The ISC Int'l Journal of Information Security

October 2025, Volume 17, Number 3 (pp. 1–12)

http://www.isecure-journal.org

Presented at the ISCISC'2025 in TEHRAN, Iran.

Secure Pairing-Free IBE and CP-ABE from Inner-Product Functional Encryption **

Ahmad Khoureich Ka [1,*]

[1] *Alioune Diop University of Bambey, Senegal.*

ARTICLE INFO.

Keywords:
Identity-Based Encryption, Attribute-Based Encryption, Inner-Product Functional Encryption, Decisional Diffie-Hellman.

Type:

doi:

Abstract

The potential of Attribute-Based Encryption (ABE) in the context of IoT has driven researchers to propose pairing-free ABE schemes that are suitable for resource-constrained devices. Unfortunately, many of these schemes turned out to be insecure. This fact reinforces the view of some researchers according to which instantiating an Identity-Based Encryption (IBE) in plain Decisional Diffie-Hellman (DDH) groups is impossible. In this paper, we provide a generic Ciphertext-Policy ABE (CP-ABE) scheme supporting secret AND-gate policy using Inner-Product Functional Encryption (IPFE). We also propose an instantiation of our generic CP-ABE scheme based on the DDH assumption. From our generic CP-ABE scheme, we derive an IBE scheme by introducing the concept of Clustered Identity-Based Encryption (CIBE). Our schemes show that it is possible to construct secure IBE and ABE schemes based on the classical DDH assumption. An implementation of our CIBE in Python using the Charm framework is available on GitHub.

1 Introduction

Attribute -Based Encryption (ABE) is an asymmetric encryption primitive that allows decryption only if an access policy is satisfied [1–3]. The access policy considered as a predicate $P(\cdot)$ is defined on attributes, which are descriptive characteristics of users. We distinguish two types of ABE. On one side, Key-Policy ABE (KP-ABE) schemes where the access policies are embedded in keys and the set of attributes S is in the ciphertext and on the other side, Ciphertext-Policy ABE (CP-ABE) schemes in which the access policy $P(\cdot)$ is integrated into the ciphertext and keys are associated with sets of attributes. In both cases, decryption is possible only if $P(S) = 1$.

ABE has difficulty penetrating the world of resource-constrained IoT devices because lightweight schemes have a reputation of having crippling security flaws [4–6], and available secure ABE schemes are resource-intensive. Secure ABE schemes are based on bilinear pairings [7–9] or lattices [10–12] but suffer from either a large decryption key size or slow encryption or decryption. Suitable ABE schemes for lightweight devices [6, 13–15] exploit classical security assumptions like RSA or the Decisional Diffie-Hellman (DDH), hence their generic name, pairing-free ABE. The discovery of security vulner-

* Corresponding author.

**The ISCISC'2025 program committee effort is highly acknowledged for reviewing this paper.

Email address: ahmadkhoureich.ka@uadb.edu.sn

ISSN: 2008-2045

abilities in many of those schemes is not surprising to some authors [5, 16, 17] since they conjectured (*the impossibility conjecture*) that designing Identity-Based Encryption (IBE) schemes relying on trapdoor permutations or the DDH assumption in a black box manner is impossible. Furthermore, Herranz [5] has claimed that there are many chances that an ABE scheme that works in classical settings (RSA or pairing-free discrete logarithm) is not secure.

1.1 Our Contributions

We provide a generic construction of AND-gate access policy CP-ABE from Inner-Product Functional Encryption (IPFE). IPFE has been used in various constructions, notably in the non-zero inner-product encryption [18], the decentralized ABE against bounded collusion [19] and the message-selection functional encryption [20].

Our construction is a secret policy CP-ABE scheme. Although the ciphertext hides the message and the encryption policy, our CP-ABE scheme is not attribute-hiding in the sense of Katz *et al.* [21], because an adversary can succeed the indistinguishability experiment, where it is asked to guess which of the tuples (m_0, p_0), (m_1, p_1) was encrypted (m_i is the message and p_i is the encryption policy). The secrecy of the access policy and the large policy space make collusion and exhaustive-search attacks infeasible. Secret-policy CP-ABE and fully hidden-policy CP-ABE provide better confidentiality than standard CP-ABE. Knowledge of the attributes embedded within the ciphertext can lead to a collapse of the probability distribution on the plaintext space (the set of likely messages a priori shrinks to a much smaller one). For example, in wartime, intercepting a ciphertext containing attributes referring to a special operations unit of a military branch offers a good chance of guessing the enemy's next move.

From our generic CP-ABE scheme, we derive a *Small Identity Space* IBE (SIS-IBE) that can only issue $\ell - 1$ secret keys, where ℓ is the functionality parameter of the underlying IPFE. We introduce the concept of *Clustered* IBE (CIBE) to convert the SIS-IBE scheme into a large identity space IBE scheme.

By instantiating our CP-ABE scheme from the DDH-based IPFE of [22], we show that it is possible to construct practical and secure ABE and IBE schemes based on the DDH assumption.

1.2 Organization

The rest of this paper is organised as follows. In Section 2 we summarise related work. Section 3 presents IPFE and CP-ABE. Our generic CP-ABE scheme is presented in Section 4. In Section 5, we present a generic IBE scheme. Section 6 concludes this work.

2 Related work

IBE and ABE are useful tools in IoT systems. ABE ensures that only IoT devices that have the required attributes can access sensitive data. IBE simplifies the deployment of public-key encryption, as the device's identity can serve as public key. Many pairing-free ABE schemes suitable for resource-constrained devices have been published. Unfortunately, many of those schemes turned out to be insecure. CP-ABE schemes in [23, 24] use scalar multiplication on elliptic curves, but they are found vulnerable to collusion attacks [25]. The KP-ABE scheme proposed in [15] is based on the Elliptic Curve Decisional Diffie-Hellman (ECDDH) assumption. However, it it has been shown in [6] that an adversary can decrypt a ciphertext without having the required attributes. Recently, Hamednejad *et al.* [26] succeeded in adding two proposals [27, 28] to the set of insecure pairing-free CP-ABE schemes. They have shown that [27, 28] are vulnerable to collusion attacks. All these failures tend to reinforce the impossibility conjecture [5, 16, 17], which states that it is impossible to design secure IBE schemes that rely solely on the hardness of the Diffie-Hellman Problem. Nonetheless, attacks against the impossibility conjecture debuted with the work of Döttling and Garg [29] who proposed a construction of IBE under DDH using Garbled Circuits. Thereafter, Blazy and Kakvi [30] provided a construction of IBE similar to that of Döttling and Garg based on Witness Encryption instead of garbled circuits. Unfortunately, these schemes are inefficient and therefore do not completely contradict the impossibility conjecture. In 2024, Yao *et al.* [31] proposed a lightweight pairing-free CP-ABE scheme leveraging the masked authenticated message mechanism of the IOTA protocol to manage authorisation. Moreover, Wang *et al.* [32] recently proposed a KP-ABE scheme relying on the DDH assumption. In addition to requiring no bilinear pairings, their scheme has a decentralized architecture.

3 Preliminaries

3.1 Inner-Product Functional Encryption

Inner-Product Functional Encryption allows a recipient who has a secret key derived from a vector $\boldsymbol{x}$ to obtain from the encryption of a vector $\boldsymbol{y}$ the inner product $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ and nothing more. Abdalla et *al.* are the first to formalise the notion of IPFE [33]. The syntax of IPFE and its INDistinguishability under Chosen Plaintext Attack (IND-CPA) security are given in Appendix A.

3.2 Identity-Based Encryption

In 1984 Shamir [34] introduced the concept of identity-based encryption. IBE is an attractive public key encryption (PKE) primitive because there is no need to tie a random public key to the owner's identity by using a certificate management system (as in traditional PKE). In IBE, the user's identity is their public key. The syntax of IBE and its indistinguishability-based security are given in Appendix C.

3.3 Ciphertext-policy ABE

A Ciphertext-Policy ABE allows a recipient who has a secret key associated with a set of attributes $\mathbb{S}$ to correctly decrypt a ciphertext integrating an access policy $\mathbb{P}$ if and only if $\mathbb{S}$ satisfies $\mathbb{P}$. In this work, we restrict ourselves to access policies that consist of a single AND gate. Also, our construction is a secret access policy, CP-ABE. The syntax of CP-ABE and the indistinguishability-based security for a secret policy CP-ABE scheme are given in Appendix B.

4 Construction of CP-ABE from IPFE

To prevent collusion attacks, our construction is a secret access policy CP-ABE scheme. We use IPFE as building blocks. Let ℓ be the functionality parameter of that IPFE. We denote by $[\ell]$ the set of integers $\{1, \ldots, \ell\}$. Let the attribute space be $\mathbb{Z}_q^* \times [\ell]$ for some prime q. Our construction operates on sets of attributes which have at most a cardinality ℓ where the second coordinates of the attributes are distinct. A set of attributes $\mathbb{S}$ can be represented as a vector $\boldsymbol{S} \in \mathbb{Z}_q^\ell$. For example, if $\ell = 5$, we represent the set of attributes $\mathbb{S} = \{(12, 3), (25, 5)\}$ as $\boldsymbol{S} = (0, 0, 12, 0, 25)$. The algorithm for converting a set of attributes $\mathbb{S} = \{(a_i, b_i) \in (\mathbb{Z}_q^* \times [\ell])\}_{i=1}^{n \leq \ell}$ to a vector $\boldsymbol{S} \in \mathbb{Z}_q^\ell$ is defined as follow:

Algorithm $toVector(\mathbb{S})$
Set $\boldsymbol{S} = (s_1, \ldots, s_\ell) \in \{0\}^\ell$
`For each` $(a_i, b_i) \in \mathbb{S}$ `do` $s_{b_i} = a_i$
`EndFor`
`Return` $\boldsymbol{S}$

Given a ring $\mathcal{R}$ which is either $\mathbb{Z}$ or $\mathbb{Z}_q$, we denote the coordinate-wise product of two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathcal{R}^\ell$ by $\boldsymbol{x} \odot \boldsymbol{y} = (x_1 y_1, \ldots, x_\ell y_\ell) \in \mathcal{R}^\ell$.

Now, we describe the construction of our generic CP-ABE scheme with secret AND-gate policy from stateful IPFE. The stateless version, where the underlying IPFE computes inner products in $\mathbb{Z}$, is syntactically identical but does not require a stateful key generation algorithm.

$\mathsf{Setup}(1^\lambda, 1^\ell)$. Given as input the security parameter λ and the functionality parameter ℓ of the underlying IPFE, this algorithm performs the following steps:

(1) Choose a cyclic group $\mathbb{G}$ of prime order $q > 2^\lambda$ with generator g.
(2) Call $\mathsf{IPFE.Setup}(1^\lambda, 1^\ell)$ to obtain a master secret key msk and a master public key mpk.

$\mathsf{KeyGen}(msk, st, \mathbb{S})$. Given as input the master secret key msk, the internal state st used by the underlying stateful IPFE and a set of attributes $\mathbb{S} \in (\mathbb{Z}_q^* \times [\ell])^n$ where $n \leq \ell$, this algorithm performs the following steps:

(1) Convert the set of attributes $\mathbb{S}$ to a vector $\boldsymbol{S} = (s_1, \ldots, s_\ell) \in \mathbb{Z}_q^\ell$,
(2) Call $\mathsf{IPFE.KeyGen}(msk, st, \boldsymbol{S})$ to obtain a secret key $sk_{\boldsymbol{S}}$ and an updated state st. Notice that in our scheme, $\boldsymbol{S}$ is considered secret and is therefore included in $sk_{\boldsymbol{S}}$.
(3) Return $(sk_{\boldsymbol{S}}, st)$.

$\mathsf{Encrypt}(mpk, \mathbb{P}, m)$. Given as input the master public key mpk, a secret access policy consisting of a set of attributes $\mathbb{P} \in (\mathbb{Z}_q^* \times [\ell])^n$ where $n \leq \ell$ and a message $m \in \mathbb{G}$, this algorithm performs the following steps:

(1) Convert the access policy $\mathbb{P}$ to a vector $\boldsymbol{P} = (p_1, \ldots, p_\ell) \in \mathbb{Z}_q^\ell$,
(2) Choose a random vector $\boldsymbol{r} = (r_1, \ldots, r_\ell) \in (\mathbb{Z}_q^*)^\ell$.
(3) Set $\boldsymbol{y} = \boldsymbol{P} \odot \boldsymbol{r} = (p_1 r_1, \ldots, p_\ell r_\ell)$.
(4) Compute $c_1 = m g^{\langle \boldsymbol{P}, \boldsymbol{y} \rangle}$.
(5) Compute $c_2 \leftarrow \mathsf{IPFE.Encrypt}(mpk, \boldsymbol{y})$.
(6) Return the ciphertext $C = (c_1, c_2)$.

$\mathsf{Decrypt}(mpk, sk_{\boldsymbol{S}}, C)$. Given as input the master public key mpk, a secret key $sk_{\boldsymbol{S}}$ and a ciphertext $C = (c_1, c_2)$, this algorithm performs the following steps:

(1) Compute $\sigma = \mathsf{IPFE.Decrypt}(mpk, sk_{\boldsymbol{S}}, c_2)$.
(2) Return the plaintext $m = c_1 / g^\sigma$.

Correctness. We recall that in an AND-gate access policy CP-ABE scheme, $\mathbb{S}$ satisfies $\mathbb{P}$ if $\mathbb{S} \supseteq \mathbb{P}$ meaning that for all $i \in [\ell]$ if $p_i \neq 0$ then $s_i = p_i$. Therefore, $\mathbb{S} \supseteq \mathbb{P} \iff \boldsymbol{S} \odot \boldsymbol{P} = \boldsymbol{P} \odot \boldsymbol{P}$. The correctness of the scheme follows from the observation that:

$$\begin{aligned}\sigma &= \mathsf{IPFE.Decrypt}(mpk, sk_{\boldsymbol{S}}, c_2) \\ &= \mathsf{IPFE.Decrypt}(mpk, sk_{\boldsymbol{S}}, \mathsf{IPFE.Encrypt}(mpk, \boldsymbol{y})) \\ &= \langle \boldsymbol{S}, \boldsymbol{y} \rangle = \langle \boldsymbol{S}, \boldsymbol{P} \odot \boldsymbol{r} \rangle = \langle \boldsymbol{S} \odot \boldsymbol{P}, \boldsymbol{r} \rangle = \langle \boldsymbol{P} \odot \boldsymbol{P}, \boldsymbol{r} \rangle \\ &= \langle \boldsymbol{P}, \boldsymbol{P} \odot \boldsymbol{r} \rangle = \langle \boldsymbol{P}, \boldsymbol{y} \rangle\end{aligned}$$

Therefore, $c_1 / g^\sigma = c_1 / g^{\langle \boldsymbol{P}, \boldsymbol{y} \rangle} = m$.

4.1 Security

We recall that, in our setting, the encryptor keeps the access policy secret and that the attributes associated

with a secret key are also secret. These requirements are necessary to ensure that collusion attacks are infeasible. If the access policy is known, the adversary can find vectors that do not satisfy the access policy but for which there exists a linear combination that does satisfy it. The adversary can request secret keys associated with these vectors. By decrypting c_2 (the IPFE component of the ciphertext) with each of these secret keys and summing the set of obtained numbers, the adversary recovers the secret $\langle \boldsymbol{P}, \boldsymbol{y} \rangle$ used to mask the plaintext. The same reasoning applies when the set of attributes associated with a secret key that can decrypt the ciphertext is discovered. With these requirements in mind, the security of the underlying IPFE scheme reduces to the security of our CP-ABE scheme.

Theorem 1. *If the underlying IPFE scheme is* IND-CPA *secure, then our CP-ABE scheme is also* IND-CPA *secure. (The proof is presented in Appendix D.)*

4.2 Instantiation from the DDH Assumption

We present an instantiation of our generic CP-ABE scheme from a modified version of the DDH-based IPFE scheme of [22] that we denote IPFE-DDH*. The modification is only made to the decryption algorithm to return $g_1^{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$ instead of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$, thus avoiding the computation of the discrete logarithm, which is too expensive. IPFE-DDH* is presented in Appendix E. The KeyGen algorithm of this instantiation proceeds exactly as in the generic construction in Section 4. Therefore, we only present algorithms (or steps) that change.

$\mathsf{Setup}(1^\lambda, 1^\ell)$

2. Call $\mathsf{IPFE\text{-}DDH}^\star.\mathsf{Setup}(1^\lambda, 1^\ell)$ to obtain the master secret key msk and a master public key $mpk = (\mathbb{G}, g_1, g_2, \{h_i\}_{i=1}^{\ell})$.

$\mathsf{Encrypt}(mpk, \mathbb{P}, m)$

4. Compute $c_1 = m g_1^{\langle \boldsymbol{P}, \boldsymbol{y} \rangle}$.

$\mathsf{Decrypt}(mpk, sk_{\boldsymbol{S}}, C)$

(1) Compute $\rho = \mathsf{IPFE\text{-}DDH}^\star.\mathsf{Decrypt}(mpk, sk_{\boldsymbol{S}}, c_2)$

(2) Return the plaintext $m = c_1/\rho$.

4.3 Theoretical Evaluation

We perform a theoretical comparison of our DDH-based CP-ABE scheme with other schemes such as FAME [7], FABEO [8] and Easy-ABE [9]. All these schemes are pairing-based ABE schemes. This comparison is made only on AND-gate access policies (the only ones supported by our scheme). FAME and FABEO support access policies described by boolean formulas that are convertible into Monotone Span Programs (MSPs). An MSP consists of a matrix M and a mapping π that assigns each row of M to an attribute.

For many ABE schemes, it is required that π be injective (the access policy cannot use the same attribute multiple times). This is termed the one-use restriction. Techniques [35] to avoid the one-use restriction can be used, but lead to a less efficient scheme [8]. Easy-ABE does not use MSPs, therefore does not suffer from the one-use restriction. It derives the collection of authorised sets of attributes from the access policy, then converts each of them into a bit string using the universe of attributes. In this comparison (see Table 1, Table 2, Table 3 and Table 4), we use the one-use restriction versions of FAME and FABEO.

Some may think that it is unfair to compare our CP-ABE scheme to pairing-based CP-ABE schemes that support more complex policies. Nevertheless, this comparison provides insight into the behaviour of our scheme in the setting of AND-gate access policy.

Table 1. Number of group operations used for key generation. T denotes the number of attributes input to the KeyGen algorithm, ℓ is the functionality parameter of the DDH-based IPFE.

	Key generation			
	$\mathbb{G}_1$ or $\mathbb{Z}_q$ for Our			$\mathbb{G}_2$
Scheme	Mul	Exp	Hash	Exp
FAME	$8T+9$	$9T+9$	$6(T+1)$	3
FABEO	1	$T+2$	$T+1$	1
Easy-ABE	1	2	1	1
Our	2ℓ	–	–	–

Table 2. Number of group operations used for encryption. n_1, n_2 are the number of rows and columns of the MSP in FAME and FABEO, m is the number of attribute strings of the access policy in Easy-ABE, ℓ is the functionality parameter of the DDH-based IPFE.

	Encryption			
	$\mathbb{G}_1$			$\mathbb{G}_2$
Scheme	Mul	Exp	Hash	Exp
FAME	$12n_1n_2 + 6n_1$	$6n_1$	$6(n_1+n_2)$	3
FABEO	n_1	$2n_1$	n_1+1	2
Easy-ABE	1	$m+2$	m	1
Our	$2\ell+1$	$2\ell+3$	–	–

Table 3. Number of group operations used for decryption. I is the number of attributes used in decryption, ℓ is the functionality parameter of the DDH-based IPFE.

	Decryption			
	$\mathbb{G}_1$		$\mathbb{G}_T$	
Scheme	Mul	Exp	Mul	Pairing
FAME	$6I+3$	–	6	6
FABEO	$2I$	–	3	3
Easy-ABE	–	–	1	2
Our	$\ell+2$	$\ell+2$	–	–

Table 4. Size of keys and ciphertexts. T denotes the number of attributes input to the KeyGen algorithm, n_1 is the number of rows of the MSPs in FAME and FABEO, m is the number of attribute strings of the access policy in Easy-ABE, ℓ is the functionality parameter of the DDH-based IPFE.

	Storage cost			
	Key size		Ciphertext size	
Scheme	$\mathbb{G}_1$ or $\mathbb{Z}_q$ for Our	$\mathbb{G}_2$	$\mathbb{G}_1$	$\mathbb{G}_2$
FAME	$3(T+1)$	3	$3n_1$	3
FABEO	$T+1$	1	n_1	2
Easy-ABE	1	1	$m+1$	1
Our	$\ell+2$	–	$\ell+3$	–

4.4 Experimental Evaluation

We implemented our DDH-based CP-ABE scheme using the Charm 0.5 framework [36]. The implementations of FAME [7], FABEO [8] and Easy-ABE [9] are taken from GitHub [37]. All the schemes are run on an Ubuntu 22.04 desktop computer with an Intel Core i5-4590S CPU and 8GB RAM. The pairing-based schemes (FAME, FABEO, Easy-ABE) use the MNT224 curve, our scheme uses the group $\mathbb{G}_1$ of the MNT224 curve. The experiments are performed for all four algorithms in each scheme. For FAME, FABEO and Easy-ABE, we use AND-gate access policies of the form "1 AND 2 AND ... AND N" where attributes are in $[N]$ for $N \in \{10, 20, \ldots, 100\}$ as in [7–9]. For our DDH-based CP-ABE scheme, we use the same form of access policies, but the attributes are large integers chosen randomly from the ZR integer group of the Charm framework. Indeed, the size of the numbers used as attributes has an impact on the execution times of our scheme, but does not affect the other schemes. Results are obtained by averaging over 20 executions.

Figure 1 shows how the running times of the setup, key generation, encryption, and decryption algorithms scale with the size of the access policy or the number of attributes.

Table 5. Size of keys and ciphertexts for 100 attributes with the AND-gate policy. Ciphertext sizes are obtained by encrypting a random plaintext of 1024 bytes.

	Storage cost (bytes)	
Scheme	Key size	Ciphertext size
FAME	20800	23322
FABEO	7695	10300
Easy-ABE	435	2261
Our	6762	9050

Note that Table 5, which gives the storage cost of the different schemes, is consistent with Table 4.

5 The case $n = \ell$ in our CP-ABE scheme

In our CP-ABE construction, the AND-gate access policy is a set of attributes, $\mathbb{P} \in (\mathbb{Z}_q^* \times [\ell])^n$ where $n \leq \ell$. The case $n = \ell$ allows us to derive an Identity-Based Encryption scheme that we construct in two stages. First, we construct in Section 5.1 a *small-identity-space* IBE that (to be secure) can only issue a small number of secret keys. Next, we overcome this limitation in Section 5.2 by introducing the notion of *clustered* IBE, which consists of grouping many instances of our small-identity-space IBE. The resulting scheme has a large ID space and is secure against adaptive chosen-plaintext attacks.

5.1 Small ID Space Identity-Based Encryption

As its name suggests, this IBE scheme can only issue a small number of secret keys. Therefore, the underlying IPFE does not need to be stateful since fewer than ℓ (its functionality parameter) keys are produced.

$\mathsf{Setup}(1^\lambda, 1^\ell)$. λ is the security parameter and ℓ is the functionality parameter of the underlying IPFE. The setup algorithm performs the following steps:

(1) Choose a cyclic group $\mathbb{G}$ of prime order $q > 2^\lambda$ with generator g.
(2) Run $(mpk, msk) \leftarrow \mathsf{IPFE.Setup}(1^\lambda, 1^\ell)$.
(3) Create a list $\mathbb{L}$ (initially empty) of vectors in $(\mathbb{Z}_q^*)^\ell$. $\mathbb{L}$ has a maximum capacity of $\ell - 1$ vectors.
(4) Define $params = (\mathbb{G}, g, mpk, \mathbb{L})$.

$\mathsf{KeyGen}(params, msk, \mathsf{ID})$. Given $\mathsf{ID} \in [\ell - 1]$, this algorithm performs the following steps:

(1) Check if the vector $\boldsymbol{x}_{\mathsf{ID}}$ is in $\mathbb{L}$. $\boldsymbol{x}_{\mathsf{ID}}$ is the random vector in $(\mathbb{Z}_q^*)^\ell$ associated to ID.
(2) If yes, run $sk_{\boldsymbol{x}_{\mathsf{ID}}} \leftarrow \mathsf{IPFE.KeyGen}(msk, \boldsymbol{x}_{\mathsf{ID}})$ and return $sk_{\boldsymbol{x}_{\mathsf{ID}}}$.
(3) Otherwise:
  3.1. Pick a random $\boldsymbol{x}_{\mathsf{ID}} \in (\mathbb{Z}_q^*)^\ell$ such that the set of vectors $\{\boldsymbol{x}\}_{\boldsymbol{x} \in \mathbb{L}} \cup \{\boldsymbol{x}_{\mathsf{ID}}\}$ is linearly independent, and add $\boldsymbol{x}_{\mathsf{ID}}$ to $\mathbb{L}$.
  3.2. Run $sk_{\boldsymbol{x}_{\mathsf{ID}}} \leftarrow \mathsf{IPFE.KeyGen}(msk, \boldsymbol{x}_{\mathsf{ID}})$ and return $sk_{\boldsymbol{x}_{\mathsf{ID}}}$.

$\mathsf{Encrypt}(params, \mathsf{ID}, m)$. Given $\mathsf{ID} \in [\ell - 1]$ and $m \in \mathbb{G}$, this algorithm performs the following steps:

(1) Check if $\boldsymbol{x}_{\mathsf{ID}}$ is in $\mathbb{L}$.
(2) If no then abort. We require that encryption is only possible for identities for which a secret key is already produced.
(3) Otherwise:
  3.1. $\boldsymbol{x}_{\mathsf{ID}} = (x_1, \ldots, x_\ell) \in \mathbb{L}$.
  3.2. Choose a random vector $\boldsymbol{r} = (r_1, \ldots, r_\ell) \in (\mathbb{Z}_q^*)^\ell$.
  3.3. Set $\boldsymbol{y} = \boldsymbol{x}_{\mathsf{ID}} \odot \boldsymbol{r} = (x_1 r_1, \ldots, x_\ell r_\ell)$.
  3.4. Compute $c_1 = m g^{\langle \boldsymbol{x}_{\mathsf{ID}}, \boldsymbol{y} \rangle}$.
  3.5. Compute $c_2 \leftarrow \mathsf{IPFE.Encrypt}(mpk, \boldsymbol{y})$.

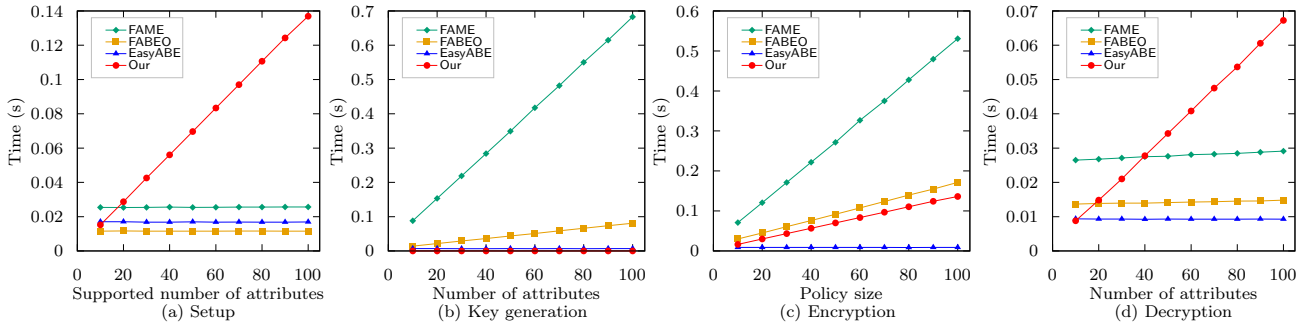

Figure 1. Average running times for the setup, key generation, encryption and decryption algorithms with AND-gate access policies. The pairing based schemes support public and expressive policies while our DDH-based CP-ABE only supports AND-gates policy. This comparison serves only to have a reference for the order of magnitude of the execution times of our scheme.

3.6. Return the ciphertext $C = (c_1, c_2)$.

$\mathsf{Decrypt}(params, sk_{\boldsymbol{x}_{\mathsf{ID}}}, C)$. Given a secret key $sk_{\boldsymbol{x}_{\mathsf{ID}}}$ and a ciphertext $C = (c_1, c_2)$ this algorithm performs the following steps:

(1) Compute $\sigma = \mathsf{IPFE.Decrypt}(mpk, sk_{\boldsymbol{x}_{\mathsf{ID}}}, c_2)$.
(2) Return the plaintext $m = c_1/g^{\sigma}$.

5.1.1 Security

Intuitively, the restriction on the size of the identity space ($\ell - 1$ identities corresponding to $\ell - 1$ linearly independent vectors) guarantees our SIS-IBE scheme resistance to collusion attacks and prevents an adversary from recovering $\boldsymbol{y}$ from c_2. Therefore, the security of the underlying IPFE scheme reduces to the security of our SIS-IBE scheme.

Theorem 2. *If the underlying IPFE scheme is* IND-CPA *secure, then our small-identity-space IBE scheme is also* IND-CPA *secure. (The proof is presented in Appendix F.)*

5.2 Extensions

Our SIS-IBE scheme can only support $\ell - 1$ identities where ℓ is the functionality parameter of the underlying IPFE scheme. This limitation can be overcome by grouping multiple SIS-IBE instances. We call the resulting scheme a *Clustered Identity-Based Encryption*. A Python implementation of our clustered IBE is available on GitHub [38]. Hereafter is a description of its construction.

$\mathsf{Setup}(1^{\lambda}, 1^{\ell})$. λ is the security parameter and ℓ is the functionality parameter of the underlying IPFE. The setup algorithm performs the following steps:

(1) Choose a cyclic group $\mathbb{G}$ of prime order $q > 2^{\lambda}$ with generator g.
(2) Define two empty lists denoted MPK (of master public keys) and MSK (of master secret keys).
(3) Create a set of lists $\{\mathbb{L}_{\alpha}\}_{\alpha \in \mathbb{Z}}$ initially empty. Each $\mathbb{L}_{\alpha}$ is initially empty and has a maximum capacity of $\ell - 1$ vectors.
(4) Define $params = (1^{\lambda}, 1^{\ell}, \mathbb{G}, g, \mathsf{MPK}, \{\mathbb{L}_{\alpha}\}_{\alpha \in \mathbb{Z}})$.

$\mathsf{KeyGen}(params, \mathsf{MSK}, \mathsf{ID})$. Given $\mathsf{ID} \in \mathbb{Z}$, this algorithm performs the following steps:

(1) Compute $\alpha = \lfloor \frac{\mathsf{ID}}{\ell - 1} \rfloor$ and $\beta = \mathsf{ID} \bmod (\ell - 1)$.
(2) Check if msk_{α} is in MSK.
(3) If no, run $(mpk_{\alpha}, msk_{\alpha}) \leftarrow \mathsf{IPFE.Setup}(1^{\lambda}, 1^{\ell})$, add mpk_{α} to MPK and msk_{α} to MSK.
(4) Check if the vector $\boldsymbol{x}_{\beta}$ is in $\mathbb{L}_{\alpha}$.
(5) If yes, run $sk_{\boldsymbol{x}_{\beta}} \leftarrow \mathsf{IPFE.KeyGen}(msk_{\alpha}, \boldsymbol{x}_{\beta})$, set $sk_{\boldsymbol{x}_{\beta}} = (sk_{\boldsymbol{x}_{\beta}}, \alpha)$ and return $sk_{\boldsymbol{x}_{\beta}}$.
(6) Otherwise:
  6.1. Pick a random $\boldsymbol{x}_{\beta} \in (\mathbb{Z}_q^*)^{\ell}$ such that the set of vectors $\{\boldsymbol{x}\}_{\boldsymbol{x} \in \mathbb{L}_{\alpha}} \cup \{\boldsymbol{x}_{\beta}\}$ is linearly independent and add $\boldsymbol{x}_{\beta}$ to $\mathbb{L}_{\alpha}$.
  6.2. Run $sk_{\boldsymbol{x}_{\beta}} \leftarrow \mathsf{IPFE.KeyGen}(msk_{\alpha}, \boldsymbol{x}_{\beta})$, set $sk_{\boldsymbol{x}_{\beta}} = (sk_{\boldsymbol{x}_{\beta}}, \alpha)$ and return $sk_{\boldsymbol{x}_{\beta}}$.

Note that the clusters are created by KeyGen. A cluster consists of a master key pair (msk, mpk) and a set of $\ell - 1$ linearly independent vectors. A given ID in $\mathbb{Z}$ is mapped to a tuple $(\alpha, \boldsymbol{x})$ where $\boldsymbol{x} \in (\mathbb{Z}_q^*)^{\ell}$ is a vector of the cluster numbered α.

$\mathsf{Encrypt}(params, \mathsf{ID}, m)$. Given $\mathsf{ID} \in \mathbb{Z}$ and $m \in \mathbb{G}$, this algorithm performs the following steps:

(1) Compute $\alpha = \lfloor \frac{\mathsf{ID}}{\ell - 1} \rfloor$ and $\beta = \mathsf{ID} \bmod (\ell - 1)$.
(2) Check if $\boldsymbol{x}_{\beta} = (x_1, \ldots, x_{\ell})$ is in $\mathbb{L}_{\alpha}$.
(3) If no then abort. We require that encryption is only possible for identities for which a secret key is already produced.
(4) Otherwise: (mpk_{α} necessarily exists in MPK)
  4.1. Choose a random vector $\boldsymbol{r} = (r_1, \ldots, r_{\ell}) \in (\mathbb{Z}_q^*)^{\ell}$.
  4.2. Set $\boldsymbol{y} = \boldsymbol{x}_{\beta} \odot \boldsymbol{r} = (x_1 r_1, \ldots, x_{\ell} r_{\ell})$.
  4.3. Compute $c_1 = m g^{\langle \boldsymbol{x}_{\beta}, \boldsymbol{y} \rangle}$.
  4.4. Compute $c_2 \leftarrow \mathsf{IPFE.Encrypt}(mpk_{\alpha}, \boldsymbol{y})$.
  4.5. Return the ciphertext $C = (c_1, c_2)$.

$\mathsf{Decrypt}(params, sk_{\boldsymbol{x}}, C)$. Given a ciphertext $C = (c_1, c_2)$ and a secret key $sk_{\boldsymbol{x}}$ (containing α), this algorithm performs the following steps:

(1) Compute $\sigma = \mathsf{IPFE.Decrypt}(mpk_\alpha, sk_{\boldsymbol{x}}, c_2)$.
(2) Return the plaintext $m = c_1/g^\sigma$.

Note that the size of the master key pair $(\mathsf{MPK}, \mathsf{MSK})$ increases linearly with the number of clusters, but instantiation using a DDH-based IPFE (such as [22, 39]) can give short, constant-size secret keys.

The encryption algorithm fails if we try to encrypt an identity that does not yet have a decryption key. This limitation does not make our proposal impractical. It can be used in web-based email services where users obtain their decryption key when creating their account. It can also be deployed in IoT systems where devices obtain their decryption key when registering with the IoT infrastructure. In these cases, it is guaranteed that the recipient for whom we are encrypting has a decryption key. Therefore, the encryption algorithm will not abort.

The key generation algorithm is responsible for creating clusters and maintaining the list of vectors in each cluster. Therefore, KeyGen is stateful. This is not a new concept. IPFE schemes in [22, 39], which compute inner products in $\mathbb{Z}_q$, must use stateful key generation to be functional. This design choice raises several issues, including: secure storage of master secret keys, search efficiency for KeyGen and Encrypt, and communication overhead. Nonetheless, a good choice of cluster size can mitigate these problems because the larger the clusters, the fewer there are.

5.2.1 Security.

We state the IND-CPA security of our CIBE scheme by the following theorem with an overview of its proof.

Theorem 3. *If the underlying IPFE scheme is* IND-CPA *secure, then our clustered IBE scheme is also* IND-CPA *secure.*

Proof overview. This theorem can be proved by reduction. One can convert an instance of our CIBE scheme into an instance of the SIS-IBE scheme and prove that an adversary attacking our CIBE scheme has the same advantage as an adversary attacking our SIS-IBE scheme. Therefore, with Theorem 2, one can conclude with the statement of Theorem 3. □

By using a clustering technique, we converted our SIS-IBE scheme into a scalable and much more efficient IBE scheme. For instance, a SIS-IBE scheme that can produce 10000 secret keys ($\ell = 10001$) can be converted into a clustered IBE consisting of 477 clusters each producing 21 secret keys ($\ell = 22$). In this context, the sizes of the master keys of the two schemes (SIS-IBE and CIBE instantiated from the DDH assumption) are roughly the same. However, the clustered IBE is scalable and provides a better user experience because it enjoys a smaller secret key storage cost and its encryption and decryption processes are much more efficient (Figure 1.c and Figure 1.d give a glimpse of how the running times for the encryption and decryption algorithms scale with ℓ).

6 Conclusion

We have used IPFE as building blocks to construct a generic secure AND-gate CP-ABE scheme. A practical implementation of our CP-ABE scheme from the DDH assumption is also provided. A theoretical and experimental evaluation of our proposal has been done. From our generic CP-ABE construction, we derived a *clustered* IBE scheme. Although the latter scheme may suffer from large master keys (as the number of clusters increases), on the user side, a short and constant-size secret key can be appreciated. Additionally, an appropriate cluster size provides acceptable execution time for the decryption algorithm. Our constructions are non-black-box; therefore, they do not contradict the impossibility conjecture, but they prove that it is possible to construct practical and secure IBE and ABE schemes based on the DDH assumption.

Acknowledgments

We thank the anonymous reviewers for their constructive comments and valuable suggestions.

References

[1] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, page 89–98, 2006.

[2] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, pages 457–473, 2005.

[3] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy – SP 2007*, pages 321–334, 2007. .

[4] Javier Herranz. Attribute-based encryption implies identity-based encryption. *IET Information Security*, pages 332–337, 2017. .

[5] Javier Herranz. Attacking pairing-free attribute-based encryption schemes. *IEEE Access*, pages 222226–222232, 2020. .

[6] Syh-Yuan Tan, Kin-Woon Yeow, and Seong Oun

Hwang. Enhancement of a lightweight attribute-based encryption scheme for the internet of things. *IEEE Internet of Things Journal*, pages 6384–6395, 2019. .

[7] Shashank Agrawal and Melissa Chase. Fame: Fast attribute-based message encryption. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, page 665–682, 2017. .

[8] Doreen Riepel and Hoeteck Wee. Fabeo: Fast attribute-based encryption with optimal security. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, page 2491–2504, 2022. .

[9] Ahmad Khoureich Ka. Easy-ABE: An easy ciphertext-policy attribute-based encryption. In Giampaolo Bella, Mihai Doinea, and Helge Janicke, editors, *Innovative Security Solutions for Information Technology and Communications*, pages 168–183, 2023.

[10] Wei Dai, Yarkın Doröz, Yuriy Polyakov, Kurt Rohloff, Hadi Sajjadpour, Erkay Savaş, and Berk Sunar. Implementation and evaluation of a lattice-based key-policy ABE scheme. *IEEE Transactions on Information Forensics and Security*, pages 1169–1184, 2018. .

[11] Geng Wang, Ming Wan, Zhen Liu, and Dawu Gu. Fully secure lattice-based ABE from noisy linear functional encryption. In Yu Yu and Moti Yung, editors, *Information Security and Cryptology*, pages 421–441, 2021.

[12] Boxue Huang, Juntao Gao, and Xuelian Li. Efficient lattice-based revocable attribute-based encryption against decryption key exposure for cloud file sharing. *Journal of Cloud Computing*, page 37, 2023. .

[13] Vanga Odelu and Ashok Kumar Das. Design of a new CP-ABE with constant-size secret keys for lightweight devices using elliptic curve cryptography. *Sec. and Commun. Netw.*, page 4048–4059, 2016. .

[14] Vanga Odelu, Ashok Kumar Das, Muhammad Khurram Khan, Kim-Kwang Raymond Choo, and Minho Jo. Expressive CP-ABE scheme for mobile devices in IoT satisfying constant-size keys and ciphertexts. *IEEE Access*, pages 3273–3283, 2017. .

[15] Xuanxia Yao, Zhi Chen, and Ye Tian. A lightweight attribute-based encryption scheme for the internet of things. *Future Generation Computer Systems*, pages 104–112, 2015. .

[16] Dan Boneh, Periklis Papakonstantinou, Charles Rackoff, Yevgeniy Vahlis, and Brent Waters. On the impossibility of basing identity based encryption on trapdoor permutations. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 283–292, 2008. .

[17] Periklis A. Papakonstantinou, Charles W. Rackoff, and Yevgeniy Vahlis. How powerful are the DDH hard groups? Cryptology ePrint Archive, Paper 2012/653, 2012.

[18] Shuichi Katsumata and Shota Yamada. Non-zero inner product encryption schemes from various assumptions: LWE, DDH and DCR. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography – PKC 2019*, pages 158–188, 2019.

[19] Zhedong Wang, Xiong Fan, and Feng-Hao Liu. FE for inner products and its application to decentralized ABE. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography – PKC 2019*, pages 97–127, 2019.

[20] Ahmad Khoureich Ka. M-Sel: A message selection functional encryption from simple tools. In Mark Manulis, Diana MaimuŢ, and George Teşeleanu, editors, *Innovative Security Solutions for Information Technology and Communications*, pages 79–96, 2024.

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

[22] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016*, pages 333–362, 2016.

[23] Sheng Ding, Chen Li, and Hui Li. A novel efficient pairing-free CP-ABE based on elliptic curve cryptography for IoT. *IEEE Access*, pages 27336–27345, 2018. .

[24] Yong Wang, Biwen Chen, Lei Li, Qiang Ma, Huicong Li, and Debiao He. Efficient and secure ciphertext-policy attribute-based encryption without pairing for cloud-assisted smart grid. *IEEE Access*, pages 40704–40713, 2020. .

[25] Yi-Fan Tseng and Jheng-Jia Huang. Cryptanalysis on two pairing-free ciphertext-policy attribute-based encryption schemes. In *2020 International Computer Symposium (ICS)*, pages 403–407, 2020. .

[26] Hamednejad Farnoosh, Mohajeri Javad, and Mohammad Reza A. Attacking two pairing-free ciphertext-policy attribute-based encryption schemes. *The ISC International Journal of Information Security (ISeCure)*, page 151–160, 2025. .

[27] Sangjukta Das and Suyel Namasudra. Multiauthority cp-abe-based access control model for iot-enabled healthcare infrastructure. *IEEE Transactions on Industrial Informatics*, pages 821–829,

2023. .

[28] K. Sowjanya and Mou Dasgupta. A ciphertext-policy attribute based encryption scheme for wireless body area networks based on ecc. *Journal of Information Security and Applications*, 54: 102559, 2020. ISSN 2214-2126. .

[29] Nico Döttling and Sanjam Garg. Identity-based encryption from the diffie-hellman assumption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 537–569, 2017.

[30] Olivier Blazy and Saqib A. Kakvi. Identity-based encryption in DDH hard groups. In Lejla Batina and Joan Daemen, editors, *Progress in Cryptology – AFRICACRYPT 2022*, pages 81–102, 2022.

[31] Xuanxia Yao, Jinyuan Zhou, Xiaojiang Du, and Shurong Zhang. A cp-abe and iota-based lightweight sensitive data access control scheme for iot. *IEEE Internet of Things Journal*, pages 40831–40844, 2024. .

[32] Miao Wang, Zhenfu Cao, Xiaolei Dong, Runmeng Du, and Jiasheng Chen. Decentralized multiauthority kp-abe scheme without bilinear pairings. *IEEE Internet of Things Journal*, pages 726–738, 2025. .

[33] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *Public-Key Cryptography – PKC 2015*, pages 733–751, 2015.

[34] Adi Shamir. Identity-based cryptosystems and signature schemes. In George Robert Blakley and David Chaum, editors, *Advances in Cryptology*, pages 47–53, 1985.

[35] Lucas Kowalczyk and Hoeteck Wee. Compact adaptively secure ABE for NC^1 from k-lin. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019*, pages 3–33, 2019.

[36] Joseph A. Akinyele, Christina Garman, Ian Miers, Matthew W. Pagano, Michael Rushanan, Matthew Green, and Aviel D. Rubin. Charm: a framework for rapidly prototyping cryptosystems. *Journal of Cryptographic Engineering*, pages 111–128, 2013. .

[37] Ahmad Khoureich Ka. EasyABE github, 2023. URL https://github.com/khoureich/EasyABE.

[38] Ahmad Khoureich Ka. CIBE github, 2024. URL https://github.com/khoureich/Clustered-IBE.

[39] Guilhem Castagnos, Fabien Laguillaumie, and Ida Tucker. Practical fully secure unrestricted inner product functional encryption modulo p. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018*, pages 733–764, 2018.

[40] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, pages 213–229, 2001.

A Syntax of IPFE and its security definition

In the case where the key space is $\mathbb{Z}_q^\ell$, the IPFE scheme is stateful since it must maintain the list of previously key queries as internal state to prevent an adversary from having ℓ independent secret keys, allowing to uncover the master secret key [22].

A stateful IPFE scheme computing inner products in $\mathbb{Z}_q$ consists of four polynomial-time algorithms:

(1) $\mathsf{Setup}(1^\lambda, 1^\ell)$. Input: a security parameter λ and a functionality parameter ℓ. Output: a master public key mpk and a master secret key msk.

(2) $\mathsf{KeyGen}(msk, st, \boldsymbol{x})$. Input: the master secret key msk, an internal state st and a vector $\boldsymbol{x} \in \mathbb{Z}_q^\ell$. Output: a secret key $sk_{\boldsymbol{x}}$.

(3) $\mathsf{Encrypt}(mpk, \boldsymbol{y})$. Input: the master public key mpk and a vector $\boldsymbol{y} \in \mathbb{Z}_q^\ell$. Output: a ciphertext $C_{\boldsymbol{y}}$.

(4) $\mathsf{Decrypt}(mpk, sk_{\boldsymbol{x}}, C_{\boldsymbol{y}})$. Input: the master public key mpk, a secret key $sk_{\boldsymbol{x}}$ and a ciphertext $c_{\boldsymbol{y}}$. Output: the inner product $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}_q$.

For correctness, it is required that for all $\boldsymbol{x} \in \mathbb{Z}_q^\ell$ and all $\boldsymbol{y} \in \mathbb{Z}_q^\ell$, we have $\mathsf{Decrypt}(mpk, sk_{\boldsymbol{x}}, \mathsf{Encrypt}(mpk, \boldsymbol{y})) = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$ or $\perp$ with negligible probability.

A.1 IND-CPA security

Indistinguishability under Chosen Plaintext Attack security is defined via the following game:

(1) Setup: the challenger $\mathcal{C}$ runs $\mathsf{Setup}(1^\lambda, 1^\ell)$, obtains (msk, mpk) and gives mpk to the adversary $\mathcal{A}$.

(2) $\mathsf{Key\ query\ 1}$: the adversary $\mathcal{A}$ requests from $\mathcal{C}$ the secret keys associated to vectors $\{\boldsymbol{x}_i\}_{i=1}^q$, $q \in poly(\lambda)$ of its choice. These queries can be made adaptively. For each $\boldsymbol{x}_i$, $\mathcal{C}$ runs $\mathsf{KeyGen}(msk, st, \boldsymbol{x}_i)$ and gives the result to $\mathcal{A}$.

(3) $\mathsf{Challenge}$: $\mathcal{A}$ outputs two messages $\boldsymbol{y}_0$ and $\boldsymbol{y}_1$ of the same length subject to the restriction that $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle$ for all $\boldsymbol{x}$ queried during the phase $\mathsf{Key\ query\ 1}$. $\mathcal{C}$ randomly selects $b \in \{0, 1\}$, runs $\mathsf{Encrypt}(mpk, \boldsymbol{y}_b)$ and sends the result to $\mathcal{A}$.

(4) Key query 2: key query 1 is repeated with the restriction that $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle$ for each query $\boldsymbol{x}$.

(5) Guess: $\mathcal{A}$ outputs a guess b' of b and wins if $b' = b$.

The advantage of an adversary $\mathcal{A}$ in this IND-CPA game is defined as

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{IPFE},\mathcal{A}}(\lambda) = \mathsf{Pr}[b' = b] - \frac{1}{2}$$

Definition 1. An IPFE scheme is adaptively IND-CPA secure if for any polynomial-time adversary $\mathcal{A}$, the function $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{IPFE},\mathcal{A}}(\cdot)$ is negligible.

B Syntax of CP-ABE and its security definition

A ciphertext-policy ABE scheme consists of four polynomial-time algorithms:

(1) $\mathsf{Setup}(1^\lambda)$. Input: the security parameter λ. Output: a master public key mpk and a master secret key msk.

(2) $\mathsf{KeyGen}(msk, \mathbb{S})$. Input: the master secret key msk and a set of attributes $\mathbb{S}$. Output: a secret key sk.

(3) $\mathsf{Encrypt}(mpk, \mathbb{P}, m)$. Input: the master public key mpk, an access policy $\mathbb{P}$ and a message m. Output: a ciphertext C.

(4) $\mathsf{Decrypt}(mpk, C, sk)$. Input: the master public key mpk, a ciphertext C and a secret key sk. Output: the plaintext m or $\perp$ meaning that C is malformed or sk does not satisfy the access policy $\mathbb{P}$.

B.1 IND-CPA security for secret policy CP-ABE

We define indistinguishability under chosen plaintext attack security for a secret policy CP-ABE scheme via the following game. The challenger encrypts messages provided by the adversary $\mathcal{A}$ using an access policy $\mathbb{P}$ unknown to $\mathcal{A}$.

(1) Setup: the challenger $\mathcal{C}$ runs $\mathsf{Setup}(1^\lambda)$, obtains (msk, mpk) and gives mpk to the adversary.

(2) Key query 1: $\mathcal{A}$ requests from $\mathcal{C}$ the secret keys associated to attribute sets $\{\mathbb{S}_i\}_{i=1}^q$, $q \in poly(\lambda)$ of its choice. These requests can be made adaptively. For each $\mathbb{S}_i$, $\mathcal{C}$ runs $\mathsf{KeyGen}(msk, \mathbb{S}_i)$ and gives the result to $\mathcal{A}$.

(3) Challenge: $\mathcal{A}$ outputs two distinct messages m_0 and m_1 of the same length. The challenger selects an access policy $\mathbb{P}$ and chooses uniformly a bit $b \in \{0, 1\}$, runs $\mathsf{Encrypt}(mpk, \mathbb{P}, m_b)$ and sends the result to $\mathcal{A}$.

(4) Key query 2: key query 1 is repeated.

(5) Guess: $\mathcal{A}$ outputs a guess b' of b and wins if $b' = b$.

The advantage of an adversary $\mathcal{A}$ in this IND-CPA game is defined as

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{CPABE},\mathcal{A}}(\lambda) = \mathsf{Pr}[b' = b] - \frac{1}{2}$$

Definition 2. A CP-ABE scheme is adaptively IND-CPA secure if for any polynomial-time adversary $\mathcal{A}$, the function $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{CPABE},\mathcal{A}}(\cdot)$ is negligible.

C Syntax of IBE and its security definition

An identity-based encryption scheme consists of four polynomial-time algorithms:

(1) $\mathsf{Setup}(1^\lambda)$. Input: the security parameter λ. Output: a master secret key msk and system parameters denoted by $params$.

(2) $\mathsf{KeyGen}(params, msk, \mathsf{ID})$. Input: the system parameters $params$, the master secret key msk and an identity $\mathsf{ID} \in \mathbb{Z}_q$. Output: a secret key sk.

(3) $\mathsf{Encrypt}(params, \mathsf{ID}, m)$. Input: the system parameters $params$, an identity $\mathsf{ID} \in \mathbb{Z}_q$, and a message m. Output: a ciphertext C.

(4) $\mathsf{Decrypt}(params, sk_{\mathsf{ID}}, C)$. Input: the system parameters $params$, a secret key sk_{ID} and a ciphertext C. Output: the plaintext m or $\perp$ meaning that C is malformed or is not obtained using ID.

C.1 IND-CPA security

Boneh et *al.* [40] defined IND-CPA security for IBE via the following security game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:

(1) Setup: $\mathcal{C}$ runs $\mathsf{Setup}(1^\lambda)$, obtains $(params, msk)$ and gives $params$ to $\mathcal{A}$.

(2) Key query 1: Adversary $\mathcal{A}$ requests from $\mathcal{C}$ the secret keys associated to identities $\{\mathsf{ID}_i\}_{i=1}^q$, $q \in poly(\lambda)$ of its choice. These requests can be made adaptively. For each ID_i, $\mathcal{C}$ runs $\mathsf{KeyGen}(params, msk, \mathsf{ID}_i)$ and gives the result to $\mathcal{A}$.

(3) Challenge: $\mathcal{A}$ outputs two distinct messages m_0, m_1 of the same length and an identity $\mathsf{ID}^\star$ subject to the restriction that no secret key has been previously generated for it. $\mathcal{C}$ randomly se-

lects $b \in \{0,1\}$, runs $\mathsf{Encrypt}(params, \mathsf{ID}^\star, m_b)$ and sends the result to $\mathcal{A}$.

(4) Key query 2: key query 1 is repeated with the restriction that $\mathsf{ID}_i \neq \mathsf{ID}^\star$.

(5) Guess: $\mathcal{A}$ outputs a guess b' of b and wins if $b' = b$.

The advantage of an adversary $\mathcal{A}$ in this IND-CPA game is defined as

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{IBE},\mathcal{A}}(\lambda) = \mathsf{Pr}[b' = b] - \frac{1}{2}$$

Definition 3. An IBE scheme is adaptively IND-CPA secure if for any polynomial-time adversary $\mathcal{A}$, the function $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{IBE},\mathcal{A}}(\cdot)$ is negligible.

D Proof of Theorem 1

Assume there exists an IND-CPA adversary $\mathcal{A}$ that has a non-negligible advantage against our CP-ABE scheme. We show below how an adversary $\mathcal{B}$ would interact with $\mathcal{A}$ in five phases to break the IND-CPA security of the underlying IPFE scheme. Let $\mathcal{C}$ be the challenger of $\mathcal{B}$.

(1) Setup: $\mathcal{C}$ runs $(msk, mpk) \leftarrow \mathsf{IPFE.Setup}(1^\lambda, 1^\ell)$ and gives mpk to $\mathcal{B}$. The latter chooses a cyclic group $\mathbb{G}$ of prime order $q > 2^\lambda$ with generator g and gives to $\mathcal{A}$ the tuple $(\mathbb{G}, g, mpk)$.

(2) Key query 1: $\mathcal{A}$ makes repeated secret key queries to $\mathcal{B}$. When $\mathcal{B}$ receives one of these requests associated with a set of attributes $\mathbb{S} \in (\mathbb{Z}_q^* \times [\ell])^n$ where $n \leq \ell$, it does the following:
- converts $\mathbb{S}$ to a vector $\boldsymbol{S} = (s_1, \ldots, s_\ell) \in \mathbb{Z}_q^\ell$.
- sends to $\mathcal{C}$ a secret key request for $\boldsymbol{S}$ to obtain $sk_{\boldsymbol{S}}$. By the way, $\mathcal{C}$ runs $sk_{\boldsymbol{S}} \leftarrow \mathsf{IPFE.KeyGen}(msk, st, \boldsymbol{S})$ to respond to this query.
- returns $sk_{\boldsymbol{S}}$ to $\mathcal{A}$.

(3) Challenge: $\mathcal{A}$ outputs two distinct messages m_0, $m_1 \in \mathbb{G}$ of the same length. $\mathcal{B}$ chooses an access policy $\mathbb{P} \in (\mathbb{Z}_q^* \times [\ell])^n$ where $n \leq \ell$, converts it to a vector $\boldsymbol{P} \in \mathbb{Z}_q^\ell$. $\mathcal{B}$ also chooses two random vectors $\boldsymbol{r}_0, \boldsymbol{r}_1 \in (\mathbb{Z}_q^*)^\ell$, computes $\boldsymbol{y}_0 = \boldsymbol{P} \odot \boldsymbol{r}_0$, $\boldsymbol{y}_1 = \boldsymbol{P} \odot \boldsymbol{r}_1$ and sends $\boldsymbol{y}_0, \boldsymbol{y}_1$ as challenge messages to $\mathcal{C}$. The latter randomly selects $b \in \{0,1\}$, computes $c_2 \leftarrow \mathsf{IPFE.Encrypt}(mpk, \boldsymbol{y}_b)$ and sends the result to $\mathcal{B}$. Upon receiving c_2, $\mathcal{B}$ chooses $b' \in \{0,1\}$, computes $c_1 = m_{b'} g^{\langle \boldsymbol{P}, \boldsymbol{y}_{b'} \rangle}$ and returns (c_1, c_2) to $\mathcal{A}$ as the challenge ciphertext.

(4) Key query 2: Similar to Key query 1.

(5) Guess: $\mathcal{A}$ outputs a guess b^*. $\mathcal{B}$ outputs the same guess as $\mathcal{A}$.

When $b = b'$, adversary $\mathcal{B}$ simulates perfectly for $\mathcal{A}$ the challenger during a chosen plaintext attack against our CP-ABE scheme. This event happens with probability $1/2$ since b is independent of b'. When $b \neq b'$ $\mathcal{B}$ produces an invalid ciphertext and $\mathcal{A}$ can do nothing but guess which of the two messages was encrypted. Therefore, we have

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{CPABE},\mathcal{A}} = 2 \cdot \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{IPFE},\mathcal{B}}$$

Since the IPFE scheme is IND-CPA secure, $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{IPFE},\mathcal{B}}$ is negligible. Thus, the IND-CPA advantage of $\mathcal{A}$ against our CP-ABE scheme is negligible. This concludes the proof.

E IPFE-DDH* : the modified version of the DDH based IPFE scheme of Agrawal *et al.*

The original version of the DDH-based IPFE scheme is available in [22]. The modification only concerns the decryption algorithm. It returns $g_1^{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$ instead of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$, thus avoiding the computation of the discrete logarithm, which is too expensive.

$\mathsf{Setup}(1^\lambda, 1^\ell)$

(1) Choose a cyclic group $\mathbb{G}$ of prime order $q > 2^\lambda$ with generators g_1, g_2.
(2) Choose 2 vectors $\boldsymbol{u} = (u_1, \ldots, u_\ell) \xleftarrow{R} \mathbb{Z}_q^\ell$, $\boldsymbol{v} = (v_1, \ldots, v_\ell) \xleftarrow{R} \mathbb{Z}_q^\ell$.
(3) For each $i \in [\ell]$ compute $h_i = g_1^{u_i} \cdot g_2^{v_i}$.
(4) Return $msk = (\boldsymbol{u}, \boldsymbol{v})$, $mpk = (\mathbb{G}, g_1, g_2, \{h_i\}_{i=1}^\ell)$.

$\mathsf{KeyGen}(msk, st, \boldsymbol{x} \in \mathbb{Z}_q^\ell)$. The internal state st contains at most ℓ tuples of the form $(\boldsymbol{z}_i, \boldsymbol{z}'_i, u_{\boldsymbol{z}_i}, v_{\boldsymbol{z}_i})$ where $(\boldsymbol{z}'_i, u_{\boldsymbol{z}_i}, v_{\boldsymbol{z}_i})$ are previously generated secret keys for $\boldsymbol{z}_i \in \mathbb{Z}_q^\ell$. To generate the j^{th} secret key, this algorithm does the following:

- checks if $\boldsymbol{x}$ is linearly independent of the $\boldsymbol{z}_i$'s:
  If $\nexists \{\gamma_i\}_{i=1}^{j-1} \in \mathbb{Z}^{j-1}$ such that $\boldsymbol{x} = \sum_{i=1}^{j-1} \gamma_i \boldsymbol{z}_i \mod q$ then
    set $\boldsymbol{x}' = \boldsymbol{x}$, $u_{\boldsymbol{x}} = \langle \boldsymbol{u}, \boldsymbol{x} \rangle$, $v_{\boldsymbol{x}} = \langle \boldsymbol{v}, \boldsymbol{x} \rangle$ and add $(\boldsymbol{x}, \boldsymbol{x}', u_{\boldsymbol{x}}, v_{\boldsymbol{x}})$ to st.
  Else set
    $\boldsymbol{x}' = \sum_{i=1}^{j-1} \gamma_i \boldsymbol{z}'_i \in \mathbb{Z}^\ell$, $u_{\boldsymbol{x}} = \sum_{i=1}^{j-1} \gamma_i u_{\boldsymbol{z}_i} \in \mathbb{Z}$ and $v_{\boldsymbol{x}} = \sum_{i=1}^{j-1} \gamma_i v_{\boldsymbol{z}_i} \in \mathbb{Z}$.
- returns $sk_{\boldsymbol{x}} = (\boldsymbol{x}', u_{\boldsymbol{x}}, v_{\boldsymbol{x}})$.

$\mathsf{Encrypt}(mpk, \boldsymbol{y} = (y_1, \ldots, y_\ell) \in \mathbb{Z}_q^\ell)$

(1) Pick $r \xleftarrow{R} \mathbb{Z}_q^*$.
(2) Compute $C = g_1^r, D = g_2^r, \{E_i = g_1^{y_i} \cdot h_i^r\}_{i=1}^\ell$.
(3) Return $C_{\boldsymbol{y}} = (C, D, E_1, \ldots, E_\ell)$.

$\mathsf{Decrypt}(mpk, sk_{\boldsymbol{x}} = (\boldsymbol{x}', u_{\boldsymbol{x}}, v_{\boldsymbol{x}}), C_{\boldsymbol{y}})$

$\boldsymbol{x}' = (x'_1, \ldots, x'_\ell) \in \mathbb{Z}_q^\ell$

(1) Return $E_{\boldsymbol{x}} = (\prod_{i=1}^{\ell} E_i^{x'_i})/(C^{u_{\boldsymbol{x}}} \cdot D^{v_{\boldsymbol{x}}}) = g_1^{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$.

F Proof of Theorem 2

Assume there exists an IND-CPA adversary $\mathcal{A}$ that has a non-negligible advantage against our SIS-IBE scheme. We show below how an adversary $\mathcal{B}$ would interact with $\mathcal{A}$ in five phases to break the IND-CPA security of the underlying IPFE scheme. Let $\mathcal{C}$ be the challenger of $\mathcal{B}$.

(1) Setup: $\mathcal{C}$ runs $(msk, mpk) \leftarrow \mathsf{IPFE.Setup}(1^\lambda, 1^\ell)$ and gives mpk to $\mathcal{B}$. The latter chooses a cyclic group $\mathbb{G}$ of prime order $q > 2^\lambda$ with generator g, creates a list $\mathbb{L}$ (initially empty) of vectors in $(\mathbb{Z}_q^*)^\ell$ that has a maximum capacity of $\ell - 1$ vectors, sets $params = (mpk, \mathbb{G}, g, \mathbb{L})$ and gives $params$ to $\mathcal{A}$.

(2) Key query 1: $\mathcal{A}$ makes repeated secret key queries associated to identities $\{\mathsf{ID}_i\}_{i=1}^q$, $q \leq \ell - 2$ of its choice. Assume $\mathcal{A}$ does not make different queries for the same ID. When $\mathcal{B}$ receives one of these requests associated to an identity ID, it does the following:
- picks a random $\boldsymbol{x}_{\mathsf{ID}} \in (\mathbb{Z}_q^*)^\ell$ such that the set of vectors $\{\boldsymbol{x}\}_{\boldsymbol{x} \in \mathbb{L}} \cup \{\boldsymbol{x}_{\mathsf{ID}}\}$ is linearly independent, and adds $\boldsymbol{x}_{\mathsf{ID}}$ to $\mathbb{L}$ (at any time $\mathcal{A}$ gets an update on $\mathbb{L}$).
- sends to $\mathcal{C}$ a secret key request for $\boldsymbol{x}_{\mathsf{ID}}$ and obtains $sk_{\boldsymbol{x}_{\mathsf{ID}}}$. By the way, $\mathcal{C}$ runs $sk_{\boldsymbol{x}_{\mathsf{ID}}} \leftarrow \mathsf{IPFE.KeyGen}(msk, \boldsymbol{x}_{\mathsf{ID}})$ to respond to this query.
- returns $sk_{\boldsymbol{x}_{\mathsf{ID}}}$ to $\mathcal{A}$.

(3) Challenge: $\mathcal{A}$ outputs two distinct messages $m_0, m_1 \in \mathbb{G}$ of the same length and an identity $\mathsf{ID}^\star$ that has not been associated to a query in the previous phase. $\mathcal{B}$ picks a random $\boldsymbol{x}_{\mathsf{ID}^\star} \in (\mathbb{Z}_q^*)^\ell$ such that the set of vectors $\{\boldsymbol{x}\}_{\boldsymbol{x} \in \mathbb{L}} \cup \{\boldsymbol{x}_{\mathsf{ID}^\star}\}$ is linearly independent, chooses two random vectors $\boldsymbol{r}_0, \boldsymbol{r}_1 \in (\mathbb{Z}_q^*)^\ell$, computes $\boldsymbol{y}_0 = \boldsymbol{x}_{\mathsf{ID}^\star} \odot \boldsymbol{r}_0$, $\boldsymbol{y}_1 = \boldsymbol{x}_{\mathsf{ID}^\star} \odot \boldsymbol{r}_1$ and sends $\boldsymbol{y}_0, \boldsymbol{y}_1$ as challenge messages to $\mathcal{C}$. The latter randomly selects $b \in \{0, 1\}$, computes $c_2 \leftarrow \mathsf{IPFE.Encrypt}(mpk, \boldsymbol{y}_b)$ and sends the result to $\mathcal{B}$. Upon receiving c_2, $\mathcal{B}$ chooses $b' \in \{0, 1\}$, computes $c_1 = m_{b'} g^{\langle \boldsymbol{x}_{\mathsf{ID}^\star}, \boldsymbol{y}_{b'} \rangle}$ and returns (c_1, c_2) to $\mathcal{A}$ as the challenge ciphertext.

(4) Key query 2: Similar to Key query 1 with identities $\mathsf{ID}_i \neq \mathsf{ID}^\star$. In total, $\mathcal{A}$ makes at most $\ell - 2$ key queries.

(5) Guess: $\mathcal{A}$ outputs a guess b^*. $\mathcal{B}$ outputs the same guess as $\mathcal{A}$.

When $b = b'$, adversary $\mathcal{B}$ simulates perfectly for $\mathcal{A}$ the challenger during a chosen plaintext attack against our CP-ABE scheme. This event happens with probability $1/2$ since b is independent of b'. When $b \neq b'$ $\mathcal{B}$ produces an invalid ciphertext and $\mathcal{A}$ can do nothing but guess which of the two messages was encrypted. Therefore, we have

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{SIS\text{-}IBE},\mathcal{A}} = 2 \cdot \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{IPFE},\mathcal{B}}$$

Since the IPFE scheme is IND-CPA secure, $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{IPFE},\mathcal{B}}$ is negligible. Thus, the IND-CPA advantage of $\mathcal{A}$ against our SIS-IBE scheme is negligible. This concludes the proof.

Ahmad Khoureich Ka received the PhD degree in Signal Processing and Telecommunications from the Rennes Mathematical Research Institute (IRMAR), University of Rennes 1, France, in 2012. He is currently Maître de conférences at the Department of Information and Communication Technologies, Alioune Diop University of Bambey, Senegal. His research interests are related to applied cryptography and web security.