On (Hierarchical) Identity Based Encryption Protocols with Short Public Parameters (With an Exposition of Waters' Artificial Abort Technique)

Sanjit Chatterjee and Palash Sarkar

Applied Statistics Unit
Indian Statistical Institute
203, B.T. Road, Kolkata
India 700108.
e-mail:{sanjit_t,palash}@isical.ac.in

Abstract. At Eurocrypt 2005, Waters proposed an efficient identity based encryption (IBE) scheme. One drawback of this scheme is that the size of the public parameter is rather large. Our first contribution is a generalization of Waters scheme. In particular, we show that there is an interesting trade-off between the tightness of the security reduction and smallness of the public parameter size. For a given security level, this implies that if one reduces the public parameter size there is a corresponding increase in the computational cost. This introduces a flexibility in choosing the public parameter size without compromising in security. In concrete terms, to achieve 80-bit security for 160-bit identities we show that compared to Waters protocol the public parameter size can be reduced by almost 90% while increasing the computation cost by 30%. Our second contribution is to extend the IBE protocol to a hierarchical IBE (HIBE) protocol which can be shown to be secure in the full model without the use of random oracle. A previous construction of a HIBE in the same setting is due to Waters. Our construction improves upon Waters' suggestion by significantly reducing the number of public parameters.[1]

1 Introduction

The concept of identity based encryption (IBE) was introduced by Shamir in 1984 [18]. An IBE is a type of public key encryption where the public key can be any binary string. The corresponding secret key is generated by a private key generator (PKG) and provided to the relevant user. The notion of IBE simplifies several applications of public key cryptography. The first efficient implementation and an appropriate security model for IBE was provided by Boneh and Franklin [4].

The PKG issues a private key associated with an identity. The notion of hierarchical identity based encryption (HIBE) was introduced in [13, 12] to reduce the workload of the PKG. An entity in a HIBE structure has an identity which is a tuple $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$. The private key corresponding to such an identity can be generated by the entity whose identity is $(\mathsf{v}_1, \ldots, \mathsf{v}_{j-1})$ and which possesses a private key corresponding to this identity. The security model for IBE was extended to that of HIBE in [13, 12].

The construction of IBE in [4] and of HIBE in [12], was proved to be secure in appropriate models using the *random oracle* heuristic, i.e., the protocols make use of cryptographic hash functions that are modeled as random oracle in the security proof. This led to a search for protocols which can be proved to be secure without random oracle. The first such construction was presented in [7]. Unfortunately, the work in [7] had to relax the notion of security and consider a weaker model called the selective-ID (sID) model. A more efficient construction of (H)IBE secure in the sID model was given by Boneh and Boyen in [2].

The first construction of an IBE which can be proved to be secure in the full model without the random oracle heuristic was given by Boneh and Boyen in [3]. Later, Waters [20] presented an efficient construction of an IBE which is secure in the same setting. However, one disadvantage of the scheme

[1] The material in this paper has appeared in different and abridged forms in [8, 9].

in [20] is the requirement of a rather large public parameter file. If identities are represented by a bit string of length n, then the scheme requires a vector of length n to be maintained as part of public parameter, where each element of the vector is a point on a suitable elliptic curve group.

In the same paper [20], Waters also outlines a construction of a HIBE. The idea is to have a new set of public parameters for each of the h levels of the HIBE. This leads to a system having nh many elliptic curve points as public parameters for an h-level HIBE having n-bit identities at each level.

Our Contributions: We provide a generalization of the identity based encryption scheme of Waters [20]. This generalization shows that if one tries to reduce the public parameter size there is a corresponding degradation in the security reduction. In other words, a trade-off is involved in the tightness of security reduction and smallness of public parameter. The trade-off between tightness and smallness can be converted to a trade-off between group size and smallness of public parameter.

When desiring a specific security level, the loss of security due to loss of tightness in the security reduction can be compensated by working in a larger group. This in turn affects the efficiency of the protocol. Thus, the trade-off is actually between the space required to store the parameters and the time required to execute the protocol. For example, if identities are represented by 160-bit strings, then Waters protocol requires to store 160 extra elements (EC points) as part of the public parameter. Alternatively, using our generalization if one wants to store 16 elements, then to achieve 80-bit security, compared to Waters protocol the space requirement reduces by around 90% of Waters protocol while the computation cost increases by around 30%.

Our second contribution is to extend the IBE protocol to a HIBE protocol. This can be proved to be secure in the full model assuming the decisional bilinear Diffie-Hellman problem to be hard without using the random oracle heuristic. In the same setting, Waters had outlined a HIBE construction based on his IBE construction [20]. Waters' IBE uses U', $U_1, \ldots, U_n$ (and P, P_1, P_2) as public parameters. His suggestion to extend this to a HIBE is to have new public parameters for each level. For an h-level HIBE, the public parameters are of the form $U'_1, U_{1,1}, \ldots, U_{1,n}$, $U'_2, U_{2,1}, \ldots, U_{2,n}$, $\ldots$, $U'_h, U_{h,1}, \ldots, U_{h,n}$. The parameters P, P_1, P_2 are still required giving rise to $3 + (n + 1)h$ many parameters.

The IBE construction given in this paper uses public parameters of the form U', $U_1, \ldots, U_l$ (and P, P_1, P_2) for $1 \leq l \leq n$. For $l = n$, this is Waters' IBE. The HIBE construction in this paper uses public parameters of the form $U'_1, \ldots, U'_h$, $U_1, \ldots, U_l$ for $1 \leq l \leq n$. In other words, the parameters $U'_1, \ldots, U'_h$ correspond to the different levels of the HIBE, whereas the parameters $U_1, \ldots, U_l$ are the same for all the levels. These parameters $U_1, \ldots, U_l$ are reused in the key generation procedure. For $l = n$, we require $3 + n + h$ parameters compared to $3 + (n + 1)h$ parameters in Waters' suggestion.

Thus, our HIBE construction provides two things. First, by reusing public parameters it reduces the size of the public parameters. Second, it extends the flexibility of the IBE protocol of this paper to the HIBE setting. The reuse of public parameters over the different levels of the HIBE complicates the security proof. A straightforward extension of the independence results and lower bound proofs from [20] is not possible. We provide complete proofs of the required results. The constructed HIBE is proved to be secure under chosen plaintext attack (called CPA-secure). Standard techniques [7, 5] can convert such a HIBE into one which is secure against chosen ciphertext attack (CCA-secure).

Note: An independent work by Naccache [17], describes the IBE protocol (but not the HIBE protocol) given in this paper. However, the concrete security analysis of the IBE protocol and in particular the space/time trade-off given in this paper is not present in [17].

2 Preliminaries

The running time of all algorithms in this paper is upper bounded by a polynomial in a security parameter κ. Formally, all algorithms take 1^κ as input. We will be assuming this formalism without explicitly mentioning it.

2.1 HIBE Protocol

Following [13, 12] a HIBE scheme is specified by four probabilistic algorithms: Setup, Key Generation, Encryption and Decryption. Note that, for a HIBE of height h (henceforth denoted as h-HIBE) any identity v is a tuple $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$ where $1 \leq j \leq h$.

Setup: It takes as input a security parameter and returns the system parameters together with the master key. The system parameters include the public parameters of the PKG, a description of the message space, the ciphertext space and the identity space. These are publicly known while the master key is known only to the PKG.

Key Generation: It takes as input an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$, the public parameters of the PKG and the private key $d_{\mathsf{v}|(j-1)}$ corresponding to the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_{j-1})$ and returns a private key d_{v} for v. The identity v is used as the public key while d_{v} is the corresponding private key.

Encryption: It takes as input the identity v, the public parameters of the PKG and a message from the message space and produces a ciphertext in the ciphertext space.

Decryption: It takes as input the ciphertext and the private key of the corresponding identity v and returns the message or bad if the ciphertext is not valid.

Identity Based Encryption: An IBE is a special case of a HIBE where the number of levels h is equal to one.

Key Encapsulation Mechanism: In practice, a public key protocol is almost never used to encrypt the actual message. Instead, a symmetric encryption algorithm is used to encrypt the message and the public key part is used to derive a session key for the symmetric encryption algorithm. The later task is called a key encapsulation mechanism (KEM) and can be combined with the (hierarchical) identity based situation giving rise to (H)IBKEM. We do not explicitly consider KEM in this paper, though we remark that it is not difficult to convert the (H)IBE encryption protocols described in this paper into (H)IBKEM protocols.

2.2 Security Model for HIBE

Security is defined using an adversarial game. An adversary $\mathcal{A}$ is allowed to query two oracles – a decryption oracle and a key-extraction oracle. At the initiation, it is provided with the public parameters of the PKG. The game has two query phases with a challenge phase in between.

Query Phase 1: Adversary $\mathcal{A}$ makes a finite number of queries where each query is addressed either to the decryption oracle or to the key-extraction oracle. In a query to the decryption oracle it provides a ciphertext as well as the identity under which it wants the decryption. It gets back the corresponding message or bad if the ciphertext is invalid. Similarly, in a query to the key-extraction oracle, it asks for the private key of the identity it provides and gets back this private key. Further, $\mathcal{A}$ is allowed to make

these queries adaptively, i.e., any query may depend on the previous queries as well as their answers. The adversary is not allowed to make any useless queries, i.e., queries for which it can compute the answer itself. For example, the adversary is not allowed to ask for the decryption of a message under an identity if it has already obtained a private key corresponding to the identity.

Challenge: At this stage, $\mathcal{A}$ outputs an identity $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_j^*)$ for $1 \leq j \leq h$, and a pair of equal length messages M_0 and M_1. There is the natural restriction on the adversary, that it cannot query the key extraction oracle on v^* or any of its proper prefixes in either of the phases 1 or 2. A random bit γ is chosen and the adversary is provided with C^* which is an encryption of M_γ under v^*.

Query Phase 2: $\mathcal{A}$ now issues additional queries just like Phase 1, with the (obvious) restriction that it cannot ask the decryption oracle for the decryption of C^* under v^*.

Guess: $\mathcal{A}$ outputs a guess γ' of γ.
The advantage of the adversary $\mathcal{A}$ is defined as:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{HIBE}} = |\mathsf{Pr}[(\gamma = \gamma')] - 1/2|.$$

The quantity $\mathsf{Adv}^{\mathsf{HIBE}}(t, q_{\mathsf{ID}}, q_{\mathsf{C}})$ denotes the maximum of $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{HIBE}}$ where the maximum is taken over all adversaries running in time at most t and making at most q_{C} queries to the decryption oracle and at most q_{ID} queries to the key-extraction oracle. A HIBE protocol is said to be $(\epsilon, t, q_{\mathsf{ID}}, q_{\mathsf{C}})$-CCA secure if $\mathsf{Adv}^{\mathsf{HIBE}}(t, q_{\mathsf{ID}}, q_{\mathsf{C}}) \leq \epsilon$.

In the above game, we can restrict the adversary $\mathcal{A}$ from querying the decryption oracle. $\mathsf{Adv}^{\mathsf{HIBE}}(t, q)$ in this context denotes the maximum advantage where the maximum is taken over all adversaries running in time at most t and making at most q queries to the key-extraction oracle. A HIBE protocol is said to be (t, q, ϵ)-CPA secure if $\mathsf{Adv}^{\mathsf{HIBE}}(t, q) \leq \epsilon$.

As mentioned earlier there are generic techniques [7, 5] for converting a CPA-secure HIBE into a CCA-secure HIBE. In view of these techniques, we will concentrate only on CPA-secure HIBE.

Security Model for IBE: The security model for IBE is derived from the security model for HIBE by simply allowing only one level in the hierarchy.

2.3 Cryptographic Bilinear Map

Let G_1 and G_2 be cyclic groups having the same prime order p and $G_1 = \langle P \rangle$, where we write G_1 additively and G_2 multiplicatively. A mapping $e : G_1 \times G_1 \rightarrow G_2$ is called a cryptographic bilinear map if it satisfies the following properties.

- Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1$ and $a, b \in \mathbb{Z}_p$.
- Non-degeneracy: If $G_1 = \langle P \rangle$, then $G_2 = \langle e(P, P) \rangle$.
- Computability: There exists an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.

Since $e(aP, bP) = e(P, P)^{ab} = e(bP, aP)$, $e()$ also satisfies the symmetry property. The modified Weil pairing [4] and Tate pairing [1, 10] are examples of cryptographic bilinear maps.

Note: Known examples of $e()$ have G_1 to be a group of Elliptic Curve (EC) points and G_2 to be a subgroup of a multiplicative group of a finite field. Hence, in papers on pairing implementations [1, 10], it is customary to write G_1 additively and G_2 multiplicatively. On the other hand, some "pure" protocol papers [2, 3, 20] write both G_1 and G_2 multiplicatively though this is not true for the initial protocol papers [14, 4]. Here we follow the first convention as it is closer to the known examples of cryptographic bilinear map.

2.4 Hardness Assumption

The decisional bilinear Diffie-Hellman (DBDH) problem in $\langle G_1, G_2, e\rangle$ [4] is as follows: Given a tuple $\langle P, aP, bP, cP, Z\rangle$, where $Z \in G_2$, decide whether $Z = e(P,P)^{abc}$ (which we denote as Z is real) or Z is random.

The advantage of a probabilistic algorithm $\mathcal{B}$, which takes as input a tuple $\langle P, aP, bP, cP, Z\rangle$ and outputs a bit, in solving the DBDH problem is defined as

$$\begin{aligned}\mathsf{Adv}_{\mathcal{B}}^{\mathsf{DBDH}} = |&\mathsf{Pr}[\mathcal{B}(P, aP, bP, cP, Z) = 1 | \mathrm{Z\ is\ real}] \\ &-\mathsf{Pr}[\mathcal{B}(P, aP, bP, cP, Z) = 1 |\ \mathrm{Z\ is\ random}]|\end{aligned}$$

where the probability is calculated over the random choices of $a, b, c \in \mathbb{Z}_p$ as well as the random bits used by $\mathcal{B}$. The quantity $\mathsf{Adv}^{\mathsf{DBDH}}(t)$ denotes the maximum of $\mathsf{Adv}_{\mathcal{B}}^{\mathsf{DBDH}}$ where the maximum is taken over all adversaries $\mathcal{B}$ running in time at most t. By the (ϵ, t)-DBDH assumption we mean $\mathsf{Adv}^{\mathsf{DBDH}}(t) \leq \epsilon$.

2.5 Waters' IBE Protocol

Let $G_1 = \langle P\rangle$, G_2 and $e()$ be as defined in Section 2.3. Identities are n-bit strings. The groups $G_1 = \langle P\rangle$, G_2 and the map $e()$ are as already defined in Section 2.3. In the following, we assume the message space $\mathcal{M}$ is G_2, the cipher space $\mathcal{C}$ is $G_2 \times G_1 \times G_1$.

Setup: Randomly choose a secret $\alpha \in \mathbb{Z}_p$. Set $P_1 = \alpha P$, then choose $P_2 \in G_1$ at random. Further, choose random elements $U', U_1, \ldots, U_n$ from G_1. The master secret is αP_2 whereas the public parameters are $\langle P, P_1, P_2, U', U_1, \ldots, U_n\rangle$.

Key Generation: Let $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_n) \in \{0,1\}^n$ be any identity. A secret key for v is generated as follows. Choose a random $r \in \mathbb{Z}_p^*$, then the private key for v is

$$D_{\mathsf{v}} = (\alpha P_2 + rV, rP)$$

where $V = U' + \sum_{\{i:\mathsf{v}_i=1\}} U_i$.

Encryption: Any message $M \in G_2$ is encrypted for an identity v as

$$C = (e(P_1, P_2)^t M, tP, tV),$$

where t is a random element of $\mathbb{Z}_p$ and V is as defined in key generation algorithm.

Decryption: Let $C = (C_1, C_2, C_3)$ be a ciphertext and v be the corresponding identity. Then we decrypt C using secret key $D_{\mathsf{v}} = (D_1, D_2)$ by computing $C_1 e(D_2, C_3)/e(D_1, C_2)$.

3 Generalization of Waters' IBE

In Waters' scheme identities are represented as n-bit strings. Because of this representation, Waters requires to store n elements of G_1 i.e., $U_1, \ldots, U_n$ in the public parameter. Depending upon the choice of representation of the identities we can change the size of the public parameter.

Let $N = 2^n$, then we can consider the identities as elements of $\mathbb{Z}_N$ and one extreme case would be to consider the identities just as elements of $\mathbb{Z}_N$. A more moderate approach, however, is to fix *a-priori* a size parameter l, where $1 < l \leq n$. In this case, an identity v is represented as $\mathsf{v} = (\mathsf{v}_1, \mathsf{v}_2, \ldots, \mathsf{v}_l)$, where $\mathsf{v}_i \in \mathbb{Z}_{N^{1/l}}$ i.e., each v_i is an n/l bit string. (If identities are considered to be bit strings of arbitrary length, then as in Waters protocol we hash them into $\mathbb{Z}_N$ using a collision resistant hash function.) In this case, the protocol is changed to the following, which we call $\mathsf{IBE\text{-}SPP}(l)$.

3.1 Protocol IBE-SPP(l) with $1 < l \leq n$

Setup: Randomly choose a secret $\alpha \in \mathbb{Z}_p$. Set $P_1 = \alpha P$, then choose $P_2 \in G_1$ at random. Further, choose random elements $U', U_1, U_2, \ldots, U_l \in G_1$. The master secret is αP_2 whereas the public parameters are $\langle P, P_1, P_2, U', U_1, U_2, \ldots, U_l \rangle$.

A Useful Shorthand: Let $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_l))$ be an identity. We define

$$V(\mathsf{v}) = U' + \sum_{i=1}^{l} \mathsf{v}_i U_i. \tag{1}$$

When v is clear from the context we simply write V to denote $V(\mathsf{v})$.

Key Generation: Let v be any identity, a secret key for v is generated as follows. Choose a random $r \in \mathbb{Z}_p^*$, then the private key for v is

$$d_{\mathsf{v}} = (\alpha P_2 + rV, rP)$$

where V is as defined in Equation (1).

Encryption: Any message $M \in G_2$ is encrypted for an identity v as

$$C = (e(P_1, P_2)^s \times M, sP, sV),$$

where s is a random element of $\mathbb{Z}_p$ and V is as defined in Equation (1).

Decryption: Let $C = (C_1, C_2, C_3)$ be a ciphertext and v be the corresponding identity. Then we decrypt C using secret key $d_{\mathsf{v}} = (d_1, d_2)$ by computing

$$C_1 \times \frac{e(d_2, C_3)}{e(d_1, C_2)} = e(P_1, P_2)^s \times M \frac{e(rP, sV)}{e(\alpha P_2 + rV, sP)} = M.$$

Note that, for $l = n$ this is exactly Waters protocol.

3.2 Efficiency

Consider IBE-SPP(l) with $1 < l \leq n$. Let $\mathsf{cost}(V)$ be the cost of computing V. The cost of key generation is two scalar multiplications over G_1 plus $\mathsf{cost}(V)$. By including $e(P_1, P_2)$ instead of P_1, P_2 in the public parameter, we can avoid the pairing computation during encryption. So the cost of encryption is one exponentiation over G_2, two scalar multiplications over G_1 plus $\mathsf{cost}(V)$. The cost of decryption is two pairings, one multiplication and one inversion over G_2. The effect of l is in $\mathsf{cost}(V)$ and affects key generation and encryption costs but does not affect decryption cost.

We first consider the costs of scalar multiplication over G_1 and exponentiation over G_2. As mentioned earlier, G_1 is an elliptic curve group. Let $\mathbb{F}_a$ denote the base field over which G_1 is defined. Then G_2 is a subgroup of $\mathbb{F}_{a^k}$, where k is the MOV degree. Additions and doublings over G_1 translate into a constant number of multiplications over $\mathbb{F}_a$. The actual number is slightly different for addition and doubling, but we will ignore this difference. Let $|\mathbb{F}_a|$ be the size of the representation of an element of $\mathbb{F}_a$. Assuming the cost of multiplication over G_1 is approximately equal to $|\mathbb{F}_a|^2$, the cost of a scalar multiplication over G_1 is equal to $c_1|\mathbb{F}_a|^3$ for some constant c_1. One can also show that the cost of exponentiation

over G_2 is equal to $c_2|\mathbb{F}_a|^3$. Thus, the total cost of scalar multiplication and exponentiation is equal to $c|\mathbb{F}_a|^3$.

The cost of computing V amounts to computing l scalar multiplications where each multiplier is an (n/l)-bit string. On an average, the cost of each such multiplication will be $n/2l$ additions and $(n/l-1)$ doublings over G_1. Hence, the total cost of computing V is $n/2$ additions and $(n-l)$ doublings over G_1. This cost is equal to $d(3/2-l/n)n|\mathbb{F}_a|^2$ for some constant d.

We consider the cost of encryption. The total cost is

$$c|\mathbb{F}_a|^3 + d(3/2-l/n)n|\mathbb{F}_a|^2 = \left(c + d \times \frac{n}{|\mathbb{F}_a|}\left(\frac{3}{2}-\frac{l}{n}\right)\right)|\mathbb{F}_a|^3. \tag{2}$$

This cost is minimum when $l = n$ (as in Waters protocol). The maximum value of the coefficient of $|\mathbb{F}_a|^3$ is $(c + (3nd)/(2|\mathbb{F}_a|))$ whereas the minimum value is $(c + (nd)/(2|\mathbb{F}_a|))$. The value of $|\mathbb{F}_a|$ is usually greater than n and hence the value of $(nd)/(2|\mathbb{F}_a|)$ will be a small constant and hence there is not much effect of l on the total cost of encryption. A similar analysis shows that the effect of l is also not very significant on the cost of key generation. We note, however, that key generation is essentially a one-time offline activity.

3.3 Security Reduction

The CPA-security of the identity based encryption scheme (IBE-SPP(l)) developed above can be reduced from the hardness of the DBDH problem as stated in the following theorem.

Theorem 1. *Protocol* IBE-SPP(l) *of Section 3.1 is* (ϵ_{ibe}, t, q)*-CPA secure assuming that the* (t', ϵ_{dbdh})*-DBDH assumption holds in* $\langle G_1, G_2, e\rangle$*, where* $\epsilon_{ibe} \leq 2\epsilon_{dbdh}/\lambda$*;* $t' = t + O(\tau q) + \chi(\epsilon_{ibe})$ *and*

$\chi(\epsilon) = O(\epsilon^{-2}\ln(\epsilon^{-1})\lambda^{-1}\ln(\lambda^{-1}))$*;*
τ *is the time required for one scalar multiplication in* G_1*;*
$\lambda = 1/(8q(\mu_l+1))$ *with* $\mu_l = l(N^{1/l}-1)$*,* $N = 2^n$*.*

Theorem 1 is a special case of the more general result for HIBE stated in Theorem 2 and is obtained from it by substituting $h = 1$ (number of levels) and $\sigma = 2q$ (i.e., $2q \geq 2^{n/l}$). The essential part of Theorem 1 is the relation between ϵ_{dbdh} and ϵ_{ibe}. This is analysed in details after Theorem 2. Interpreting this analysis in the context of Theorem 1, we obtain $\epsilon_{ibe} \leq 16lq2^{n/l}\epsilon_{dbdh}$. This shows that there is a security degradation by a factor of $16lq2^{n/l}$. In Section 4, we analyse in details the implications of this security degradation.

3.4 Signature

It is an observation of Naor that any identity-based encryption scheme can be converted to a signature scheme. Waters in his paper [20] has given a construction of a signature scheme based on his IBE scheme. A similar construction is possible for the generalised scheme IBE-SPP(l). This signature scheme can be proved to be secure assuming that the CDH problem is hard in G_1. The concrete security analysis performed in Section 4 also holds for the signature protocol.

Let $G_1 = \langle P\rangle$, G_2 and $e()$ be as defined in Section 2.3. Messages are assumed to be elements of $\mathbb{Z}_N$ where $N = 2^n$. Alternatively, if messages are assumed to be bit strings of arbitrary length, then we use a collision resistant hash function to map the messages into $\mathbb{Z}_N$.

Setup: Choose a random α in $\mathbb{Z}_p$. Let $P_1 = \alpha P$. Next, choose random points $P_2, U', U_1, \ldots, U_l$ from G_1. The public key is $\langle P, P_1, P_2, U', U_1, \ldots, U_l\rangle$ and the secret key is αP_2.

Signing: Let $M = (m_1, m_2, \ldots, m_l)$ be the message to be signed, where each $m_i, 1 \leq i \leq l$ is a bit string of length n/l. To generate a signature on M, first choose a random $r \in \mathbb{Z}_p^*$. Then the signature is

$$\sigma_M = (\alpha P_2 + rV, rP),$$

where $V = V(M)$ is as defined in Equation (1).

Verification: Given a message $M = (m_1, m_2, \ldots, m_l)$ and a signature $\sigma = (\sigma_1, \sigma_2)$ on M, one accepts σ as a valid signature on M if

$$e(\sigma_1, P) = e(P_1, P_2)e(\sigma_2, V)$$

where $V = V(M)$ is as defined in Equation (1).

4 Concrete Security

From the security reduction of previous section we observe that any (t, q, ϵ) adversary $\mathcal{A}$ against IBE-SPP(l) can actually be used to build an algorithm $\mathcal{B}$ to solve the DBDH problem over (G_1, G_2, e) which runs in time t' and has a probability of success ϵ'. Then $t' = t + c\tau q + \chi(\epsilon) \approx t + c\tau q + \chi'$ for some constant c and $\epsilon' \approx \epsilon/\delta$ where τ is the time for a group operation in G_1 and δ is the corresponding degradation in the security reduction. Resistance of IBE-SPP(l) against $\mathcal{A}$ can be quantified as $\rho_{|\mathcal{A}}^{(l)} = \lg(t/\epsilon)$. To assert that IBE-SPP(l) has at least 80-bit security, we must have $\rho_{|\mathcal{A}}^{(l)} \geq 80$ for all possible $\mathcal{A}$. Similarly, the resistance of DBDH against $\mathcal{B}$ can be quantified as

$$\rho_{|\mathcal{B}} = \lg\left(\frac{t'}{\epsilon'}\right) \approx \lg\left(\delta \times \frac{t + c\tau q + \chi'}{\epsilon}\right) = \lg(\delta(A_1 + A_2))$$

where $A_1 = t/\epsilon$ and $A_2 = (c\tau q + \chi')/\epsilon$. We now use $\max(A_1, A_2) \leq A_1 + A_2 \leq 2\max(A_1, A_2)$. Since a factor of two does not significantly affect the analysis we put $\rho_{|\mathcal{B}} = \lg(\delta \times \max(A_1, A_2))$. By our assumption, $A_1 = t/\epsilon \geq 2^{80}$ and hence $\max(A_1, A_2) \geq A_1 \geq 2^{80}$. This results in the condition $\rho_{|\mathcal{B}} \geq 80 + \lg\delta$.

Thus, if we want IBE-SPP(l) to have 80-bit security, then we must choose the group sizes of G_1, G_2 in such a way that the best possible algorithm for solving DBDH in these groups takes time at least $2^{80+\lg\delta}$. Hence, in particular, the currently best known algorithm for solving the DBDH should also take this time. Currently the only method to solve the DBDH problem over (G_1, G_2, e) is to solve the discrete log problem (DLP) over either G_1 or G_2. The best known algorithm for the former is the Pollard's rho method while that for the later is number/function field sieve. Thus, if we want IBE-SPP(l) to have 80-bit security, then we must choose the group sizes such that, $2^{80+\lg\delta} \leq \mathsf{min}(t_{G_1}, t_{G_2})$, where t_{G_i} stands for the time to solve DLP in G_i for $i \in \{1, 2\}$.

We have assumed that G_1 is a group of elliptic curve points of order p defined over a finite field $\mathbb{F}_a$ (a is a prime power). Suppose G_2 is a subgroup of order p of the finite field $\mathbb{F}_{a^k}$ where k is the embedding degree. The Pollard's rho algorithm to solve ECDLP takes time $t_{G_1} = O(\sqrt{p})$, while the number/function field seive method to solve the DLP in $\mathbb{F}_{a^k}$ takes time $t_{G_2} = O(e^{c^{1/3}\ln^{1/3}a^k \ln^{2/3}(\ln a^k)})$ where $c = 64/9$ (resp. $32/9$) in large characteristic fields (resp. small characteristic fields).

4.1 Space/time trade-off

In this section, we parametrize the quantities by l wherever necessary. Let, $\delta^{(l)}$ denote the degradation factor in IBE-SPP(l). We have already noted in Section 3.1 that $l = n$ stands for Waters protocol. $\delta^{(l)}$ and hence $\rho^{(l)}$ is minimum when $l = n$ and we use this as a benchmark to compare with other values

Table 1. *Approximate group sizes for attaining 80-bit security for* IBE-SPP(l) *for different values of* l *and relative space and time requirement. The first part corresponds to* $n = 160$ *and the second to* $n = 256$.

l	$\Delta\rho^{(l)}$	$\lvert p^{(l)}\rvert$	$\lvert G_2^{(l)}\rvert$		$\alpha^{(l)}$		$\beta^{(l)}$	
			(a)	(b)	(a)	(b)	(a)	(b)
160	–	246	1891(2225)	3284(3872)	–	–	–	–
8	15	276	2443(2831)	4258(4944)	6.5(6.4)	6.5(6.4)	2.16(2.06)	2.18(2.08)
16	6	258	2102(2457)	3655(4288)	11.1(11.0)	11.1(11.1)	1.37(1.35)	1.38(1.35)
32	2	250	1960(2300)	3405(4006)	20.7(20.7)	20.7(20.7)	1.11(1.11)	1.12(1.11)
80	1	248	1924(2262)	3344(3939)	50.9(50.8)	50.9(50.9)	1.05(1.05)	1.06(1.05)
256	–	246	1891(2225)	3284(3872)	–	–	-	–
8	27	300	2948(3381)	5151(5919)	4.9(4.7)	4.9(4.8)	3.79(3.51)	3.86(3.57)
16	12	270	2326(2703)	4051(4717)	7.7(7.6)	7.7(7.6)	1.86(1.79)	1.88(1.81)
32	5	256	2066(2417)	3592(4212)	13.7(13.6)	13.7(13.6)	1.30(1.28)	1.31(1.29)
64	2	250	1960(2300)	3405(4006)	25.9(25.8)	25.9(25.9)	1.11(1.11)	1.11(1.11)
128	1	248	1924(2262)	3344(3939)	50.9(50.8)	50.9(50.9)	1.05(1.05)	1.06(1.05)

of l. Suppose $\Delta\rho^{(l)} = \rho^{(l)} - \rho^{(n)} = \lg(\delta^{(l)}/\delta^{(n)}) = (n/l) - \lg(n/l)$. This parameter $\Delta\rho^{(l)}$ gives us an estimate of the extra bits required in case of IBE-SPP(l), to achieve the same security level as that of IBE-SPP(n) i.e., Waters protocol.

Suppose, $|p^{(l)}|$ (resp. $|G_2^{(l)}|$) denotes the bit length of representation of $p^{(l)}$ (resp. an element of $G_2^{(l)}$). Like [11], we assume that the adversary $\mathcal{A}$ is allowed to make a maximum of $q = 2^{30}$ number of queries. For a given security level, we can now find the values of $|p^{(l)}|$ and $|G_2^{(l)}|$ for IBE-SPP(l) based on the bit length of the identities (i.e., n), q and l. Note that, the value of $|p^{(l)}|$ (resp. $|G_2^{(l)}|$) thus obtained is the *minimum* required to avoid the Pollard's rho (resp. number/function field seive) attack. In actual implementation, these values give an estimate of the size of suitable pairing groups G_1, G_2 and the embedding degree so that the requirements can be optimally met. In our comparison, the embedding degree k is taken to be same for different values of l and $|G_2^{(l)}| = k \lg a$ ($G_2^{(l)}$ is a multiplicative subgroup of order $p^{(l)}$ of the finite field $\mathbb{F}_{a^k}$). As already noted, the value of $p^{(l)}$ is given by Pollard's rho. On the other hand, the logarithm of the size of $G_1^{(l)}$ is equal to $\mathsf{max}(p^{(l)}, |G_2^{(l)}|/k)$. For relatively small embedding degree (i.e., $k \leq 6$), $|G_2^{(l)}|/k > |p^{(l)}|$ and so the logarithm of the size of $G_1^{(l)}$ is equal to $|G_2^{(l)}|/k = |\mathbb{F}_a^{(l)}|$. For a given l, we have to store l elements of $G_1^{(l)}$ in the public parameter file and a scalar multiplication in $G_1^{(l)}$ takes time proportional to $(|\mathbb{F}_a^{(l)}|)^3$.

Now, we are in a position to compare the space requirement in the public parameter file and the time requirement for a scalar multiplication in $G_1^{(l)}$ for different values of l. Let $\alpha^{(l)} = \frac{l \times |G_1^{(l)}|}{n \times |G_1^{(n)}|} \times 100$ i.e., the relative amount of space (expressed in percentage) required to store the public parameters in case of IBE-SPP(l) with respect to IBE-SPP(n) and $\beta^{(l)} = |\mathbb{F}_a^{(l)}|^3/|\mathbb{F}_a^{(n)}|^3$, i.e., the relative increase in time for scalar multiplication in $G_1^{(l)}$in the case of IBE-SPP(l) with respect to IBE-SPP(n). Note that, $\beta^{(l)}$ can be computed from $|G_2^{(l)}|$ and $|G_2^{(n)}|$ since k cancels out from both numerator and denominator. The pairing computation is also of order $|\mathbb{F}_a^{(l)}|^3$ (but with a larger constant factor). So, the ratio $\beta^{(l)}$ also holds for pairing computation and exponentiation in case of IBE-SPP(l) with respect to Waters protocol.

In Table 1, we sum-up these results for $n = 160$ and 256 for different values of l ranging from 8 to n for 80-bit security. The subcolumns (a) and (b) under $\alpha^{(l)}$ and $\beta^{(l)}$ stand for the values obtained for general characteristic field and field of characteristic three respectively. The values of $|G_2^{(l)}|, \alpha^{(l)}, \beta^{(l)}$ are computed using the formula as suggested in [11] (see Section 3); while in parenthesis we give the

corresponding values as computed from the formula obtained from [15] (as given in Section 3 of [11]). Note that, the values of $\alpha^{(l)}$ and $\beta^{(l)}$ being the ratio of two quantities remain more or less invariant whether the underlying field is a general characteristic field or a field of characteristic three or which formula (of [11] or of [15]) is used.

Public parameter consists of $(l + 4)$ elements of G_1. From Table 1, for 80-bit security in general characteristic fields using EC with MOV degree 2, the public parameter size for Waters protocol will be around 37 kilobyte (kb) for 160-bit identities and 59 kb for 256-bit identities. The corresponding values in case of $\mathsf{IBE\text{-}SPP}(l)$ with $l = 16$ will be around 4 kb and 4.5 kb respectively. Similarly, in characteristic three field EC with MOV degree 6, the corresponding values are respectively 21.5 kb and 34.2 kb and for $\mathsf{IBE\text{-}SPP}(l)$ with $l = 16$ these are respectively 2.4 kb and 2.64 kb. There is an associated increase in computation cost by 30%. In typical applications, the protocol will be used in a key encapsulation mechanism (KEM). Thus the encryption and decryption algorithms will be invoked once for a message irrespective of its length. Also the key generation procedure is essentially a one-time offline activity. In view of this, the increase in computation cost will not substantially affect the throughput. On the other hand, the significant reduction in space requirement will be an advantage in implementing the protocol and also in reducing the time for downloading or transmitting the public parameter file over the net. Overall, we suggest $l = 16$ to be a good choice for implementing the protocol.

5 HIBE Construction

The IBE protocol of Waters (see Section 2.5) has some similarities with the 1-level (H)IBE scheme of Boneh-Boyen [2]. Waters in his paper [20] has suggested that this similarity can be utilized to build a HIBE in an obvious manner, i.e., for each level we have to generate new parameters. This makes the public parameters quite large – for a HIBE of height h with n-bit identities, the number of public parameters becomes $n \times h$.

In this section, we suggest an alternative construction where the public parameters can be significantly reduced. We base our protocol on the generalization of Waters' protocol presented in Section 3.1 where each n-bit identity is represented by l blocks of n/l bits each. We show that for a h-HIBE it suffices to store $(l + h)$ elements in the public parameter. If a similar representation is used for Waters' suggestion then the public parameter size would be $l \times h$.

The identities are of the type $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$, for $j \in \{1, \ldots, h\}$ where each $\mathsf{v}_k = (\mathsf{v}_{k,1}, \ldots, \mathsf{v}_{k,l})$ and $\mathsf{v}_{k,i}$ is an (n/l)-bit string which will also be considered to be an integer in the set $\{0, \ldots, 2^{n/l} - 1\}$. Choosing $l = n$ gives v_k to be an n-bit string as considered by Waters [20].

Set-Up: The protocol is built from groups G_1, G_2 and a bilinear map e as mentioned in Section 2.3. The public parameters are the following elements: P, $P_1 = \alpha P$, P_2, $U'_1, \ldots, U'_h$, $U_1, \ldots, U_l$, where $G_1 = \langle P \rangle$, α is chosen randomly from $\mathbb{Z}_p$ and the other quantities are chosen randomly from G_1.

The master secret is αP_2. (The quantities P_1 and P_2 are not directly required; instead $e(P_1, P_2)$ is required. Hence one may store $e(P_1, P_2)$ as part of the public parameters instead of P_1 and P_2.)

A Useful Shorthand: Let $v = (v_1, \ldots, v_l)$, where each v_i is an (n/l)-bit string and is considered to be an element of $\mathbb{Z}_{2^{n/l}}$. For $1 \le k \le h$ we define,

$$V_k(v) = U'_k + \sum_{i=1}^{l} v_i U_i. \tag{3}$$

When v is clear from the context we will write V_k instead of $V_k(v)$. The modularity introduced by this notation allows an easier understanding of the protocol.

Note that for the jth level of the HIBE, we add a single element, i.e., U'_j in the public parameter while the elements $U_1, \ldots, U_l$ are re-used for each level. This way we are able to shorten the public parameter size. Later in the security reduction we show that the simulator forms U'_js, $1 \leq j \leq h$ in such a way that it is able to answer the adversarial queries.

Key Generation: Let $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$, $j \leq h$, be the identity for which the private key is required. Choose $r_1, \ldots, r_j$ randomly from $\mathbb{Z}_p$ and define $d_{\mathsf{v}} = (d_0, d_1, \ldots, d_j)$ where

$$d_0 = \alpha P_2 + \sum_{k=1}^{j} r_k V_k(\mathsf{v}_k)$$

and $d_k = r_k P$ for $1 \leq k \leq j$.

Key delegation can be done in the manner shown in [2]. Suppose $(d'_0, d'_1, \ldots, d'_{j-1})$ is a private key for the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_{j-1})$. To generate a private key for v, first choose r_j randomly from $\mathbb{Z}_p$ and compute d_{v} as follows.

$$\begin{aligned} d_0 &= d'_0 + r_j V_j(\mathsf{v}_j); \\ d_i &= d'_i \qquad 1 \leq i \leq j-1; \\ d_j &= r_j P. \end{aligned}$$

Encryption: Let $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ be the identity under which a message $M \in G_2$ is to be encrypted. Choose t to be a random element of $\mathbb{Z}_p$. The ciphertext is

$$(C_0 = M \times e(P_1, P_2)^t, C_1 = tP, B_1 = tV_1(\mathsf{v}_1), \ldots, B_j = tV_j(\mathsf{v}_j)).$$

Decryption: Let $C = (C_0, C_1, B_1, \ldots, B_j)$ be a ciphertext and the corresponding identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$. Let $(d_0, d_1, \ldots, d_j)$ be the decryption key corresponding to the identity v. The decryption steps are as follows.

Verify whether C_0 is in G_2, C_1 and the B_i's are in G_1. If any of these verifications fail, then return bad, else proceed with further decryption as follows. Compute $V_1(\mathsf{v}_1), \ldots, V_j(\mathsf{v}_j)$. Return

$$C_0 \times \frac{\prod_{k=1}^{j} e(B_i, d_i)}{e(d_0, C_1)}.$$

It is standard to verify the consistency of decryption.

Note: If $h = 1$, then this reduces to the IBE protocol of Section 3.1.

6 Security of the HIBE Construction

In this section, we state the result on security and discuss its implications. The proof is given in Section 7.

Theorem 2. *The HIBE protocol described in Section 5 is (ϵ_{hibe}, t, q)-CPA secure assuming that the (t', ϵ_{dbdh})-DBDH assumption holds in $\langle G_1, G_2, e\rangle$, where $\epsilon_{hibe} \leq 2\epsilon_{dbdh}/\lambda$; $t' = t + O(\tau q) + \chi(\epsilon_{hibe})$ and*

$\chi(\epsilon) = O(\epsilon^{-2} \ln(\epsilon^{-1}) \lambda^{-1} \ln(\lambda^{-1}))$;
τ is the time required for one scalar multiplication in G_1;
$\lambda = 1/(2(2\sigma(\mu_l + 1))^h)$ with $\mu_l = l(N^{1/l} - 1)$, $N = 2^n$ and $\sigma = \max(2q, 2^{n/l})$.

We further assume $2\sigma(1 + \mu_l) < p$.

The last assumption is practical and similar assumptions are also made in [20, 17], though not quite so explicitly. Before proceeding to the proof, we discuss the above result. The main point of the theorem is the bound on ϵ_{hibe}. This is given in terms of λ and in turn in terms of μ_l. We simplify this bound.

Since $l \geq 1$, we have $1 + \mu_l = 1 + l(N^{1/l} - 1) \leq lN^{1/l} = l2^{n/l}$. Consequently,

$$\begin{aligned} \epsilon_{hibe} &\leq \frac{2\epsilon_{dbdh}}{\lambda} = 4(2\sigma(\mu_l + 1))^h \epsilon_{dbdh} \\ &\leq 4(2\sigma l 2^{n/l})^h \epsilon_{dbdh} \\ &= 4(2l2^{n/l})^h \sigma^h \epsilon_{dbdh} \end{aligned} \tag{4}$$

The reduction is not tight; security degrades by a factor of $4(2l2^{n/l})^h\sigma^h$. We now consider several cases. The actual value of degradation depends on the value of q, the number of key extraction queries made by the adversary. A value of q used in earlier analysis is $q = 2^{30}$ [11]. We will use this value of q in the subsequent analysis.

$h = 1$ and $l = n$: The value of $h = 1$ implies that the HIBE is actually an IBE and $l = n$ implies that each identity is a bit vector of length n. This is the situation originally considered by Waters [20]. In this case, $2q = \max(2q, 2^{n/l})$ and Equation (4) reduces to $\epsilon_{hibe} \leq 32nq\epsilon_{dbdh}$. For $n = 160$, the degradation is by a factor of 10×2^{38}.

$h > 1$: This corresponds to a proper HIBE. If $l = n$, then we obtain $\epsilon_{hibe} \leq 4(8nq)^h\epsilon_{dbdh}$. For $n = 160$ (and $q = 2^{30}$), this amounts to $\epsilon_{hibe} \leq 4(10 \times 2^{37})^h$. We consider a few other values of l. If $l = 10$, then $\epsilon_{hibe} \leq 4(10 \times 2^{48})^h\epsilon_{dbdh}$ and if $l = 32$, then $\epsilon_{hibe} \leq 2^{42h+2}\epsilon_{dbdh}$.

In Table 2, we compare the known HIBE protocols which are secure in the full model. We note that HIBE protocols which are secure in the selective-ID model are also secure in the full model with a security degradation of $\approx q^{nh}$, where h is the number of levels in the HIBE and n is number of bits in the identity. This degradation is far worse than the protocols in Table 2. The parameter j in the private

Table 2. Comparison of HIBE Protocols. In the table, q_H is the number of random oracle hash queries made by an adversary. Usually, q_H is taken to be at least 2^{64}.

Protocol	Hardness Assumption	Random Oracle	Security Degradtion	Pub. Para. size (elts. of G_1)	Pvt. Key size (elts. of G_1)	Ciphertext size (elts. of G_1)	Pairing	
							Enc.	Dec.
GS [12]	BDH	Yes	$q_H q^h$	2	j	j	1	j
BB [2]	DBDH	Yes	q_H^h	$h + 3$	$j + 1$	$j + 1$	None	$j + 1$
Waters [20]	DBDH	No	$(32nq)^h$	$(n + 1)h + 3$	$j + 1$	$j + 1$	None	$j + 1$
Our	DBDH	No	$4(2l2^{n/l}\sigma)^h$	$h + l + 3$	$j + 1$	$j + 1$	None	$j + 1$

key size, ciphertext size and the encryption and decryption columns of Table 2 represents the number of levels of the identity on which the operations are performed. The parameter h is the maximum number of levels in the HIBE. Recall that $1 \leq l \leq n$ and $\sigma = \max(2q, 2^{n/l})$. For $l = n$, the construction in this paper requires $(h + n + 3)$ many elements of G_1 as public parameters whereas Waters' construction requires $(n + 1)h + 3$ many elements. The security degradation remains the same in both cases. For $l < n$, the new construction extends the IBE protocol of Section 3.1.

7 Proof of Theorem 2

The security reduction follows along standard lines. We need to lower bound the probability of the simulator aborting on certain queries and in the challenge stage. The details of obtaining this lower

bound is given in Section 7.1. In the following proof, we simply use the lower bound. We want to show that the HIBE is (ϵ_{hibe}, t, q)-CPA secure. In the game sequence style of proofs, we start with the adversarial game defining the CPA-security of the protocol against an adversary $\mathcal{A}$ and then obtain a sequence of games as usual. In each of the games, the simulator chooses a bit γ and the adversary makes a guess γ'. By X_i we will denote the event that the bit γ is equal to the bit γ' in the ith game.

Game 0: This is the usual adversarial game used in defining CPA-secure HIBE. We assume that the adversary's runtime is t and it makes q key extraction queries. Also, we assume that the adversary maximizes the advantage among all adversaries with similar resources. Thus, we have $\epsilon_{hibe} = \left|\mathsf{Pr}[X_0] - \frac{1}{2}\right|$.

Game 1: In this game, we setup the protocol from a tuple $\langle P, P_1 = aP, P_2 = bP, P_3 = cP, Z = e(P_1, P_2)^{abc}\rangle$ and answer key extraction queries and generate the challenge. The simulator is assumed to know the values a, b and c. However, the simulator can setup the protocol as well as answer certain private key queries without the knowledge of these values. Also, for certain challenge identities it can generate the challenge ciphertext without the knowledge of a, b and c. In the following, we show how this can be done. If the simulator cannot answer a key extraction query or generate a challenge without using the knowledge of a, b and c, it sets a flag flg to one. The value of flg is initially set to zero.

Note that the simulator is always able to answer the adversary (with or without using a, b and c). The adversary is provided with proper replies to all its queries and is also provided the proper challenge ciphertext. Thus, irrespective of whether flg is set to one, the adversary's view in Game 1 is same as that in Game 0. Hence, we have $\mathsf{Pr}[X_0] = \mathsf{Pr}[X_1]$.

We next show how to setup the protocol and answer the queries based on the tuple $\langle P, P_1 = aP, P_2 = bP, P_3 = cP, Z = e(P_1, P_2)^{abc}\rangle$.

Set-Up: Recall that $\sigma = \max(2q, 2^{n/l})$. Let m be a prime such that $\sigma < m < 2\sigma$. Our choice of m is different from that of previous works [20, 17] where m was chosen to be equal to $4q$ and $2q$.

Choose $x'_1, \ldots, x'_h$ and $x_1, \ldots, x_l$ randomly from $\mathbb{Z}_m$; $y'_1, \ldots, y'_h$ and $y_1, \ldots, y_l$ randomly from $\mathbb{Z}_p$. Choose $k_1, \ldots, k_h$ randomly from $\{0, \ldots, \mu_l\}$.

For $1 \leq j \leq h$, define $U'_j = (p - mk_j + x'_j)P_2 + y'_j P$ and for $1 \leq i \leq l$ define $U_i = x_i P_2 + y_i P$. Set the public parameters of HIBE to be $(P, P_1, P_2, U'_1, \ldots, U'_h, U_1, \ldots, U_l)$. The master secret is $aP_2 = abP$. The distribution of the public parameters is as expected by $\mathcal{A}$. In its attack, $\mathcal{A}$ will make some queries, which have to be properly answered by the simulator.

For $1 \leq j \leq h$, we define several functions. Let $v = (v_1, \ldots, v_l)$ where each v_i is an n/l-bit string considered to be an integer from the set $\{0, \ldots, 2^{n/l} - 1\}$. We define

$$\left.\begin{array}{l} F_j(v) = p - mk_j + x'_j + \sum_{i=1}^{l} x_i v_i \\ J_j(v) = y'_j + \sum_{i=1}^{l} y_i v_i \\ L_j(v) = x'_j + \sum_{i=1}^{l} x_i v_i \pmod m \\ K_j(v) = \begin{cases} 0 \text{ if } L_j(v) = 0 \\ 1 \text{ otherwise.} \end{cases} \end{array}\right\} \tag{5}$$

Recall that we have assumed $2\sigma(1 + \mu_l) < p$. Let $F_{\min}$ and $F_{\max}$ be the minimum and maximum values of $F_j(v)$. $F_{\min}$ is achieved when k_j is maximum and x'_j and the x_i's are all zero. Thus, $F_{\min} = p - m\mu_l$. We have $m\mu_l < 2\sigma(1 + \mu_l)$ and by assumption $2\sigma(1 + \mu_l) < p$. Hence, $F_{\min} > 0$. Again $F_{\max}$ is achieved when $k_j = 0$ and x'_j and the x_i's and v_i's are equal to their respective maximum values. We get $F_{\max} < p + m(1 + l(2^{n/l} - 1)) = p + m(1 + \mu_l) < p + 2\sigma(1 + \mu_l) < 2p$. Thus, we have $0 < F_{\min} \leq F_j(v) \leq F_{\max} < 2p$. Consequently, $F_j(v) \equiv 0 \bmod p$ if and only if $F_j(v) = p$ which holds if and only if $-mk_j + x'_j + \sum_{i=1}^{l} x_i v_i = 0$.

Now we describe how the queries made by $\mathcal{A}$ are answered. The queries can be made in both Phases 1 and 2 of the adversarial game (subject to the usual restrictions). The manner in which they are answered by the simulator is the same in both the phases.

Key Extraction Query: Suppose $\mathcal{A}$ makes a key extraction query on the identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$. Suppose there is a u with $1 \leq u \leq j$ such that $K_u(\mathsf{v}_u) = 1$. Otherwise set flg to one. In the second case, the simulator uses the value of a to return the proper decryption key $d_{\mathsf{v}} = (aP_2 + \sum_{i=1}^{j} r_i V_i, r_1 V_1, \ldots, r_j V_j)$. In the first case, the simulator constructs a decryption key in the following manner.

Choose random $r_1, \ldots, r_j$ from $\mathbb{Z}_p$ and define

$$\left.\begin{array}{l} d_{0|u} = -\frac{J_u(\mathsf{v}_u)}{F_u(\mathsf{v}_u)} P_1 + r_u(F_u(\mathsf{v}_u)P_2 + J_u(\mathsf{v}_u)P) \\ d_u = \frac{-1}{F_u(\mathsf{v}_u)} P_1 + r_u P \\ d_k = r_k P \text{ for } k \neq u \\ d_{\mathsf{v}} = (d_{0|u} + \sum_{k \in \{1,\ldots,j\} \setminus \{u\}} r_k V_k, d_1, \ldots, d_j) \end{array}\right\} \quad (6)$$

The quantity d_{v} is a proper private key corresponding to the identity v. The algebraic verification of this fact is similar to that in [2, 20]. This is provided to $\mathcal{A}$.

Challenge: Let the challenge identity be $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{h^*}^*)$, $1 \leq h^* \leq h$ and the (equal length) messages be M_0 and M_1. Choose a random bit b. We need to have $F_k(\mathsf{v}_k^*) \equiv 0 \bmod p$ for all $1 \leq k \leq h^*$. If this condition does not hold, then set flg to one. In the second case, the simulator uses the value of c to provide a proper encryption of M_γ to $\mathcal{A}$ by computing $(M_\gamma \times e(P_1, P_2)^c, cP, cV_1, \ldots, cV_{h^*})$. In the first case, it constructs a proper encryption of M_γ in the following manner.

$$(M_\gamma \times Z, C_1 = P_3, B_1 = J_1(\mathsf{v}_1^*)P_3, \ldots, B_{h^*} = J_{h^*}(\mathsf{v}_{h^*}^*)P_3).$$

We require B_j to be equal to $cV_j(\mathsf{v}_j^*)$ for $1 \leq j \leq h^*$. Recall that the definition of $V_j(v)$ is $V_j(v) = U_j' + \sum_{k=1}^{l} v_k U_k$. Using the definition of U_j' and the U_k's as defined in the setup by the simulator, we obtain, $cV_i = c(F_i(\mathsf{v}_i^*)P_2 + J_i(\mathsf{v}_i^*)P) = J_i(\mathsf{v}_i^*)cP = J_i(\mathsf{v}_i^*)P_3$. Here we use the fact, $F_i(\mathsf{v}_i^*) \equiv 0 \bmod p$. Hence, the quantities $B_1, \ldots, B_{h^*}$ are properly formed.

Guess: The adversary outputs a guess γ' of γ.

Game 2: This is a modification of Game 1 whereby the Z in Game 1 is now chosen to be a random element of G_2. This Z is used to mask the message M_γ in the challenge ciphertext. Since Z is random, the first component of the challenge ciphertext is a random element of G_2 and provides no information to the adversary about γ. Thus, $\mathsf{Pr}[X_2] = \frac{1}{2}$.

Let λ be a lower bound on the probability that flg remains zero throughout Games 1 and 2. The expression for λ is obtained in Proposition 3 of Section 7.1. We have the following claim.

Claim:

$$|\mathsf{Pr}[X_1] - \mathsf{Pr}[X_2]| \leq \frac{\epsilon_{dbdh}}{\lambda} + \frac{\epsilon_{hibe}}{2}.$$

Proof (Of Claim). The change from Game 1 to Game 2 corresponds to an "indistinguishability" step in Shoup's tutorial [19] on such games. Usually, it is easy to bound the probability difference. In this case, the situation is complicated by the fact that there is a need to abort and the fact that the probability of aborting depends on the adversary's queries.

We show that it is possible to obtain an algorithm $\mathcal{B}$ for DBDH by extending Games 1 and 2. The extension of both the games is same and is described as follows. $\mathcal{B}$ takes as input a tuple (P, aP, bP, cP, Z)

and sets up the HIBE protocol as in Game 1 (The setup of Games 1 and 2 are the same). The key extraction queries are answered and the challenge ciphertext is generated as in Game 1. If flg is set to one, then $\mathcal{B}$ outputs a random bit and aborts. This is because the query cannot be answered or the challenge ciphertext cannot be generated using the input tuple. At the end of the game, the adversary outputs the guess γ'. Note that if Z is real, then the adversary is playing Game 1 and if Z is random, then the adversary is playing Game 2. $\mathcal{B}$ now goes through a separate abort stage as follows.

"Artificial Abort": (This technique was introduced by Waters [20]). The probability that $\mathcal{B}$ aborts in the query or challenge phases depends on the adversary's input. The goal of the artificial abort step is to make the probability of abort "independent" of the adversary's queries by ensuring that in all cases its probability of abort is the maximum possible. This is done by sampling the transcript of adversary's query and in certain cases aborting. The sampling procedure introduces the extra component $O(\epsilon_{hibe}^{-2} \ln(\epsilon_{hibe}^{-1})\lambda^{-1} \ln(\lambda^{-1}))$ into the simulator's runtime. A detailed exposition of this technique and its relevance in the current context is given in Section A.

Output: If $\mathcal{B}$ has not aborted up to this stage, then it outputs 1 if $\gamma = \gamma'$; else 0.

The time taken by the simulator in either Game 1 or 2 is clearly $t + \chi(\epsilon_{hibe})$. Let Y_i be the event that the simulator outputs 1 in Game i, $i = 1, 2$. Then, we have

$$|\mathsf{Pr}[Y_1] - \mathsf{Pr}[Y_2]| \leq \epsilon_{dbdh}.$$

Let ab_i be the event that the simulator aborts in Game i, $i = 1, 2$. This includes both protocol and artificial abort.

$$\begin{aligned}
\mathsf{Pr}[Y_i] &= \mathsf{Pr}[Y_i \wedge (\mathsf{ab}_i \vee \overline{\mathsf{ab}_i})] \\
&= \mathsf{Pr}[(Y_i \wedge \mathsf{ab}_i) \vee (Y_i \wedge \overline{\mathsf{ab}_i})] \\
&= \mathsf{Pr}[Y_i \wedge \mathsf{ab}_i] + \mathsf{Pr}[Y_i \wedge \overline{\mathsf{ab}_i}] \\
&= \mathsf{Pr}[Y_i \mid \mathsf{ab}_i]\mathsf{Pr}[\mathsf{ab}_i] + \mathsf{Pr}[Y_i \mid \overline{\mathsf{ab}_i}]\mathsf{Pr}[\overline{\mathsf{ab}_i}] \\
&= \frac{1}{2}(1 - \mathsf{Pr}[\overline{\mathsf{ab}_i}]) + \mathsf{Pr}[X_i \mid \overline{\mathsf{ab}_i}]\mathsf{Pr}[\overline{\mathsf{ab}_i}] \qquad (7) \\
&= \frac{1}{2}(1 - \mathsf{Pr}[\overline{\mathsf{ab}_i} \wedge (X_i \vee \overline{X_i})]) + \mathsf{Pr}[X_i \wedge \overline{\mathsf{ab}_i}] \\
&= \frac{1}{2} + \frac{1}{2}\left(\mathsf{Pr}[\overline{\mathsf{ab}_i}|X_i]\mathsf{Pr}[X_i] - \mathsf{Pr}[\overline{\mathsf{ab}_i}|\overline{X_i}]\mathsf{Pr}[\overline{X_i}]\right) \qquad (8)
\end{aligned}$$

To proceed further, we need bounds on $\mathsf{Pr}[\overline{\mathsf{ab}_i}|X_i]$ and $\mathsf{Pr}[\overline{\mathsf{ab}_i}|\overline{X_i}]$. Recall that X_i is the event $\gamma = \gamma'$ in Game i. From (30) (proved later), we obtain

$$\lambda - \frac{\lambda\epsilon}{2} \leq \mathsf{Pr}[\overline{\mathsf{ab}_i}|X_i], \mathsf{Pr}[\overline{\mathsf{ab}_i}|\overline{X_i}] \leq \lambda + \frac{\lambda\epsilon}{2}. \qquad (9)$$

Here $\epsilon = \epsilon_{hibe}$. Now we need to do some manipulations with inequalities and for convenience we set $A_i = \mathsf{Pr}[\overline{\mathsf{ab}_i}|X_i]$, $B_i = \mathsf{Pr}[X_i]$ and $C_i = \mathsf{Pr}[\overline{\mathsf{ab}_i}|\overline{X_i}]$ and $D = \mathsf{Pr}[Y_1] - \mathsf{Pr}[Y_2]$. We have from (9)

$$\lambda - \frac{\lambda\epsilon}{2} \leq A_i, C_i \leq \lambda + \frac{\lambda\epsilon}{2}.$$

Also, using (8)

$$2D = (A_1B_1 - C_1(1 - B_1)) - (A_2B_2 - C_2(1 - B_2)). \qquad (10)$$

Since both B_1 and $(1 - B_1)$ are non-negative, we have

$$\begin{array}{rcl} B_i(\lambda - \frac{\lambda\epsilon}{2}) \leq & A_iB_i & \leq B_i(\lambda + \frac{\lambda\epsilon}{2}) \\ (1 - B_i)(-\lambda - \frac{\lambda\epsilon}{2}) \leq & -C_i(1 - B_i) & \leq (1 - B_i)(-\lambda + \frac{\lambda\epsilon}{2}). \end{array}$$

Hence,

$$\lambda(2B_i - 1) - \frac{\lambda\epsilon}{2} \leq A_iB_i - C_i(1 - B_i) \leq \lambda(2B_i - 1) + \frac{\lambda\epsilon}{2}. \tag{11}$$

Putting $i = 1$ in (11), we obtain

$$\lambda(2B_1 - 1) - \frac{\lambda\epsilon}{2} \leq A_1B_1 - C_1(1 - B_1) \leq \lambda(2B_1 - 1) + \frac{\lambda\epsilon}{2}. \tag{12}$$

Multiplying (11) by -1 and putting $i = 2$ we obtain

$$-\lambda(2B_2 - 1) - \frac{\lambda\epsilon}{2} \leq -(A_2B_2 - C_2(1 - B_2)) \leq -\lambda(2B_2 - 1) + \frac{\lambda\epsilon}{2}. \tag{13}$$

Combining (10), (12) and (13) we get

$$2\lambda(B_1 - B_2) - \lambda\epsilon \leq 2D \leq 2\lambda(B_1 - B_2) + \lambda\epsilon. \tag{14}$$

This shows that $|\lambda(B_1 - B_2) - D| \leq \frac{\lambda\epsilon}{2}$. Now $|\lambda(B_1 - B_2)| - |D| \leq |\lambda(B_1 - B_2) - D| \leq \frac{\lambda\epsilon}{2}$. Note that $|D| = |\mathsf{Pr}[Y_1] - \mathsf{Pr}[Y_2]| \leq \epsilon_{dbdh}$ and recalling the values of B_1 and B_2, we have

$$|\mathsf{Pr}[X_1] - \mathsf{Pr}[X_2]| \leq \frac{\epsilon_{dbdh}}{\lambda} + \frac{\epsilon_{hibe}}{2}. \tag{15}$$

This completes the proof of the claim. □

Now we can complete the proof in the following manner.

$$\begin{aligned} \epsilon_{hibe} &= \left|\mathsf{Pr}[X_0] - \frac{1}{2}\right| \\ &\leq |\mathsf{Pr}[X_0] - \mathsf{Pr}[X_2]| \\ &\leq |\mathsf{Pr}[X_0] - \mathsf{Pr}[X_1]| + |\mathsf{Pr}[X_1] - \mathsf{Pr}[X_2]| \\ &\leq \frac{\epsilon_{hibe}}{2} + \frac{\epsilon_{dbdh}}{\lambda}. \end{aligned}$$

Rearranging the inequality gives the desired result. This completes the proof of Theorem 2. □

7.1 Lower Bound on Not Abort

We require the following two independence results in obtaining the required lower bound. Similar independence results have been used in [20, 17] in connection with IBE protocols. The situation for HIBE is more complicated than IBE and especially so since we reuse some of the public parameters over different levels of the HIBE. This makes the proofs more difficult. Our independence results are given in Proposition 1 and 2 and these subsume the results of previous work. We provide complete proofs for these two propositions as well as a complete proof for the lower bound. The probability calculation for the lower bound is also more complicated compared to the IBE case.

Proposition 1. *Let m be a prime and $L(\cdot)$ be as defined in (5). Let $\mathsf{v}_1, \ldots, \mathsf{v}_j$ be identities, i.e., each $\mathsf{v}_i = (\mathsf{v}_{i,1}, \ldots, \mathsf{v}_{i,l})$, with $\mathsf{v}_{i,k}$ to be an n/l-bit string (and hence $0 \leq \mathsf{v}_{i,k} \leq 2^{n/l} - 1$). Then*

$$\mathsf{Pr}\left[\bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)\right] = \frac{1}{m^j}.$$

The probability is over the independent and uniform random choices of $x'_1, \ldots, x'_j, x_1, \ldots, x_l$ from $\mathbb{Z}_m$. Consequently, for any $\theta \in \{1, \ldots, j\}$, we have

$$\mathsf{Pr}\left[L_\theta(\mathsf{v}_\theta) = 0 \,\middle|\, \bigwedge_{k=1, k\neq\theta}^{j} (L_k(\mathsf{v}_k) = 0)\right] = \frac{1}{m}.$$

Proof: Since $\mathbb{Z}_m$ forms a field, we can do linear algebra with vector spaces over $\mathbb{Z}_m$. The condition $\bigwedge_{k=1}^{j} (L_j(\mathsf{v}_j) = 0)$ is equivalent to the following system of equations over $\mathbb{Z}_m$.

$$\begin{array}{c} x'_1 + \mathsf{v}_{1,1}x_1 + \cdots + \mathsf{v}_{1,l}x_l = 0 \\ x'_2 + \mathsf{v}_{2,1}x_1 + \cdots + \mathsf{v}_{2,l}x_l = 0 \\ \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \\ x'_j + \mathsf{v}_{j,1}x_1 + \cdots + \mathsf{v}_{j,l}x_l = 0 \end{array}$$

This can be rewritten as

$$(x'_1, \ldots, x'_j, x_1, \ldots, x_l) A_{(j+l)\times(j+l)} = (0, \ldots, 0)_{1\times(j+l)}$$

where

$$A = \begin{bmatrix} I_j & O_{j\times l} \\ \mathsf{V}_{l\times j} & O_{l\times l} \end{bmatrix} \text{ and } \mathsf{V}_{l\times j} = \begin{bmatrix} \mathsf{v}_{1,1} & \cdots & \mathsf{v}_{j,1} \\ \cdots & \cdots & \cdots \\ \mathsf{v}_{1,l} & \cdots & \mathsf{v}_{j,l} \end{bmatrix};$$

I_j is the identity matrix of order j; O is the all zero matrix of the specified order. The rank of A is clearly j and hence the dimension of the solution space is l. Hence, there are m^l solutions in $(x'_1, \ldots, x'_j, x_1, \ldots, x_l)$ to the above system of linear equations. Since the variables $x'_1, \ldots, x'_j, x_1, \ldots, x_l$ are chosen independently and uniformly at random, the probability that the system of linear equations is satisfied for a particular choice of these variables is $m^l/m^{l+j} = 1/m^j$. This proves the first part of the result.

For the second part, note that we may assume $\theta = j$ by renaming the x''s if required. Then

$$\mathsf{Pr}\left[L_j(\mathsf{v}_j) = 0 \,\middle|\, \bigwedge_{k=1}^{j-1} (L_k(\mathsf{v}_k) = 0)\right] = \frac{\mathsf{Pr}\left[\bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)\right]}{\mathsf{Pr}\left[\bigwedge_{k=1}^{j-1} (L_k(\mathsf{v}_k) = 0)\right]} = \frac{m^{j-1}}{m^j} = \frac{1}{m}.$$

□

Proposition 2. *Let m be a prime and $L(\cdot)$ be as defined in (5). Let $\mathsf{v}_1, \ldots, \mathsf{v}_j$ be identities, i.e., each $\mathsf{v}_i = (\mathsf{v}_{i,1}, \ldots, \mathsf{v}_{i,l})$, with $\mathsf{v}_{i,k}$ to be an n/l-bit string. Let $\theta \in \{1, \ldots, j\}$ and let v'_θ be an identity such that $\mathsf{v}'_\theta \neq \mathsf{v}_\theta$. Then*

$$\mathsf{Pr}\left[(L_\theta(\mathsf{v}'_\theta) = 0) \wedge \bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)\right] = \frac{1}{m^{j+1}}.$$

The probability is over the independent and uniform random choices of $x'_1, \ldots, x'_j, x_1, \ldots, x_l$ from $\mathbb{Z}_m$. Consequently, we have

$$\mathsf{Pr}\left[L_\theta(\mathsf{v}'_\theta) = 0 \,\middle|\, \bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)\right] = \frac{1}{m}.$$

Proof: The proof is similar to the proof of Proposition 1. Without loss of generality, we may assume that $\theta = j$, since otherwise we may rename variables to achieve this. The condition $(L_\theta(\mathsf{v}'_\theta) = 0) \wedge \bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)$ is equivalent to a system of linear equations $xA = 0$ over $\mathbb{Z}_m$. In this case, the form of A is the following.

$$A = \begin{bmatrix} I_j & c^T & O_{j\times l} \\ V_{l\times j} & (\mathsf{v}'_j)^T & O_{l\times l} \end{bmatrix}$$

where $c = (0, \ldots, 0, 1)$; c^T denotes the transpose of c and $(\mathsf{v}'_j)^T$ is the transpose of v'_j. The first j columns of A are linearly independent. The $(j+1)$th column of A is clearly linearly independent of the first $(j-1)$ columns. We have $\mathsf{v}_j \neq \mathsf{v}'_j$. Since each component of both v_j and v'_j is less than $2^{n/l}$ and $m > 2^{n/l}$, we have $\mathsf{v}_j \not\equiv \mathsf{v}'_j \bmod m$. Using this, it is not difficult to see that the first $(j+1)$ columns of A are linearly independent and hence the rank of A is $(j+1)$. (Note that if $m \leq 2^{n/l}$, then it is possible to have $\mathsf{v}_j \neq \mathsf{v}'_j$ but $\mathsf{v}_j \equiv \mathsf{v}'_j \bmod m$. Then the jth and $(j+1)$th columns of A are equal and the rank of A is j.) Consequently, the dimension of the solution space is $l-1$ and there are m^{l-1} solutions in $(x'_1, \ldots, x'_j, x_1, \ldots, x_l)$ to the system of linear equations. Since the x''s and the x's are chosen independently and uniformly at random from $\mathbb{Z}_m$, the probability of getting a solution is $m^{l-1}/m^{l+j} = 1/m^{j+1}$. This proves the first part of the result. The proof of the second part is similar to that of Proposition 1. □

Proposition 3. *The probability that the simulator in the proof of Theorem 2 does not abort before the artificial abort stage is at least* $\frac{1}{2(2\sigma(\mu_l+1))^h}$.

Proof: We consider the simulator in the proof of Theorem 2. Up to the artificial abort stage, the simulator could abort on either a key extraction query or in the challenge stage. Let abort be the event that the simulator aborts before the artificial abort stage. For $1 \leq i \leq q$, let E_i denote the event that the simulator does not abort on the ith key extraction query and let C be the event that the simulator does not abort in the challenge stage. We have

$$\begin{aligned}
\mathsf{Pr}[\overline{\mathsf{abort}}] &= \mathsf{Pr}\left[\left(\bigwedge_{i=1}^{q} E_i\right) \wedge C\right] \\
&= \mathsf{Pr}\left[\left(\bigwedge_{i=1}^{q} E_i\right) | C\right] \mathsf{Pr}[C] \\
&= \left(1 - \mathsf{Pr}\left[\left(\bigvee_{i=1}^{q} \neg E_i\right) | C\right]\right) \mathsf{Pr}[C] \\
&\geq \left(1 - \sum_{i=1}^{q} \mathsf{Pr}\left[\neg E_i \,|C\right]\right) \mathsf{Pr}[C].
\end{aligned}$$

We first consider the event C. Suppose the challenge identity is $\mathsf{v}^* = (\mathsf{v}^*_1, \ldots, \mathsf{v}^*_{h^*})$. Event C holds if and only if $F_j(\mathsf{v}^*_j) \equiv 0 \bmod p$ for $1 \leq j \leq h^*$. Recall that by choice of p, we can assume $F_j(\mathsf{v}^*_j) \equiv 0 \bmod p$ if and only if $x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} = mk_j$. Hence,

$$\mathsf{Pr}[C] = \mathsf{Pr}\left[\bigwedge_{j=1}^{h^*} \left(x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} = mk_j\right)\right]. \tag{16}$$

For $1 \leq j \leq h^*$ and $0 \leq i \leq \mu_l$, denote the event $x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} = mi$ by $A_{j,i}$ and the event $k_j = i$ by $B_{j,i}$. Also, let $C_{j,i}$ be the event $A_{j,i} \wedge B_{j,i}$.

Note that the event $\bigvee_{i=0}^{\mu_l} A_{j,i}$ is equivalent to the condition $x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} \equiv 0 \bmod m$ and hence equivalent to the condition $L_j(\mathsf{v}_j) = 0$. Since k_j is chosen uniformly at random from the set $\{0, \ldots, \mu_l\}$, we have $\mathsf{Pr}[B_{j,i}] = 1/(1+\mu_l)$ for all j and i. The events $B_{j,i}$'s are independent of each other and also independent of the $A_{j,i}$'s. We have

$$
\begin{aligned}
\mathsf{Pr}\left[\bigwedge_{j=1}^{h^*}\left(x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} = mk_j\right)\right] &= \mathsf{Pr}\left[\bigwedge_{j=1}^{h^*}\left(\bigvee_{i=0}^{\mu_l} C_{j,i}\right)\right] \\
&= \mathsf{Pr}\left[\bigvee_{i_1,\ldots,i_{h^*}\in\{0,\ldots,\mu_l\}} (C_{1,i_1} \wedge \cdots \wedge C_{h^*,i_{h^*}})\right] \\
&= \mathsf{Pr}\left[\bigvee_{i_1,\ldots,i_{h^*}\in\{0,\ldots,\mu_l\}} (A_{1,i_1} \wedge B_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}} \wedge B_{h^*,i_{h^*}})\right] \\
&= \sum_{i_1,\ldots,i_{h^*}\in\{0,\ldots,\mu_l\}} \mathsf{Pr}\left[A_{1,i_1} \wedge B_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}} \wedge B_{h^*,i_{h^*}}\right] \\
&= \sum_{i_1,\ldots,i_{h^*}\in\{0,\ldots,\mu_l\}} \mathsf{Pr}\left[A_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}}\right] \times \mathsf{Pr}\left[B_{1,i_1} \wedge \cdots \wedge B_{h^*,i_{h^*}}\right] \\
&= \frac{1}{(1+\mu_l)^{h^*}} \sum_{i_1,\ldots,i_{h^*}\in\{0,\ldots,\mu_l\}} \mathsf{Pr}\left[A_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}}\right] \\
&= \frac{1}{(1+\mu_l)^{h^*}} \mathsf{Pr}\left[\bigvee_{i_1,\ldots,i_{h^*}\in\{0,\ldots,\mu_l\}} (A_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}})\right] \\
&= \frac{1}{(1+\mu_l)^{h^*}} \mathsf{Pr}\left[\bigwedge_{j=1}^{h^*}\left(\bigvee_{i=0}^{\mu_l} A_{j,i}\right)\right] \\
&= \frac{1}{(1+\mu_l)^{h^*}} \mathsf{Pr}\left[\bigwedge_{j=1}^{h^*} (L_j(\mathsf{v}_j) = 0)\right] \\
&= \frac{1}{(m(1+\mu_l))^{h^*}}
\end{aligned}
$$

The last equality follows from Proposition 1.

Now we turn to bounding $\mathsf{Pr}[\neg E_i | C]$. For simplicity of notation, we will drop the subscript i from E_i and consider the event E that the simulator does not abort on a particular key extraction query on an identity $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$. By the simulation, the event $\neg E$ implies that $L_i(\mathsf{v}_i) = 0$ for all $1 \le i \le j$. This holds even when the event is conditioned under C. Thus, we have $\mathsf{Pr}[\neg E|C] \le \mathsf{Pr}[\wedge_{i=1}^{j} L_i(\mathsf{v}_i) = 0 | C]$. The number of components in the challenge identity is h^* and now two cases can happen:
$j \le h^*$: By the protocol constraint (a prefix of the challenge identity cannot be queried to the key extraction oracle), we must have a θ with $1 \le \theta \le j$ such that $\mathsf{v}_\theta \neq \mathsf{v}^*_\theta$.
$j > h^*$: In this case, we choose $\theta = h^* + 1$.

Now we have

$$
\mathsf{Pr}[\neg E|C] \le \mathsf{Pr}\left[\bigwedge_{i=1}^{j} L_i(\mathsf{v}_i) = 0 | C\right] \le \mathsf{Pr}[L_\theta(\mathsf{v}_\theta) = 0 | C] = \mathsf{Pr}\left[L_\theta(\mathsf{v}_\theta) = 0 | \bigwedge_{i=1}^{h^*} L_i(\mathsf{v}^*_i) = 0\right] = 1/m.
$$

The last equality follows from an application of either Proposition 1 or Proposition 2 according as whether $j > h^*$ or $j \leq h^*$. Substituting this in the bound for $\mathsf{Pr}[\overline{\mathsf{abort}}]$ we obtain

$$\begin{aligned}
\mathsf{Pr}[\overline{\mathsf{abort}}] &\geq \left(1 - \sum_{i=1}^{q} \mathsf{Pr}\left[\neg E_i \,|C\right]\right) \mathsf{Pr}[C]. \\
&\geq \left(1 - \frac{q}{m}\right) \frac{1}{(m(\mu_l + 1))^{h^*}} \\
&\geq \left(1 - \frac{q}{m}\right) \frac{1}{(m(\mu_l + 1))^{h}} \\
&\geq \frac{1}{2} \times \frac{1}{(2\sigma(\mu_l + 1))^{h}}.
\end{aligned}$$

We use $h \geq h^*$ and $2q \leq \sigma < m < 2\sigma$ to obtain the inequalities. This completes the proof. □

8 Conclusion

Waters presented a construction of IBE [20] which significantly improves upon the previous construction of Boneh-Boyen [3]. In his paper, Waters also suggested a method to extend his IBE to a HIBE. We first present a construction of an IBE protocol with shorter public parameters. This leads to a security degradation which is converted into a space/time trade-off. Our second construction is that of a HIBE which significantly reduces the number of public parameters in Waters' construction. All known HIBE protocols have a security degradation which is exponential in the number of levels. The main open problem in the construction of HIBE protocols is to avoid or control this security degradation.

Acknowledgement: We would like to thank Rana Barua for carefully reading the paper. Mridul Nandi pointed out that the independence of X and Y is required in Lemma 2.

A An Exposition of the Artificial Abort Technique

The purpose of this section is to present a detailed description of the technique of artificial abort. This technique is new to security proofs and was introduced by Waters [20]. Since the technique is important a good exposition of the technique will be important. The original paper by Waters provides only a sketch. The work by Naccache [17] provides more details, but is not complete. Our description is based on that of Waters and Naccache, but we work out all the details. We feel that this will be useful to researchers, especially to those who are new to this area.

Actually the entire technique of artificial abort can be explained in terms of elementary probability theory without reference to security proofs. We have taken this approach. After outlining the necessary abstraction, we obtain the relevant bounds. Then, we go back to the security proof and explain how the technique fits within that scenario.

One of the basic questions that arises is why this technique is required, i.e., "Why artificial abort?" This question is addressed by Waters. Here we try to identify the point in the probability analysis where this technique needs to be used. The question of whether a different probability analysis can eliminate the requirement of the artificial abort technique remains open.

A.1 Preliminaries

In this section, we state a few simple results on conditional probabilities and discuss aspects of the Chernoff bound which are relevant to the artificial abort technique. Our discussion of Chernoff bound is from [16].

Lemma 1. *Let X and Y be two random variables. Then*

$$\mathsf{Pr}[X|Y] - \mathsf{Pr}[\overline{Y}] \leq \mathsf{Pr}[X] \leq \mathsf{Pr}[X|Y] + \mathsf{Pr}[\overline{Y}].$$

Proof: We start in the usual manner.

$$\begin{aligned}\mathsf{Pr}[X] &= \mathsf{Pr}[X \wedge (Y \vee \overline{Y})] \\ &= \mathsf{Pr}[X|Y]\mathsf{Pr}[Y] + \mathsf{Pr}[X|\overline{Y}]\mathsf{Pr}[\overline{Y}].\end{aligned}$$

To get the upper bound, note that $\mathsf{Pr}[X|Y]\mathsf{Pr}[Y] \leq \mathsf{Pr}[X|Y]$ and $\mathsf{Pr}[X|\overline{Y}]\mathsf{Pr}[\overline{Y}] \leq \mathsf{Pr}[\overline{Y}]$. For the lower bound, we use

$$\begin{aligned}\mathsf{Pr}[X] &\geq \mathsf{Pr}[X|Y]\mathsf{Pr}[Y] \\ &= \mathsf{Pr}[X|Y](1 - \mathsf{Pr}[\overline{Y}]) \\ &= \mathsf{Pr}[X|Y] - \mathsf{Pr}[X|Y]\mathsf{Pr}[\overline{Y}].\end{aligned}$$

The lower bound now follows by noting that $\mathsf{Pr}[X|Y]\mathsf{Pr}[\overline{Y}] \leq \mathsf{Pr}[\overline{Y}]$ as before. □

Lemma 2. *Let X and Y be independent random variables which vary over the finite sets Γ and Σ respectively. Suppose, that f is a function $f : \Gamma \times \Sigma \to \{0, 1\}$ and for all $y \in \Sigma$, we have*

$$\lambda^- \leq \mathsf{Pr}[f(X, y) = 1] \leq \lambda^+ \tag{17}$$

for some constants λ^- and λ^+. Then for any subset Σ_0 of Σ, we have

$$\lambda^- \leq \mathsf{Pr}[f(X, Y) = 1 | Y \in \Sigma_0] \leq \lambda^+. \tag{18}$$

Proof: Let $\Sigma_0 = \{y_1, \ldots, y_t\}$. Then $\mathsf{Pr}[Y \in \Sigma_0] = \mathsf{Pr}[Y = y_1] + \cdots + \mathsf{Pr}[Y = y_t]$. Note that since X and Y are independent, $\mathsf{Pr}[f(X, y) = 1]$ is the same as $\mathsf{Pr}[f(X, Y) = 1 | Y = y]$. We can write

$$\begin{aligned}\mathsf{Pr}[f(X,Y) = 1 | Y \in \Sigma_0] &= \frac{\mathsf{Pr}[f(X,Y) = 1 \wedge Y \in \Sigma_0]}{\mathsf{Pr}[Y \in \Sigma_0]} \\ &= \frac{\mathsf{Pr}[f(X,Y) = 1 \wedge Y = y_1] + \cdots + \mathsf{Pr}[f(X,Y) = 1 \wedge Y = y_t]}{\mathsf{Pr}[Y \in \Sigma_0]}.\end{aligned}$$

The last quantity can be written as

$$\frac{\mathsf{Pr}[f(X,Y) = 1 | Y = y_1]\mathsf{Pr}[Y = y_1] + \cdots + \mathsf{Pr}[f(X,Y) = 1 | Y = y_t]\mathsf{Pr}[Y = y_t]}{\mathsf{Pr}[Y \in \Sigma_0]}.$$

For each i, we are given $\lambda^- \leq \mathsf{Pr}[f(X, Y) = 1 | Y = y_i] \leq \lambda^+$. The required bound follows on substituting this bound in the last expression. □

Theorem 3 (Chernoff Bound). *Let $X_1, \ldots, X_k$ be independent Bernoulli trials with $p_i = \mathsf{Pr}[X_i = 1]$. Let $X = \sum_{i=1}^{k} X_i$ and $\mu = \sum_{i=1}^{k} p_i$. Then for any $\delta > 0$,*

Upper bound: $\mathsf{Pr}[X > (1+\delta)\mu] < \left[\dfrac{e^\delta}{(1+\delta)^{1+\delta}}\right]^\mu$.

Lower bound: $\mathsf{Pr}[X < (1-\delta)\mu] < \exp\left(\dfrac{-\mu\delta^2}{2}\right)$.

Following the analysis in [16], we obtain that for $0 < \delta \leq 2e - 1$,

$$\begin{aligned}\mathsf{Pr}[X > (1+\delta)\mu] &< \exp\left(\frac{-\mu\delta^2}{4}\right) \\ \mathsf{Pr}[X < (1-\delta)\mu] &= \exp\left(\frac{-\mu\delta^2}{2}\right) \\ &< \exp\left(\frac{-\mu\delta^2}{4}\right)\end{aligned} \tag{19}$$

Combining these, we have that for $0 < \delta \leq 2e - 1$,

$$\mathsf{Pr}[|X - \mu| \geq \delta\mu] \leq 2 \times \exp\left(\frac{-\mu\delta^2}{4}\right). \tag{20}$$

Suppose all the X_i's follow the same Bernoulli distribution with $p_i = \eta$ for all i. Then $\mu = \eta k$ and let $\eta' = X/k$. We obtain,

$$\mathsf{Pr}[|\eta' - \eta| \geq \delta\eta] \leq 2 \times \exp\left(\frac{-k\eta\delta^2}{4}\right). \tag{21}$$

Let λ be a lower bound on η, i.e., $\lambda \leq \eta$ and ϵ be such that $0 < \epsilon < 1$. We want to ensure $2 \times \exp\left(\dfrac{-k\eta\delta^2}{4}\right) \leq \dfrac{\lambda\epsilon}{8}$ and $\delta = \epsilon/8$. This gives us two conditions.

- $2 \times \exp\left(\dfrac{-k\eta\epsilon^2}{256}\right) \leq \dfrac{\lambda\epsilon}{8}$

- $\sqrt{\frac{4 \ln 1/\epsilon}{k\eta}} \leq \frac{\epsilon}{8}$.

The second inequality comes from the bound on δ. Using the fact that $\lambda \leq \eta$, it is not too difficult to work out that these two conditions hold when

$$k \geq \max\left(\frac{256}{\lambda\epsilon^2} \ln \frac{16}{\lambda\epsilon}, \frac{32}{\lambda\epsilon^2} \ln \frac{1}{\epsilon}\right). \tag{22}$$

Since $\lambda < 1$, the first component is greater and consequently we have that for $k \geq \frac{256}{\lambda\epsilon^2} \ln \frac{16}{\lambda\epsilon}$,

$$\mathsf{Pr}\left[|\eta' - \eta| \geq \frac{\eta\epsilon}{8}\right] \leq \frac{\lambda\epsilon}{8}. \tag{23}$$

Summary. A brief summary of the discussion on Chernoff bound presented so far will be helpful for later reference. In short, the situation is the following. Let $X_1, \ldots, X_k$ be independent Bernoulli trials with $\mathsf{Pr}[X_i = 1] = \eta$, where η is unknown. Let λ be lower bound on η, i.e., $\lambda \leq \eta$ and suppose that λ is known. Set $\eta' = (\sum_{i=1}^{k} X_i)/k$. From the discussion so far, we have that for $\epsilon > 0$, if $k \geq \frac{256}{\lambda\epsilon^2} \ln \frac{16}{\lambda\epsilon}$, then

$$\mathsf{Pr}\left[|\eta' - \eta| \geq \frac{\eta\epsilon}{8}\right] \leq \frac{\lambda\epsilon}{8}. \tag{24}$$

Remark. Note that the term $\frac{256}{\lambda\epsilon^2} \ln \frac{16}{\lambda\epsilon}$ gives rise to the expression $\chi(\epsilon) = O(\epsilon^{-2} \ln(\epsilon^{-1})\lambda^{-1} \ln(\lambda^{-1}))$ in Theorem 2.

A.2 Evaluating a Function

Let X (resp. Y) be a random variable which varies over a finite set Γ (resp. Σ). Suppose that the random variable X follows the uniform distribution, while the distribution of Y is unknown. Also, let X and Y be independent. Let f be a function $f : \Gamma \times \Sigma \to \{0, 1\}$ and consider the following procedure.

Procedure-A.

1. Choose x from Γ according to the uniform distribution.
2. A value y is obtained from Σ according to some unknown distribution.
3. Output $f(x, y)$.

X (resp. Y) is the random variable representing x (resp. y). The probability of outputting 1 in the above game depends on the distribution of Y. Let η_y be the probability that $f(X, y) = 1$, i.e., $\eta_y = \mathsf{Pr}[f(X, y) = 1] = \mathsf{Pr}[f(X, Y) = 1 | Y = y]$. Since the distribution of Y is not known, the value of η_y is also not known. Let λ be a known lower bound on η_y, i.e. $\lambda \leq \eta_y$ for all $y \in \Sigma$.

We want to augment Procedure-A, such that the probability of outputting 1 remains more or less the same irrespective of the choice of y. This is done in the following manner. By $\mathsf{Ber}(p)$ we mean the Bernoulli experiment where 1 is produced with probability p and 0 is produced with probability $(1-p)$.

Procedure-B.

1. Choose x from Γ according to the uniform distribution.
2. A value y is obtained from Σ according to some unknown distribution;
3. if $f(x, y) = 1$, then
4. choose $x^{(1)}, \ldots, x^{(k)}$ uniformly at random from Γ;
5. define $\eta' = \frac{\sum_{i=1}^{k} f(x^{(i)}, y)}{k}$;
6. if $\eta' \geq \lambda$, then perform $\mathsf{Ber}(\lambda/\eta')$;
7. else output 1;
8. else output 0.

The quantity η' computed in Step 5 is an estimate of η_y, for the y given in Step 2. To understand what is happening, first suppose that the estimate η' of η_y is exact. Since λ is a lower bound on η_y, the condition in Step 6 will be true and hence the above procedure will output 1 with probability $\eta \times \lambda/\eta = \lambda$. In other words, the above procedure will output 1 with probability λ irrespective of the value of y which is what we want. Now the estimate η' will not be exact and consequently the probability the Procedure-B outputs 1 will be between two bounds λ^- and λ^+ as we outline below.

A.3 Analysis of Procedure-B

We now perform the probability analysis that Procedure-B outputs 1. In the following, we will use η to denote η_y, where y is chosen in Step 2 of Procedure-B. Let ab be the event that Procedure-B outputs 0. We are interested in the event $\overline{\mathsf{ab}}$, i.e., the event that Procedure-B outputs 1. More particularly, we are interested in the probability of $\overline{\mathsf{ab}}$ when the choice of y in Step 2 of Procedure-B is fixed. Let $\overline{\mathsf{AB}}$ denote the event $(\overline{\mathsf{ab}}|Y = y)$. Next we analyse $\mathsf{Pr}[\overline{\mathsf{AB}}]$.

Let $\mathbf{A}$ be the event that $|\eta' - \eta| \leq \frac{\eta\epsilon}{8}$. Using Lemma 1, we have

$$\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}] - \mathsf{Pr}[\overline{\mathbf{A}}] \leq \mathsf{Pr}[\overline{\mathsf{AB}}] \leq \mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}] + \mathsf{Pr}[\overline{\mathbf{A}}]. \tag{25}$$

First note that using the Chernoff bound analysis, we have $\mathsf{Pr}[\overline{\mathbf{A}}] \leq \lambda\epsilon/8$. We now consider $\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}]$. There are two cases to consider.

Case $\lambda \leq \eta - \frac{\eta\epsilon}{8}$. Suppose that $\mathbf{A}$ holds. Then

$$\eta - \tfrac{\eta\epsilon}{8} \leq \eta' \leq \eta + \tfrac{\eta\epsilon}{8}. \tag{26}$$

By the condition of this case, we have $\lambda \leq \eta - \frac{\eta\epsilon}{8}$ and so $\eta' \geq \lambda$. Hence, the "if" condition in Step 6 of Procedure-B is satisfied and therefore $\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}] = \eta\lambda/\eta'$. Using (26), it is easy to work out that for $\epsilon \leq 4$,

$$\lambda\left(1 - \frac{\epsilon}{4}\right) \leq \frac{\lambda}{1 + \frac{\epsilon}{8}} \leq \frac{\lambda\eta}{\eta'} \leq \frac{\lambda}{1 - \frac{\epsilon}{8}} \leq \lambda\left(1 + \frac{\epsilon}{4}\right).$$

This combined with the bound on $\mathsf{Pr}[\overline{\mathbf{A}}]$ shows that

$$\lambda\left(1 - \frac{\epsilon}{2}\right) \leq \lambda\left(1 - \frac{\epsilon}{4}\right) - \frac{\lambda\epsilon}{8} \leq \mathsf{Pr}[\overline{\mathsf{AB}}] \leq \lambda\left(1 + \frac{\epsilon}{4}\right) + \frac{\lambda\epsilon}{8} \leq \lambda\left(1 + \frac{\epsilon}{2}\right).$$

Put differently, we have

$$|\mathsf{Pr}[\overline{\mathsf{AB}}] - \lambda| \leq \frac{\lambda\epsilon}{2}. \tag{27}$$

Case $\lambda > \eta - \frac{\eta\epsilon}{8}$. As in Case 1, let $\mathbf{A}$ be the event $|\eta' - \eta| \leq \frac{\eta\epsilon}{8}$. We identify two other events $\mathbf{A}_1$ and $\mathbf{A}_2$ as follows. $\mathbf{A}_1$ is the event $\eta - \eta\epsilon/8 \leq \eta' \leq \lambda$ and $\mathbf{A}_2$ is the event $\lambda \leq \eta' \leq \eta + \eta\epsilon/8$. Clearly, $\mathbf{A} = \mathbf{A}_1 \vee \mathbf{A}_2$ and $\mathsf{Pr}[\mathbf{A}] = \mathsf{Pr}[\mathbf{A}_1] + \mathsf{Pr}[\mathbf{A}_2]$. Now, $\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}_2] = \eta\lambda/\eta'$ and $\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}_1] = \eta$. When $\mathbf{A}_1$ occurs, $\eta' \leq \lambda$ and so $\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}_1] = \eta \leq \eta\lambda/\eta'$. We can write

$$\begin{aligned}
\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}] &= \frac{\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}_1]\mathsf{Pr}[\mathbf{A}_1] + \mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}_2]\mathsf{Pr}[\mathbf{A}_2]}{\mathsf{Pr}[\mathbf{A}]} \\
&\leq \frac{\eta\lambda}{\eta'} \\
&\leq \lambda + \frac{\lambda\epsilon}{4}.
\end{aligned}$$

The last inequality follows from the analysis of the expression $\eta\lambda/\eta'$ when $\mathbf{A}$ occurs as done for Case 1. Then, as in Case 1, we have $\mathsf{Pr}[\overline{\mathsf{AB}}] \leq \lambda + \lambda\epsilon/2$.

We next consider the lower bound. Again, using the analysis of Case 1, we have

$$\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}_2] = \frac{\lambda\eta}{\eta'} \geq \lambda - \frac{\lambda\epsilon}{4}.$$

Also,

$$\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}_1] = \eta \geq \lambda \geq \lambda - \frac{\lambda\epsilon}{4}.$$

Using these two bounds, we obtain $\mathsf{Pr}[\overline{\mathsf{AB}}|\mathbf{A}] \geq \lambda - \frac{\lambda\epsilon}{4}$ and consequently as in Case 1, $\mathsf{Pr}[\overline{\mathsf{AB}}] \geq \lambda - \frac{\lambda\epsilon}{2}$. Thus, in both Cases 1 and 2, we have $|\mathsf{Pr}[\overline{\mathsf{AB}}] - \lambda| \leq \frac{\lambda\epsilon}{2}$. Recalling that $\overline{\mathsf{AB}}$ is the event $(\overline{\mathsf{ab}}|Y = y)$, we have

$$|\mathsf{Pr}[\overline{\mathsf{ab}}|Y = y] - \lambda| \leq \frac{\lambda\epsilon}{2}. \tag{28}$$

Summary. Thus, (28) shows that the probability that Procedure-B outputs 1 is "very close" to λ. Let Σ' be any subset of Σ. Then using (28) and Lemma 2, we have

$$|\mathsf{Pr}[\overline{\mathsf{ab}}|Y \in \Sigma'] - \lambda| \leq \frac{\lambda\epsilon}{2}. \tag{29}$$

A.4 Separation of Random Variables

So far, we have been analysing probabilities without any reference to the actual adversarial game that arises in the proof of security of the HIBE protocol (Theorem 2). We now relate this analysis to the proof. First we identify the various random variables involved in the proof.

1. The quantities $x_1, \ldots, x_l$; $x'_1, \ldots, x'_h$ and $k_1, \ldots, k_h$.
2. The quantities $y_1, \ldots, y_l$ and $y'_1, \ldots, y'_h$.
3. The public parameters $U_1, \ldots, U_l$ and $U'_1, \ldots, U'_h$. Note that the U's are fixed once the x's, k's and the y's are fixed. On the other hand, we could randomly choose the U's, the x's, and the k's and then appropriately fix the y's. This is the view that will be required in the following analysis.
4. The random bit γ chosen during the challenge step.
5. The random elements r's chosen during the simulation of key extraction oracle queries.
6. The random bits used by the adversary.
7. The random instance of the DBDH problem.

The adversary's view is determined by Items 3 to 7 above. Let y be the concatenation of all these quantities and Σ be the set of all possible y's. Let x be the concatenation of all the quantities in Item 1 above and Γ be the set of all possible x's. Let X and Y be random variables over Γ and Σ respectively. Then, X is uniformly distributed and is independent of Y. The distribution of Y on the other hand need not be uniform as it depends on the random bits used by the adversary. But, to apply Lemma 2 we need X and Y to be independent, which holds in the present case.

Suppose we fix a particular y in Σ. Then all the randomness encountered by the adversary is fixed while we can still vary x uniformly at random from Γ. Thus, for a fixed y, the adversary becomes deterministic and hence a particular value of y determines all the key extraction queries, the challenge identity as well as the adversary's guess γ'. We identify the subset Σ_0 of Σ to be such that $\gamma = \gamma'$, i.e., $\Sigma_0 = \{y : \gamma = \gamma'\}$. This set represents all possible adversarial behaviours where the adversary is successful (in breaking the protocol). Also, we define $\Sigma_1 = \Sigma \setminus \Sigma_0$, i.e., $\Sigma_1 = \{y : \gamma \neq \gamma'\}$.

Let $f : \Gamma \times \Sigma \rightarrow \{0,1\}$ be a function defined as follows. For $x \in \Gamma$ and $y \in \Sigma$, $f(x,y)$ is defined to be 1 if the adversary does not abort during the simulation (i.e., if flg is set to 0 at the end of the simulation). In Proposition 3, we have already obtained a lower bound λ on $\mathsf{Pr}[f(X,Y) = 1|Y = y]$.

The evaluation procedure for f is exactly that of Procedure-A in Section A.2. Procedure-B bounds the probability of outputting 1 by outputting 0 in certain cases, even when Procedure-A outputs 1. In terms of the simulator, this means that Procedure-B aborts in certain cases even when Procedure-A does not abort. This additional abort has been termed "artificial abort" by Waters [20].

When we interpret Procedure-B in terms of the security proof, several things should be kept in mind. Steps 1 and 2 of Procedure-B correspond to the portion of the simulator which is concerned with answering the adversary's queries and generating the challenge. From the probability point of view, the actual algebraic techniques for doing this is not important. The important thing to note is that for the choice of x in Step 1 and the choice of y in Step 2 of Procedure-B the value of $f(x,y)$ becomes fixed. The condition $f(x,y) = 1$ corresponds to the fact that the simulator does not abort for this choice of x and y. Steps 4 to 6 of Procedure-B correspond to the artificial abort stage. At this point y is fixed, which means that the adversary's queries and the challenge identity has been fixed. In Steps 4 and 5 of Procedure-B, $x^{(1)}, \ldots, x^{(k)}$ are chosen uniformly at random from Σ and $f(x^{(i)}, y)$ is evaluated. (The quantities $x^{(i)}$'s in Steps 4 and 5 of Procedure-B should not be confused with $x_1, \ldots, x_l$ and $x'_1, \ldots, x'_h$ chosen as part of the security proof of Theorem 2.) In terms of the security proof, this means that for the fixed choice of the adversary's queries and the challenge identity, the simulation is executed a total of k times with a different random choice of $x_1, \ldots, x_l$ and $x'_1, \ldots, x'_h$ each time. The evaluation $f(x^{(i)}, y)$ records whether the simulator has to abort on the ith execution. One important thing to note is that this procedure does not require the adversary's participation.

We now follow the analysis of Procedure-B given in Section A.3. Successively using Σ_0 and Σ_1 in place of Σ' in (29), we have,

$$\lambda - \frac{\lambda\epsilon}{2} \leq \mathsf{Pr}[\overline{\mathsf{ab}}|\gamma = \gamma'], \mathsf{Pr}[\overline{\mathsf{ab}}|\gamma \neq \gamma'], \leq \lambda + \frac{\lambda\epsilon}{2}. \tag{30}$$

Using this equation, the rest of the proof of Theorem 2 has already been described.

A.5 Why Artificial Abort?

An intuitive justification for artificial abort was provided by Waters. The argument is that it is possible for $|\mathsf{Pr}[\gamma = \gamma'|\overline{\mathsf{ab}}] - 1/2|$ to be negligible even when $|\mathsf{Pr}[\gamma = \gamma'] - 1/2|$ is non-negligible. The problem is that the probability of not aborting (in the actual game) depends on the adversary's query. This creates a problem in the probability calculation as we discuss below.

Let us go back to Equation (7). We have

$$\mathsf{Pr}[Y_i] = \frac{1}{2}(1 - \mathsf{Pr}[\overline{\mathsf{ab}_i}]) + \mathsf{Pr}[X_i \mid \overline{\mathsf{ab}_i}]\mathsf{Pr}[\overline{\mathsf{ab}_i}].$$

Now suppose, we have $\mathsf{Pr}[\overline{\mathsf{ab}_i} \mid X_i] = \mathsf{Pr}[\overline{\mathsf{ab}_i}]$. Then $\mathsf{Pr}[X_i \mid \overline{\mathsf{ab}_i}]\mathsf{Pr}[\overline{\mathsf{ab}_i}] = \mathsf{Pr}[\overline{\mathsf{ab}_i}]\mathsf{Pr}[X_i] = \lambda\mathsf{Pr}[X_i]$ and it is easy to work out that

$$|\mathsf{Pr}[X_1] - \mathsf{Pr}[X_2]| \leq \frac{|\mathsf{Pr}[Y_1] - \mathsf{Pr}[Y_2]|}{\lambda} \leq \frac{\epsilon_{dbdh}}{\lambda}.$$

Thus, the independence condition $\mathsf{Pr}[\overline{\mathsf{ab}_i} \mid X_i] = \mathsf{Pr}[\overline{\mathsf{ab}_i}]$ makes the analysis easy. Not having this independence condition forces the use of artificial abort technique.

A.6 Knowing a Lower Bound is Not Sufficient

The artificial abort technique ensures that $\mathsf{Pr}[\overline{\mathsf{ab}}|\gamma = \gamma']$ is between two bounds λ^- and λ^+, i.e.,

$$\lambda^- \leq \mathsf{Pr}[\overline{\mathsf{ab}}|\gamma = \gamma'] \leq \lambda^+. \tag{31}$$

Even without artificial abort, we know that if we substitute λ for λ^- and 1 for λ^+, then (31) holds. We would like to point out these bounds are not good enough to obtain the desired relationship between ϵ_{hibe} and ϵ_{dbdh}.

To see this, we go back to the proof of Claim in Theorem 2. In terms of the quantities in the proof of the Claim, we have $\lambda^- \leq A_i, C_i \leq \lambda^+$, for $i = 1, 2$. Also, $2D = (A_1B_1 - C_1(1-B_1)) - (A_2B_2 - C_2(1-B_2))$, where $|D| \leq \epsilon_{dbdh}$. Now, inequality calculations as in the proof of Claim shows that

$$|(\lambda^+ + \lambda^-)(B_1 - B_2) - 2D| \leq \lambda^+ - \lambda^-.$$

Using the fact that $|x| - |y| \leq |x - y|$, we have

$$\left|\frac{(\lambda^+ + \lambda^-)(B_1 - B_2)}{2}\right| - |D| \leq \frac{\lambda^+ - \lambda^-}{2}.$$

From this, we get

$$|B_1 - B_2| \leq \frac{2}{\lambda^+ + \lambda^-}\epsilon_{dbdh} + \frac{\lambda^+ - \lambda^-}{\lambda^+ + \lambda^-}.$$

If $\lambda^- = \lambda$ and $\lambda^+ = 1$, then we have

$$|B_1 - B_2| \leq \frac{2}{1 + \lambda}\epsilon_{dbdh} + \frac{1 - \lambda}{1 + \lambda}.$$

The second term is significantly high and cannot be upper bounded by a constant multiple of ϵ. On the other hand, if we put $\lambda^- = \lambda - \lambda\epsilon/2$ and $\lambda^+ = \lambda + \lambda\epsilon/2$, then

$$|B_1 - B_2| \leq \frac{\epsilon_{dbdh}}{\lambda} + \frac{\epsilon}{2}.$$

The last expression provides the required relationship between ϵ and ϵ_{dbdh}, i.e., Equation (15).