf{id}\right)\right)^{r\cdot z_j}}\right)^{1/z_1} \\ &= M \end{aligned}$$

Therefore as long as $z_1 \neq 0 \pmod q$ (an event which occurs with probability $1 - 1/q$ over the randomness of KeyGen), the message is recovered correctly.

Adaptive Data Privacy. We state the following theorem for the adaptive data privacy of $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}}$:

Theorem E.1 *The IBE scheme* $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}}$ *is adaptively data private under the* DBDH *assumption.*

Proof. We define a series of experiments $\left\{\mathsf{Expt}^{(b,k')}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\},k'\in[1,k+1]}$ as follows:

- The experiments $\left\{\mathsf{Expt}^{(b,0)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\}}$ are identical to the experiments $\left\{\mathsf{Expt}^{(b)}_{\mathsf{DP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\}}$.
- For each $k' \in [1, k+1]$, the experiments $\left\{\mathsf{Expt}^{(b,k')}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\}}$ are identical to their predecessor experiments $\left\{\mathsf{Expt}^{(b,k'-1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda)\right\}_{b\in\{0,1\}}$ except that, in the challenge ciphertext $C^* = \left(\left\{c^*_j\right\}_{j\in[1,k+2]}\right)$, the component $c^*_{k'+1}$ is additionally statistically independent of b.

Quite obviously, the following holds for any probabilistic polynomial-time adversary $\mathcal{A}$:

$$\left|\Pr\left[\mathsf{Expt}^{(0,k+1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1,k+1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1\right]\right| = 0$$

Now, for $b \in \{0, 1\}$, let $\mathcal{A}$ be any probabilistic polynomial-time adversary such that:

$$\left|\Pr\left[\mathsf{Expt}^{(b,0)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(b,1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1\right]\right| = \epsilon$$

where $\epsilon > \mathsf{negl}(\lambda)$. We construct a polynomial-time algorithm $\mathcal{B}$ that solves an instance of the DBDH problem with advantage $\epsilon' \geq \epsilon / (e(Q+1))$, where Q is the maximum number of oracle queries made by $\mathcal{A}$ during the data privacy experiment. $\mathcal{B}$ receives as input a DBDH instance $(g, g^{a_1}, g^{a_2}, g^{a_3}, Z)$ over a bilinear group $\mathbb{G}$ with prime order q and generator g, and interacts with $\mathcal{A}$ as follows:

- ***Setup:*** $\mathcal{B}$ randomly samples $a_4, \cdots, a_{k+2} \xleftarrow{R} \mathbb{Z}_q$, and provides $\mathcal{A}$ with the public parameter pp as:
$$\mathsf{pp} = (g, g^{a_1}, g^{a_4}, \cdots, g^{a_{k+2}})$$
where g and g^{a_1} are part of its input DBDH instance. Then observe that $\mathcal{B}$'s choice of public parameters *formally* fixes the master secret key to be:
$$\mathsf{msk} = (s_1 = a_1, s_2 = a_4, \cdots, s_k = a_{k+2})$$

- $H_1, \cdots, H_k$ ***Query Phase-1***: $\mathcal{A}$ is allowed to issue queries to the random oracles for $H_1, H_2 \cdots, H_k$. $\mathcal{B}$ maintains a list of $(k+2)$-tuples $\left(\mathsf{id}_i, \{y_{j,i}\}_{j\in[1,k]}, \delta_i\right) \in \mathcal{ID}_\lambda \times (\mathbb{Z}_q)^k \times \{0,1\}$. Upon receiving a query for an identity id_i it first looks up the list for a matching $\left(\mathsf{id}_i, \{y_{j,i}\}_{j\in[1,k]}, \delta_i\right)$ entry. If not found, it uniformly samples $y_{1,i}, \cdots, y_{k,i} \xleftarrow{R} \mathbb{Z}_q$, and samples δ_i from a distribution over $\{0,1\}$ such that $\Pr[\delta_i = 1] = \gamma$. It then adds the tuple $\left(\mathsf{id}_i, \{y_{j,i}\}_{j\in[1,k]}, \delta_i\right)$ to the list and proceeds as follows:
  - If $\delta_i = 0$, $\mathcal{B}$ responds with $H_1(\mathsf{id}_i) = g^{a_2 \cdot y_{1,i}}$ and $H_j(\mathsf{id}_i) = g^{y_{j,i}}$ for $j \in [2,k]$, where g^{a_2} is part of its input DBDH instance.
  - If $\delta_i = 1$, $\mathcal{B}$ responds with $H_j(\mathsf{id}_i) = g^{y_{j,i}}$ for $j \in [1,k]$.

- ***Secret-Key Queries:*** When $\mathcal{A}$ issues a secret-key query for some identity id_i, $\mathcal{B}$ looks up $H_j(\mathsf{id}_i)$ for $j \in [1,k]$, as described above. Let $\{y_{j,i}\}_{j\in[1,k]}$ and δ_i be the corresponding tuple entries. $\mathcal{B}$ proceeds as follows:
  - If $\delta_i = 0$, $\mathcal{B}$ *outputs a random bit and aborts.*
  - If $\delta_i = 1$, $\mathcal{B}$ samples $z_{1,i}, \cdots, z_{k,i} \xleftarrow{R} \mathbb{Z}_q$ and responds with:
$$\mathsf{sk}_{\mathsf{id}_i} = \left(\left(g^{a_1 \cdot y_{1,i} \cdot z_{1,i}} \cdot \prod_{j=2}^{k} g^{a_{j+2} \cdot y_{j,i} \cdot z_{j,i}}\right), \{z_{j,i}\}_{j\in[1,k]}\right)$$

  Observe that if $\delta_i = 1$, we have $H_j(\mathsf{id}_i) = g^{y_{j,i}}$ for $j \in [1,k]$; consequently, the secret-key $\mathsf{sk}_{\mathsf{id}_i}$ is well-formed with respect to the public parameter pp.

- ***Challenge:*** $\mathcal{A}$ outputs the challenge pair $((\mathsf{id}_0^*, M_0^*), (\mathsf{id}_1^*, M_1^*))$. $\mathcal{B}$ samples a random bit $b \xleftarrow{R} \{0,1\}$ and looks up $H_j(\mathsf{id}_b^*)$ for $j \in [1,k]$, as described above. Let $\{y_j^*\}_{j\in[1,k]}$ and δ^* be the corresponding tuple entries. $\mathcal{B}$ proceeds as follows:
  - If $\delta^* = 1$, $\mathcal{B}$ *outputs a random bit and aborts.*

- If $\delta^* = 0$, we must have $H_1(\mathsf{id}_b^*) = g^{a_2 \cdot y_1^*}$ and $H_j(\mathsf{id}_b^*) = g^{y_j^*}$ for $j \in [2, k]$. In this case, $\mathcal{B}$ outputs the challenge ciphertext as:

$$C^* = \left(g^{a_3}, M_b \cdot Z^{y_1^*}, e\left(g^{a_3}, g^{a_4}\right)^{y_2^*}, \cdots, e\left(g^{a_3}, g^{a_{k+2}}\right)^{y_k^*}\right)$$

- ***Output:*** Finally, $\mathcal{A}$ outputs a guess b' for b. If $b' = b$, $\mathcal{B}$ outputs 1, else it outputs 0.

We now state and prove the following claims:

Claim E.1 *When $Z = e(g,g)^{a_1 \cdot a_2 \cdot a_3}$, the joint distribution of b and the challenge ciphertext C^* in the simulation by $\mathcal{B}$ is computationally indistinguishable from that in the experiment $\mathsf{Expt}^{(b,0)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}}, \mathcal{A}}(\lambda)$.*

Proof. Let $Z = e(g,g)^{a_1 \cdot a_2 \cdot a_3}$. As already discussed above, if $\mathcal{B}$ does not abort, we must have $H_1(\mathsf{id}_b^*) = g^{a_2 \cdot y_1^*}$ and $H_j(\mathsf{id}_b^*) = g^{y_j^*}$ for $j \in [2, k]$. Then the challenge ciphertext C^* takes the form:

$$\begin{aligned} C^* &= \left(g^{a_3}, M_b \cdot e(g,g)^{a_1 \cdot a_2 \cdot a_3 \cdot y_1^*}, e\left(g^{a_3}, g^{a_4}\right)^{y_2^*}, \cdots, e\left(g^{a_3}, g^{a_{k+2}}\right)^{y_k^*}\right) \\ &= \left(g^{a_3}, M_b \cdot e\left(g^{s_1}, H_1\left(\mathsf{id}_b^*\right)\right)^{a_3}, \left\{e\left(g^{s_j}, H_j\left(\mathsf{id}_b^*\right)\right)^{a_3}\right\}_{j \in [2,k]}\right) \end{aligned}$$

for $s_1 = a_1$ and $s_j = a_{j+2}$ for $j \in [2, k]$. Quite evidently, C^* is identically distributed to the challenge ciphertext in the experiment $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP}, \Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}}, \mathcal{A}}(\lambda)$. This completes the proof of Claim E.1.

Claim E.2 *When Z is uniformly random in $\mathbb{G}_T$, the joint distribution of b and the challenge ciphertext C^* in the simulation by $\mathcal{B}$ is computationally indistinguishable from that in the experiment $\mathsf{Expt}^{(b,1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}}, \mathcal{A}}(\lambda)$.*

Proof. The claim follows trivially from the fact that if Z is uniformly random in $\mathbb{G}_T$, the message M_b is perfectly one-time padded in the second component of the challenge ciphertext C^*, while all other components of C^* are well-formed with respect to b.

It now follows from Claims E.1 and E.2 that the advantage ϵ' of $\mathcal{B}$ in solving the DBDH instance may be quantified as $\epsilon' = \epsilon \cdot (1 - \Pr[\mathsf{abort}])$, where ϵ is the advantage of the adversary $\mathcal{A}$, and abort denotes the event of $\mathcal{B}$ aborting the simulation. We bound the probability of this event as follows:

- Suppose that the adversary $\mathcal{A}$ makes a maximum of Q_1 and Q_2 queries to the random oracle and the secret-key generation oracle, respectively. The probability that $\mathcal{A}$ does not abort is lower-bounded by $\gamma^{Q_2} \cdot (1 - \gamma)$.
- Now, $\mathcal{B}$ can set $\gamma = 1 - 1/(Q_2 + 1)$. Then, we have:

$$\Pr[\mathsf{abort}] \leq 1 - \left(1 - 1/(Q_2 + 1)\right)^{Q_2} / (Q_2 + 1) \leq 1 - 1/\mathsf{e} \cdot (Q_2 + 1)$$

In other words, we have $\epsilon' \geq \epsilon/\mathsf{e} \cdot (Q_2 + 1)$. Hence, we must have $\epsilon \leq \mathsf{negl}(\lambda)$. Following a similar proof technique, one can also show that for any probabilistic polynomial-time adversary $\mathcal{A}$, for any $k' \in [2, k+1]$, and for $b \in \{0,1\}$, the following holds:

$$\left| \Pr\left[\mathsf{Expt}^{(b,k'-1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] - \Pr\left[\mathsf{Expt}^{(b,k')}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] \right| \leq \mathrm{negl}(\lambda)$$

The proof uses a sequence of simulations similar to the one described above, except that in the k'^{th} simulation, the algorithm $\mathcal{B}$ embeds its input DBDH instance in the $(k'+1)^{\text{th}}$ component of the challenge ciphertext C^*, while the preceding component are anyways set to a uniformly random element in $\mathbb{G}_T$, and the succeeding components are well-formed with respect to b. It is now easy to see the following:

$$\begin{aligned}
\mathbf{Adv}^{\mathsf{DP}}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) &= \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathsf{DP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathsf{DP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] \right| \\
&\leq \sum_{k' \in [1,k+1]} \sum_{b \in \{0,1\}} \left| \Pr\left[\mathsf{Expt}^{(b,k'-1)}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] - \Pr\left[\mathsf{Expt}^{(b,k')}_{\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] \right| \\
&\leq \mathsf{negl}(\lambda)
\end{aligned}$$

This completes the proof of Theorem E.1.

Computational Function Privacy. We state the following theorem for the computational function privacy of $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}}$:

Theorem E.2 *Our IBE scheme* $\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}}$ *is function private under the* k-DLIN *assumption for identities sampled uniformly from* k'*-sources with* $k' = \omega(\log \lambda)$.

Proof. The proof follows directly from the following claim:

Claim E.3 *For any probabilistic polynomial-time adversary* $\mathcal{A}$*, the following holds:*

$$\left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] \right| \leq \mathsf{negl}(\lambda)$$

To prove this claim, we assume the contrary. Let $\mathcal{A}$ be a probabilistic polynomial-time adversary such that:

$$\left| \Pr\left[\mathsf{Expt}^{\mathsf{left}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] - \Pr\left[\mathsf{Expt}^{\mathsf{right}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda) = 1 \right] \right| = \epsilon$$

where $\epsilon > \mathsf{negl}(\lambda)$. We construct an algorithm $\mathcal{B}$ that solves an instance of the k-DLIN problem with advantage ϵ' negligibly close to ϵ. $\mathcal{B}$ receives as input a k-DLIN instance $\left(\{g_j\}_{j\in[1,k+1]}, \{g_j^{a_j}\}_{j\in[1,k+1]} \right)$ over a bilinear group $\mathbb{G}$ with prime order q and generator g, and interacts with $\mathcal{A}$ as follows:

- ***Setup:*** $\mathcal{B}$ samples $x_1, x_2, \cdots, x_{k+1} \xleftarrow{R} \mathbb{Z}_q$ and provides $\mathcal{A}$ with the public parameter pp as:
$$\mathsf{pp} = \left(g, \left\{g_j^{x_j} \cdot g_{k+1}^{x_{k+1}}\right\}_{j\in[1,k]}\right)$$
where $g_1, \cdots, g_k, g_{k+1}$ are part of its input k-DLIN instance. Let $\alpha_j = \log_g g_j$ for $j \in [1, k+1]$. Then observe that $\mathcal{B}$'s choice of public parameters *formally* fixes the master secret key to be:
$$\mathsf{msk} = \left(\left\{s_j = \alpha_j \cdot x_j + \alpha_{k+1} \cdot x_{k+1}\right\}_{j\in[1,k]}\right)$$

- $H_1, \cdots, H_k$ ***Query Phase-1***: $\mathcal{A}$ is allowed to issue queries to the random oracles for $H_1, H_2 \cdots, H_k$. $\mathcal{B}$ maintains a list of $(k+1)$-tuples $\left(\mathsf{id}_i, \{y_{j,i}\}_{j\in[1,k]}\right) \in \mathcal{ID}_\lambda \times (\mathbb{Z}_q)^k$. Upon receiving a query for an identity id_i it first looks up the list for a matching $\left(\mathsf{id}_i, \{y_{j,i}\}_{j\in[1,k]}\right)$ entry. If not found, it uniformly samples $y_{1,i}, \cdots, y_{k,i} \xleftarrow{R} \mathbb{Z}_q$, and adds the tuple $\left(\mathsf{id}_i, \{y_{j,i}\}_{j\in[1,k]}\right)$ to the list. It finally responds with $H_j(\mathsf{id}_i) = g^{y_{j,i}}$ for $j \in [1, k]$.

- ***Secret-Key Query Phase-1:*** When $\mathcal{A}$ issues a secret-key query for some identity id_i, $\mathcal{B}$ looks up $H_j(\mathsf{id}_i)$ for $j \in [1, k]$, as described above. Let $\{y_{j,i}\}_{j\in[1,k]}$ be the corresponding tuple entries. $\mathcal{B}$ samples $z_{1,i}, \cdots, z_{k,i} \xleftarrow{R} \mathbb{Z}_q$, and responds with:
$$\mathsf{sk}_{\mathsf{id}_i} = \left(\prod_{j=1}^{k}\left(g_j^{x_j} \cdot g_{k+1}^{x_{k+1}}\right)^{y_{j,i} \cdot z_{j,i}}, \{z_{j,i}\}_{j\in[1,k]}\right)$$
Once again, it is again easy to see that this simulation of the key-generation oracle by $\mathcal{B}$ is computationally indistinguishable from the real oracle the experiment $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda)$.

- ***Left-or-Right Query:*** Suppose $\mathcal{A}$ queries the left-or-right oracle with $(\mathbf{ID}_0^*, \mathbf{ID}_1^*)$ - a two-tuple of circuits representing k'-sources over the identity space $\mathcal{ID}_\lambda$ such that $k' = \omega(\log \lambda)$. $\mathcal{B}$ uniformly samples $\mathsf{mode} \xleftarrow{R} \{\mathsf{left}, \mathsf{right}\}$. If $\mathsf{mode} = \mathsf{left}$, it samples $\mathsf{id}^* \xleftarrow{R} \mathbf{ID}_0^*$; otherwise, it samples $\mathsf{id}^* \xleftarrow{R} \mathbf{ID}_1^*$. If any of $H_j(\mathsf{id}^*)$ for $j \in [1, k]$ has already been looked up, $\mathcal{B}$ *outputs a random bit and aborts.* Otherwise, it samples $z_1^*, z_2^*, \cdots, z_k^* \xleftarrow{R} \mathbb{Z}_q$, and responds with:
$$\mathsf{sk}_{\mathsf{id}^*} = \left(\prod_{j=1}^{k+1} g_j^{a_j \cdot x_j}, \{z_j^*\}_{j\in[1,k]}\right)$$
where $\left(g_1^{a_1}, \cdots, g_{k+1}^{a_{k+1}}\right)$ is part of its input k-DLIN instance. It also formally sets $H_j(\mathsf{id}^*) = g^{a_j/z_j^*}$ for $j \in [1, k]$.

- $H_1, \cdots, H_k$ ***Query Phase-2***: $\mathcal{A}$ continues to issue queries to the random oracles for $H_1, H_2 \cdots, H_k$. $\mathcal{B}$ responds as in Phase-1, except if a query for id^* arrives. In this case, $\mathcal{B}$ *outputs* 1 *and aborts.*

- ***Secret-Key Query Phase-2:*** $\mathcal{A}$ continues to issue secret-key queries, and $\mathcal{B}$ continues to respond as in Phase-1, except if a query for id^* arrives. In this case, $\mathcal{B}$ *outputs* 1 *and aborts.*

- ***Output:*** Finally, $\mathcal{A}$ outputs a bit $b \in \{0,1\}$. $\mathcal{B}$ outputs 1 if the challenge identity id^* was sampled from $\mathbf{ID}_b^*$, and 0 otherwise.

Note that once again, we considered the single-shot variant of the function privacy adversary making a single left-or-right oracle query. Such an adversary is polynomially equivalent to its multi-shot variant (see Section 3.4). We now state and prove the following claims:

Claim E.4 *When $a_{k+1} = \sum_{j=1}^{k} a_j$, the joint distribution of* mode *and the challenge secret-key* $\mathsf{sk}_{\mathsf{id}^*}$ *in the simulation of the left-or-right oracle by $\mathcal{B}$ is computationally indistinguishable from that in the experiment* $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda)$.

Proof. Note that the sampling of id^* from either $\mathbf{ID}_1^*$ or $\mathbf{ID}_1^*$ by $\mathcal{B}$ is consistent with its random choice of mode. Additionally, when $a_{k+1} = \sum_{j=1}^{k} a_j$, the secret-key $\mathsf{sk}_{\mathsf{id}^*}$ takes the form:

$$\begin{aligned}
\mathsf{sk}_{\mathsf{id}^*} &= \left(\left(\prod_{j=1}^{k} g_j^{a_j \cdot x_j} \right) \cdot g_{k+1}^{\left(\sum_{j=1}^{k} a_j\right) \cdot x_{k+1}}, \{z_j^*\}_{j\in[1,k]} \right) \\
&= \left(\prod_{j=1}^{k} \left(g_j^{x_j} \cdot g_{k+1}^{x_{k+1}} \right)^{a_j}, \{z_j^*\}_{j\in[1,k]} \right) \\
&= \left(\prod_{j=1}^{k} \left(g_j^{x_j} \cdot g_{k+1}^{x_{k+1}} \right)^{\left(a_j/z_j^*\right)\cdot z_j^*}, \{z_j^*\}_{j\in[1,k]} \right) \\
&= \left(\prod_{j=1}^{k} H_j \left(\mathsf{id}^*\right)^{s_j \cdot z_j^*}, \{z_j^*\}_{j\in[1,k]} \right)
\end{aligned}$$

which is identically distributed to the response of the left-or-right oracle in the experiment $\mathsf{Expt}^{\mathsf{mode}}_{\mathsf{FP},\Pi^{\mathrm{IBE}}_{\mathrm{DBDH+k\text{-}DLIN}},\mathcal{A}}(\lambda)$. This completes the proof of Claim E.4.

Claim E.5 *When a_{k+2} is uniformly random in $\mathbb{Z}_q$, the distribution of the challenge secret-key* $\mathsf{sk}_{\mathsf{id}^*}$ *is statistically independent of $\mathcal{B}$'s choice of* mode.

Proof. Recall that $\alpha_j = \log_g g_j$ for $j \in [1, k+1]$. Consider the following system of equations, determined by the public parameters pp and the secret-key $\mathsf{sk}_{\mathsf{id}^*}$:

$$\begin{aligned}
\log_g \left(g_1^{x_1} \cdot g_{k+1}^{x_{k+1}}\right) &= \alpha_1 \cdot x_1 + \alpha_{k+1} \cdot x_{k+1} \\
\log_g \left(g_2^{x_2} \cdot g_{k+1}^{x_{k+1}}\right) &= \alpha_2 \cdot x_2 + \alpha_{k+1} \cdot x_{k+1} \\
&\vdots \\
\log_g \left(g_k^{x_k} \cdot g_{k+1}^{x_{k+1}}\right) &= \alpha_k \cdot x_k + \alpha_{k+1} \cdot x_{k+1} \\
\log_g \left(\prod_{j=1}^{k+1} g_j^{a_j \cdot x_j}\right) &= \sum_{j=1}^{k+1} a_j \cdot \alpha_j \cdot x_j
\end{aligned}$$

Since a_{k+2} is uniformly random in $\mathbb{Z}_q$, with all but negligible probability, we have that $a_{k+2} \neq \sum_{j=1}^{k+1} a_j$, *which makes the aforementioned system of equations linearly independent.* Hence, the *conditional distribution* of $\mathsf{sk}_{\mathsf{id}^*}$ (where the conditioning is on $\mathcal{B}$'s choice of mode and everything else in $\mathcal{A}$'s view) is uniform. This completes the proof of Claim E.5.

It now follows from Claims E.4 and E.5, that the advantage ϵ' of $\mathcal{B}$ in solving the k-DLIN instance may be quantified as $\epsilon' = \epsilon \cdot (1 - \Pr[\mathsf{abort}_1]) - \Pr[\mathsf{abort}_2]$, where ϵ is the advantage of the function privacy adversary $\mathcal{A}$, while abort_1 and abort_2 denote the events that aborting the simulation during the left-or-right query phase and the second hash/secret-key query phase, respectively, leads to a loss of advantage for $\mathcal{B}$. We bound the probability of this event as follows:

- **Abortion during Left-or-Right Query.** Suppose that the adversary $\mathcal{A}$ makes a maximum of Q_1 and Q_2 queries to the random oracles and the secret-key generation oracle, respectively, prior to the left-or-right query. Recall that both the circuits $\mathbf{ID}_0^*$ and $\mathbf{ID}_1^*$ generated by $\mathcal{A}$ represent k-sources over the identity space $\mathcal{ID}_\lambda$, such that $k = \omega(\log \lambda)$. Hence, the probability that a uniformly randomly sampled id^* from either distribution has already been queried by $\mathcal{A}$ may be upper bounded as $\frac{Q_1 + Q_2}{2^{\omega(\log \lambda)}} \leq \mathsf{negl}(\lambda)$.

- **Abortion during Query Phase-2.** Note that when $\mathcal{B}$ aborts during a random oracle/secret-key generation oracle query in phase-2, it always outputs 1, thereby indicating that its input k-DLIN instance is valid. Hence, a loss of advantage for $\mathcal{B}$ occurs only when it has to abort even if the k-DLIN instance is invalid. Now, as per Claim E.5, when the k-DLIN instance is invalid, the distribution of the challenge secret-key $\mathsf{sk}_{\mathsf{id}^*}$ is uniformly random and independent of mode. Given the min-entropy bounds on the circuits represented by $\mathbf{ID}_0^*$ and $\mathbf{ID}_1^*$, the probability that $\mathcal{A}$ still correctly guesses id^* and issues hash queries for the same is $\mathcal{O}\left(2^{-\omega(\log \lambda)}\right) \leq \mathsf{negl}(\lambda)$.

In summary, we have $\Pr[\mathsf{abort}_\beta] \leq \mathsf{negl}(\lambda)$ for $\beta \in \{1, 2\}$, implying that $\mathcal{B}$'s advantage in solving the k-DLIN instance is negligibly close to the advantage of the function privacy adversary $\mathcal{A}$. This completes the proof of Claim E.3.

F Proof of Theorem 5.1

Let $\mathcal{A}$ be any probabilistic polynomial-time adversary such that:

$$\mathbf{Adv}^{\mathsf{DP}}_{\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathsf{DP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathsf{DP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] \right| = \epsilon$$

where $\epsilon > \mathsf{negl}(\lambda)$. We construct a probabilistic polynomial-time algorithm $\mathcal{B}$ such that:

$$\mathbf{Adv}^{\mathrm{MDDH}}_{l_1,l_2,\log_2 q,\mathcal{B}}(\lambda) = \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{MDDH},l_1,l_2,\log_2 q,\mathcal{B}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{MDDH},l_1,l_2,\log_2 q,\mathcal{B}}(\lambda) = 1\right] \right| = \epsilon$$

$\mathcal{B}$ proceeds as follows:

- ***Init:*** $\mathcal{B}$ outputs a circuit $\mathbf{V}^*$ representing the uniform distribution over $\mathbb{Z}_q^{l_1 \times l_2}$ (indeed, the uniform distribution over $\mathbb{Z}_q^{l_1 \times l_2}$ is a (l_1, l_2, k)-matrix-source for $k = \log_2 q$). In response, $\mathcal{B}$ receives a tuple $\left(g_1^{\mathbf{A}}, g_1^{\mathbf{u}}\right) \in \left(\left(\mathbb{G}_1\right)^{l_1 \times l_2} \times \left(\mathbb{G}_1\right)^{l_1}\right)$, where $\mathbf{u}$ is either of the form $\mathbf{A} \cdot \mathbf{r}$ for some uniformly random $\mathbf{r} \in \mathbb{Z}_q^{l_2}$, or $\mathbf{u}$ is uniformly random in $\mathbb{Z}_q^{l_1}$.

- ***Setup:*** $\mathcal{B}$ samples $\mathbf{S}_0, \mathbf{S}_1, \cdots, \mathbf{S}_n, \mathbf{S}_{n+1}, \cdots, \mathbf{S}_{2n} \xleftarrow{R} \mathbb{Z}_q^{l_2 \times l_1}$. It generates the public parameter pp and the master-secret-key msk as:

$$\mathsf{pp} = \left(g_1, g_1^{\mathbf{A}}, \left\{g_1^{\mathbf{S}_j \cdot \mathbf{A}}\right\}_{j \in [0,2n]}\right) \;,\; \mathsf{msk} = \left(g_2, \{\mathbf{S}_j\}_{j \in [0,2n]}\right)$$

  It provides pp to $\mathcal{A}$.

- ***Secret-Key Queries:*** Suppose $\mathcal{A}$ issues a key-generation query for a predicate matrix $\mathbf{W} = \left[w_{i,j}\right]_{i \in [1,m], j \in [1,n]} \in \mathbb{Z}_q^{m \times n}$. Since $\mathcal{B}$ knows the master-secret-key, it responds with the appropriate secret-key $\mathsf{sk}_{\mathbf{W}}$ as in the real data privacy experiment.

- ***Challenge:*** $\mathcal{A}$ outputs the challenge attribute-pair $(\mathbf{x}_0^*, \mathbf{x}_1^*)$. $\mathcal{B}$ samples $b \xleftarrow{R} \{0,1\}$ and outputs the challenge ciphertext $C^* = \left(\{c_j^*\}_{j \in [0,2n]}\right)$ where:

$$c_0^* = g_1^{\mathbf{u}} \;,\; c_j^* = g_1^{\left(x_{j,b}^* \cdot \mathbf{S}_0 + \mathbf{S}_j\right) \cdot \mathbf{u}} \text{ for } j \in [1, 2n]$$

  where $\mathbf{x}_b^* = \left[x_{1,b}^* \; x_{2,b}^* \cdots x_{n,b}^*\right]^{\mathbf{T}} \in \mathbb{Z}_q^n$ and $x_{n+1,b}^* = \cdots = x_{2n,b}^* = 0$.

- ***Output:*** $\mathcal{A}$ outputs a guess b' for b. If $b' = b$, $\mathcal{B}$ outputs 1, else it outputs 0.

We now state and prove the following claims:

Claim F.1 *When $\mathbf{u}$ is of the form $\mathbf{A} \cdot \mathbf{r}$ for some uniformly random $\mathbf{r} \in \mathbb{Z}_q^{l_2}$, the joint distribution of b and the challenge ciphertext C^* in the simulation by $\mathcal{B}$ is computationally indistinguishable from that in the experiment $\mathsf{Expt}^{(b)}_{\mathsf{DP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda)$.*

Proof. This is obvious from substituting $\mathbf{u} = \mathbf{A} \cdot \mathbf{r}$ in the challenge ciphertext components c_0^* through c_n^*.

Claim F.2 *When* $\mathbf{u}$ *is uniformly random in* $\mathbb{Z}_q^{l_1}$*, the distribution of each challenge ciphertext component* c_j^* *for* $j \in [1, 2n]$ *is statistically independent of* $\mathcal{B}$*'s choice of* b.

Proof. We intend to prove that $c_j^* = g_1^{(x_{j,b}^* \cdot \mathbf{S}_0 + \mathbf{S}_j) \cdot \mathbf{u}}$ is independent of b for each $j \in [1, n]$ (note that $c_{n+1}^*, \cdots, c_{2n}^*$ are independent of b by default). Suppose that during the data privacy experiment, the adversary $\mathcal{A}$ makes n secret-key-generation queries on predicate matrices $\mathbf{W}_1, \cdots, \mathbf{W}_n \in \mathbb{Z}_q^{m \times n}$, and $\mathcal{B}$ responds with $\mathsf{sk}_{\mathbf{W}_l} = \left(\{d_{j,l}\}_{j \in [0,2n]}\right)$ for $l \in [1, n]$. Consider the following system of $(l_2)^2$ equations in $\mathbf{S}_0$ and $\left((2n) \times (l_2)^2\right) + (n \times l_1 \times l_2)$ equations $\{\mathbf{S}_j\}_{j \in [1,2n]}$, determined by the public parameters and the responses of $\mathcal{B}$ to the aforementioned key-generation queries:

$$\log_{g_1}\left(g^{\mathbf{S}_0 \cdot \mathbf{A}}\right) = \mathbf{S}_0 \cdot \mathbf{A}$$

$$\log_{g_1}\left(g^{\mathbf{S}_j \cdot \mathbf{A}}\right) = \mathbf{S}_j \cdot \mathbf{A} \text{ for } j \in [1, 2n]$$

$$\log_{g_2}(d_{0,l}) = \sum_{i=1}^{m} y_{i,l} \cdot \left(\sum_{j=1}^{2n} w_{i,j,l} \cdot \mathbf{S}_j\right) \text{ for } l \in [1, n]$$

Quite evidently, the aforementioned system of equations has exponentially many solutions for $(\mathbf{S}_0, \mathbf{S}_1, \cdots, \mathbf{S}_{2n})$ so long as $l_1 > l_2$.

Let $(\mathbf{S}_0^*, \mathbf{S}_1^*, \cdots, \mathbf{S}_{2n}^*)$ be such a set of solutions, chosen deterministically. Also, let $\mathbf{Z} \in \mathbb{Z}_q^{(l_1 - l_2) \times l_1}$ be a deterministically chosen basis matrix for the kernel space of $\mathbf{A}^{\mathbf{T}}$. Finally, let $\mathbf{Z}' \in \mathbb{Z}_q^{n \times 2n}$ be a second deterministically chosen basis matrix for the kernel space determined by the coefficients corresponding to the system of equations resulting from the n secret-key queries. It is easy to see that $\mathbf{S}_0$ is distributed as follows:

$$\left\{\mathbf{S}_0^* + \boldsymbol{\mu}_0 \cdot \mathbf{Z} \mid \boldsymbol{\mu}_0 \in \mathbb{Z}_q^{l_2 \times (l_1 - l_2)}\right\}$$

while each $\mathbf{S}_j$ for $j \in [1, 2n]$ is distributed as follows:

$$\left\{\mathbf{S}_j^* + \begin{bmatrix} \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{1,1} & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{1,2} & \cdots & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{1,l_1} \\ \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{2,1} & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{2,2} & \cdots & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{2,l_1} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{l_2,1} & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{l_2,2} & \cdots & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{l_2,l_1} \end{bmatrix} \cdot \boldsymbol{\mu}^{\mathbf{T}} \cdot \mathbf{Z} \mid \boldsymbol{\mu} \in \mathbb{Z}_q^{l_1 \times (l_1 - l_2)}, \boldsymbol{\mu}'_{1,1}, \cdots, \boldsymbol{\mu}'_{l_1,l_2} \in \mathbb{Z}_q^n\right\}$$

where $\mathbf{Z}'_j \in \mathbb{Z}_q^{1 \times n}$ is the j^{th} row of the matrix $\mathbf{Z}'$. Now observe that conditional distribution of $\log_{g_1}\left(c_j^*\right)$ for any $j \in [1, n]$ is as follows:

$$\left\{\left(x_{j,b}^* \cdot \left(\mathbf{S}_0^* + \boldsymbol{\mu}_0^{\mathbf{T}} \cdot \mathbf{Z}\right) + \left(\mathbf{S}_j^* + \begin{bmatrix} \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{1,1} & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{1,2} & \cdots & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{1,l_1} \\ \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{2,1} & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{2,2} & \cdots & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{2,l_1} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{l_2,1} & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{l_2,2} & \cdots & \mathbf{Z}'_j \cdot \boldsymbol{\mu}'_{l_2,l_1} \end{bmatrix} \cdot \boldsymbol{\mu}^{\mathbf{T}} \cdot \mathbf{Z}\right)\right) \cdot \mathbf{u}\right\}$$

for $\boldsymbol{\mu}_0, \boldsymbol{\mu} \in \mathbb{Z}_q^{l_2 \times (l_1 - l_2)}$ and $\boldsymbol{\mu}'_{1,1}, \cdots, \boldsymbol{\mu}'_{l_1,l_2} \in \mathbb{Z}_q^n$. Since $\mathbf{u}$ is uniformly random in $\mathbb{Z}_q^{l_1}$, we may assume that $\mathbf{u}$ is not of the form $\mathbf{A} \cdot \mathbf{r}$, since this occurs with only negligible probability. This essentially implies that, except with negligible probability, the aforementioned distribution represents a uniform distribution over $\mathbb{Z}_q^{l_2 \times l_1}$. This completes the proof of Claim F.2.

It now follows from Claims F.1 and F.2 that :

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{MDDH}}_{l_1,l_2,\log_2 q,\mathcal{B}}(\lambda) &= \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{MDDH},l_1,l_2,\log_2 q,\mathcal{B}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{MDDH},l_1,l_2,\log_2 q,\mathcal{B}}(\lambda) = 1\right] \right| \\
&= \left| \Pr\left[\mathsf{Expt}^{(0)}_{\mathsf{DP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathsf{DP},\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = 1\right] \right| \\
&= \mathbf{Adv}^{\mathsf{DP}}_{\Pi^{\mathrm{SME}}_{\mathrm{MDDH}},\mathcal{A}}(\lambda) = \epsilon
\end{aligned}$$

This completes the proof of Theorem 5.1.