Ciphertext-Policy ABE from Inner-Product FE

Ahmad Khoureich Ka

Université Alioune Diop de Bambey, Senegal
ahmadkhoureich.ka@uadb.edu.sn

Abstract. The enormous 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 seems to reinforce the point of view of some authors 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 AND gate access structured Ciphertext-Policy ABE (CP-ABE) scheme with secret access policy from Inner-Product Functional Encryption (IPFE). We also propose an instantiation of that generic CP-ABE scheme from 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 indeed possible to construct practical and 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 [21].

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

1 Introduction

Attribute-Based Encryption (ABE) is an asymmetric encryption primitive that allows decryption of the ciphertext if and only if an access policy is satisfied [15,32,6]. The access policy considered as a predicate $P(\cdot)$ is defined on attributes which can be any element that can be used to identify a user. 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 to sets of attributes. In either cases decryption is possible only if $P(S) = 1$.

ABE has a wide range of applications including social applications [34], e-health systems [27], access control in cloud computing [41,26]. However, ABE has difficulty penetrating the world of resource-constrained IoT devices because lightweight schemes have the reputation of presenting crippling security flaws [16,17,35] and available secure ABE schemes are resource-intensive. ABE schemes proven secure are based on bilinear pairing [2,31,19] or lattices [11,37,18] and suffer either of large decryption key size or being slow for encryption or decryption. Suitable ABE schemes for lightweight devices [28,29,40,35] exploit classical security assumption like RSA or the Decisional Diffie-Hellman (DDH), hence their generic name of pairing-free ABE. The discovery of security vulnerabilities in many of those schemes is not surprising to some authors [9,30] since they conjectured (*the impossibility conjecture*) that designing Identity-Based Encryption (IBE) schemes (a particular version of ABE [16] introduced by Shamir in 1984 [33]) that rely on trapdoor permutations or the DDH assumption in a black box manner is impossible. Furthermore, Herranz [17] 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 various useful applications including the computation of weighted mean over encrypted data [1], the construction of trace-and-revoke systems [3], the construction of non-zero inner product encryption [23], the construction of decentralized ABE against bounded collusion [39] and the construction of message selection functional encryption allowing decryption of selected portions of a ciphertext [22].

Our construction is a secret access 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.* [24], 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). Therefore, our scheme is a secret access policy CP-ABE and has a large policy space. This makes collusion and exhaustive-search attacks infeasible.

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

By providing an instantiation of our CP-ABE scheme from the DDH-based IPFE of [4], we show that it is possible to construct practical and secure ABE and IBE schemes based on the DDH assumption (contradicting the impossibility conjecture [9,30,16]).

1.2 Organization

The rest of this paper is organized as follows. In section 2 we summarize related work. Section 3 presents the basic knowledge related to IPFE and CP-ABE. Section 4 presents the construction of our generic CP-ABE scheme from IPFE, its security analysis and an instantiation from the DDH assumption. A comparison with other schemes is also performed in this section. In section 5, we study a special case of our generic CP-ABE scheme and derive from it a generic IBE scheme with the introduction of the notion of clustered identity-based encryption. Section 6 concludes this work.

2 Related work.

The enormous potential of attribute-based encryption in the context of IoT has motivated researchers to propose pairing-free ABE schemes suitable for resource-constrained devices. Unfortunately, many of those schemes turned out to be insecure. CP-ABE schemes in [12,38] avoid bilinear pairing by using scalar multiplication on elliptic curves but they are found vulnerable against collusion attacks [36]. Odelu *et al.* have proposed two CP-ABE schemes [28,29] one based on RSA and the other using elliptic curve cryptography for encryption and decryption. Sadly, an explicit attack against these two schemes has been successful [16]. The KP-ABE scheme proposed in [40] has its proof of security based on the Elliptic Curve Decisional Diffie-Hellman (ECDDH) assumption instead of Bilinear Diffie–Hellman (BDH) assumption, but it seems that the security proof is incorrect since it has been shown in [35] that an adversary can decrypt a ciphertext which does not satisfy the access policy of his decryption key. In [35] an improvement of [40] is proposed but it has been shown in [17] that the proposal is vulnerable to a key-recovery attack. All these failures tend to reinforce the impossibility conjecture [9,30,17] that says that designing secure IBE schemes that rely solely on the hardness of the Diffie-Hellman Problem is impossible. However, attacks against the impossibility conjecture debuted with the work of Döttling and Garg [13] who proposed a construction of IBE under DDH using Garbled Circuits. More recently, Blazy and Kakvi [7] provided a construction of IBE similar to that of Döttling and Garg based on Witness Encryption [14] instead of garbled circuits. Unfortunately, these schemes are inefficient and therefore do not completely contradict the impossibility conjecture.

In section 4.2, we show that it is indeed possible to construct a secure and efficient CP-ABE scheme based on the classical DDH assumption.

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 formalize the notion of IPFE [1].

In this work, we only consider IPFE schemes which allow the computation of inner products in the ring $\mathcal{R} = \mathbb{Z}$ or $\mathcal{R} = \mathbb{Z}_q$ for some prime q. The syntax of IPFE and its INDistinguishability under Chosen Plaintext Attack (IND-CPA) security are given in Appendix A.

Various instantiations of IPFE from standard assumptions such as the Decisional Diffie-Hellman, the Learning-With-Errors (LWE), and the Decisional

Composite Residuosity (DCR) are available in the literature [1,4,10]. Our generic constructions can benefit from them.

3.2 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 (a single subset of the attribute universe). In this case, $\mathbb{S}$ satisfies $\mathbb{P}$ if and only if $\mathbb{S}$ contains $\mathbb{P}$. The syntax of CP-ABE and its indistinguishability-based security are given in Appendix B.

3.3 Identity Based Encryption

Identity Based Encryption was introduced by Shamir in 1984 [33]. IBE is an attractive Public Key Encryption (PKE) primitive since there is no need to tie a random public key to the owner's identity using a certificate management system (as in traditional PKE) because in IBE the user's identity is his or her public key. An IBE scheme [8,33] consists of a set-up algorithm which initializes all system parameters, a key generation algorithm which computes user's private key from a string representing his or her identity, an encryption and decryption algorithm.The syntax of IBE and its indistinguishability-based security are given in Appendix C.

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 that has at most a cardinality of 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 $\mathbb{S} = \{(a_i, b_i)\}_{i=1}^n \in (\mathbb{Z}_q^* \times [\ell])^n$ where $n \leq \ell$ 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 AND gate secret access policy CP-ABE scheme from stateful IPFE. The stateless version where the underlying IPFE computes inner products in $\mathbb{Z}$ is syntactically identical but do 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 $\mathbb{S} \supseteq \mathbb{P}$ meanning 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 secret the access policy and the set of attributes associated with a secret key is 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 make queries for secret keys associated to 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 the DDH-based IPFE scheme of [4] (that we denote IPFE-DDH) described in Appendix E. We consider a modification of the decryption algorithm of IPFE-DDH that returns $E_{\boldsymbol{x}} = g_1^{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$, thus avoiding the computation of the discrete logarithm $\log_{g_1}(E_{\boldsymbol{x}})$ that is too expensive. That modification is denoted $\mathsf{Decrypt}^{\star}$ to avoid confusion. 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 $\mathsf{IPFE\text{-}DDH.Setup}(1^{\lambda}, 1^{\ell})$ 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 $\rho = \mathsf{IPFE\text{-}DDH.Decrypt}^{\star}(mpk, sk_{\boldsymbol{S}}, c_2)$
2. Return the plaintext $m = c_1/\rho$.

4.3 Theoretical Evaluation

We do a theoretical comparison of our DDH-based CP-ABE scheme with other schemes such as FAME [2], FABEO [31] and Easy-ABE [19]. All these schemes are pairing based ABE schemes. This comparison is made only on AND gate access policies (the only one supported by our scheme). FAME and FABEO support access policies described by boolean formulas that are convertible into Monotone Span Programs (MSPs). An MSPs 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 we say that those schemes have one-use restriction. Techniques [25] to avoid the one-use restriction can be used but lead to a less efficient scheme [31]. Easy-ABE does not use MSPs (therefore does not suffer from the one-use restriction), it derives from the boolean formula consisting of AND/OR gates the collection of authorized sets of attributes and then converts each authorized set into a bit string using the universe of attributes. In this comparison (see Tables 1, 2, 3), we use the one-use restriction versions of FAME and FABEO.

Some may think that comparing our CP-ABE scheme against pairing based CP-ABE ones supporting more complex policies is unfair. Nonetheless, this comparison gives a glimpse of the behavior 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 and decryption. n_1, n_2 are the number of rows and columns of the MSPs in FAME and FABEO, I is the number of attributes used in decryption, 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				Decryption			
	$\mathbb{G}_1$			$\mathbb{G}_2$	$\mathbb{G}_1$		$\mathbb{G}_T$	
Scheme	Mul	Exp	Hash	Exp	Mul	Exp	Mul	Pairing
FAME	$12n_1n_2+6n_1$	$6n_1$	$6(n_1+n_2)$	3	$6I+3$	–	6	6
FABEO	n_1	$2n_1$	n_1+1	2	$2I$	–	3	3
Easy-ABE	1	$m+2$	m	1	–	–	1	2
Our	$2\ell+1$	$2\ell+3$	–	–	$\ell+2$	$\ell+2$	–	–

4.4 Experimental Evaluation

We implemented our DDH-based CP-ABE scheme using the Charm 0.5 framework [5]. The implementations of FAME [2], FABEO [31] and Easy-ABE [19] are taken from github [20]. 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 and our scheme uses the group $\mathbb{G}_1$ of MNT224 of the Charm framework.

Table 3. 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.

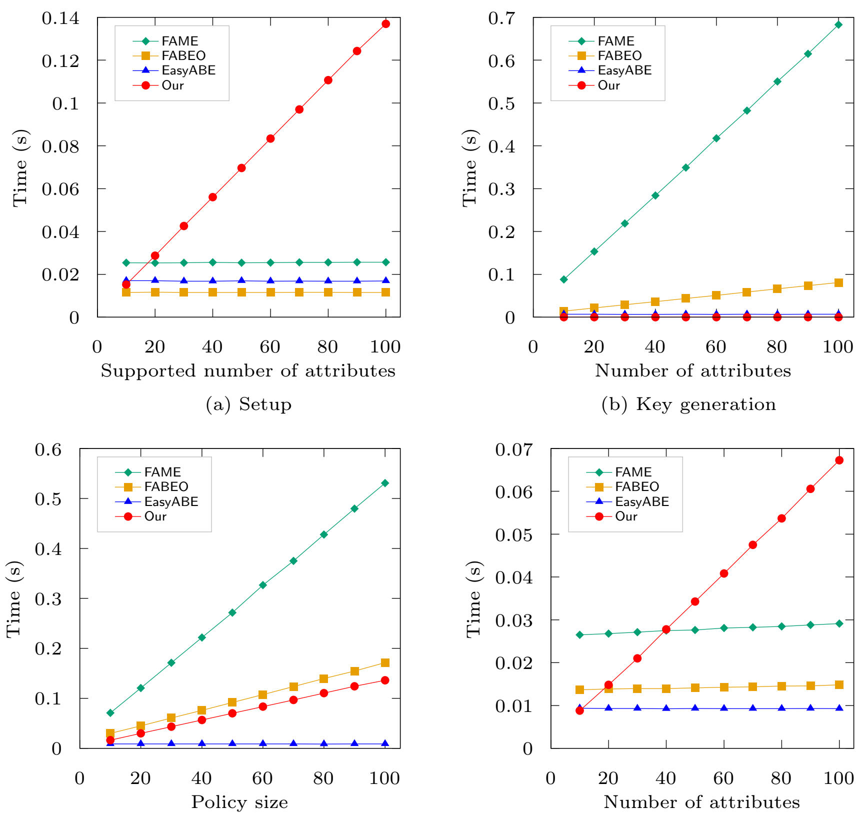
	Storage cost			
	Key size		Ciphertext size	
Scheme	$\mathbb{G}_1$ or $\mathbb{Z}_q$ for our	$\mathbb{G}_2$	$\mathbb{G}_1$	$\mathbb{G}_2$
FAME	$3(T+1)$	3	$3n_1$	3
FABEO	$T+1$	1	n_1	2
Easy-ABE	1	1	$m+1$	1
Our	$\ell+2$	–	$\ell+3$	–

The experiments are carried out for the four algorithms of 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 [2,31,19]. For our DDH-based CP-ABE scheme we use the same form of access policies but attributes are big integers randomly picked from the integer group ZR of the Charm framework. Indeed, the size of the numbers used as attributes has an effect on the execution times of our scheme. and has no effect on 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.

The setup algorithm of our scheme depends on the supported number of attributes. (figure 1.a) that is the functionality parameter ℓ of the underlying DDH-based IPFE. For AND gate access policies of size $\ell = 100$ the setup time of our DDH-based CP-ABE is around 137ms whereas it is constant (independent of the number of supported attributes) for the other schemes. For FAME, FABEO and Easy-ABE the setup time is approximately 25ms, 11ms and 17ms respectively.

The key generation algorithm of our DDH-based CP-ABE also coincides with that of the underlying DDH-based IPFE. The running time for generating a secret key with 100 attributes is 0.2ms for our scheme whereas it is approximately 682ms, 80ms and 6ms for FAME, FABEO and Easy-ABE respectively (see figure 1.b). It is worth noting that for the experiment we have not implement a stateful key generation algorithm but a stateless one. The low execution time of the key generation algorithm of our DDH-based CP-ABE comes from the fact that it requires only multiplication operations (see table 1) which are far less expensive than exponentiation.

Since the schemes operate in different plaintext spaces, we use them as key encapsulation mechanism (KEM) so that they use the same set of bit strings as plaintext space. Figure 1.c shows the running times when a random plaintext of 16 bytes is encrypted. Easy-ABE has the lowest running time (8ms) because for AND gate access policies a single authorized attribute string is input to the encryption algorithm. For the other schemes, the execution time increases



(a) Setup (b) Key generation

(c) Encryption (d) Decryption

Fig. 1. Average running times for the setup, key generation, encryption and decryption algorithms with AND gate access policies.

linearly with the number of attributes (the number of group operations is linear with respect to the number of attributes see table 2). For 100 attributes the running times are 530ms, 171ms and 137ms for FAME, FABEO and our scheme respectively.

Figure 1.d shows decryption times. For our scheme, the decryption time increases linearly with the number of attributes and reaches 67ms for 100 attributes whereas it grows slowly for the other schemes (constant for Easy-ABE). This result is consistent with the data from table 2. For FAME, FABEO and Easy-ABE the decryption times are 29ms, 14ms and 9ms respectively for 100 attributes.

From table 4 that gives the storage cost of the different schemes, we see that only Easy-ABE does better than our scheme in the AND gate access policy setting. This result is inline with table 3. Recall that for the evaluation of our scheme we choose large integers as attributes in the access policy. Therefore, the relatively large key size comes from this choice.

Table 4. 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.

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

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 less 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. $\boldsymbol{x}_{\mathsf{ID}} = (x_1, \dots, x_\ell) \in \mathbb{L}$.
   3.2. Choose a random vector $\boldsymbol{r} = (r_1, \dots, r_\ell) \in (\mathbb{Z}_q^*)^\ell$.
   3.3. Set $\boldsymbol{y} = \boldsymbol{x}_{\mathsf{ID}} \odot \boldsymbol{r} = (x_1 r_1, \dots, x_\ell r_\ell)$.
   3.4. Compute $c_1 = mg^{\langle \boldsymbol{x}_{\mathsf{ID}}, \boldsymbol{y} \rangle}$.
   3.5. Compute $c_2 \leftarrow \mathsf{IPFE.Encrypt}(mpk, \boldsymbol{y})$.
   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 $m = c_1 / g^\sigma$.

Security. Intuitively the restriction on the size of the identity space ($\ell - 1$ identities corresponding to $\ell - 1$ linearly independent vectors) guarantees to our SIS-IBE scheme resistance against collusion attacks and prevents an adversary to recover $\boldsymbol{y}$ 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 Clustered Identity-Based Encryption

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 instances of SIS-IBE. We call the resulting scheme a *Clustered Identity-Based Encryption*. An implementation in Python is available on GitHub [21]. 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 master keys (MPK and MSK) of our CIBE grows linearly as the number of clusters grows but the instantiation from the DDH assumption (using a DDH based IPFE [4,10]) can give small and constant secret key sizes.

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. To give an example, 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 but 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 (Figures 1.c and 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 from the DDH assumption is also provided. A theoretical and experimental evaluation of our proposal has been done against efficient pairing based ABE schemes (FAME, FABEO and Easy-ABE). 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 constant and small secret key size can be enjoyed. Our results prove that it is possible to construct secure and efficient ABE and IBE schemes from the DDH assumption, thereby contradicting the impossibility conjecture.

References

1. Abdalla, M., Bourse, F., De Caro, A., Pointcheval, D.: Simple functional encryption schemes for inner products. In: Katz, J. (ed.) Public-Key Cryptography – PKC 2015. pp. 733–751. Springer Berlin Heidelberg (2015)
2. Agrawal, S., Chase, M.: FAME: Fast attribute-based message encryption. p. 665–682. CCS '17, Association for Computing Machinery (2017), https://doi.org/10.1145/3133956.3134014

3. Agrawal, S., Bhattacherjee, S., Phan, D.H., Stehlé, D., Yamada, S.: Efficient public trace and revoke from standard assumptions: Extended abstract. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. p. 2277–2293. CCS '17, Association for Computing Machinery (2017), https://doi.org/10.1145/3133956.3134041
4. Agrawal, S., Libert, B., Stehlé, D.: Fully secure functional encryption for inner products, from standard assumptions. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016. pp. 333–362. Springer Berlin Heidelberg (2016)
5. Akinyele, J.A., Garman, C., Miers, I., Pagano, M.W., Rushanan, M., Green, M., Rubin, A.D.: Charm: a framework for rapidly prototyping cryptosystems. Journal of Cryptographic Engineering **3**(2), 111–128 (2013). https://doi.org/10.1007/s13389-013-0057-3
6. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy (SP '07). pp. 321–334 (2007). https://doi.org/10.1109/SP.2007.11
7. Blazy, O., Kakvi, S.A.: Identity-based encryption in DDH hard groups. In: Batina, L., Daemen, J. (eds.) Progress in Cryptology - AFRICACRYPT 2022. pp. 81–102. Springer Nature Switzerland (2022)
8. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) Advances in Cryptology — CRYPTO 2001. pp. 213–229. Springer Berlin Heidelberg (2001)
9. Boneh, D., Papakonstantinou, P., Rackoff, C., Vahlis, Y., Waters, B.: On the impossibility of basing identity based encryption on trapdoor permutations. In: 2008 49th Annual IEEE Symposium on Foundations of Computer Science. pp. 283–292 (2008). https://doi.org/10.1109/FOCS.2008.67
10. Castagnos, G., Laguillaumie, F., Tucker, I.: Practical fully secure unrestricted inner product functional encryption modulo p. In: Peyrin, T., Galbraith, S.D. (eds.) Advances in Cryptology - ASIACRYPT 2018. Lecture Notes in Computer Science, vol. 11273, pp. 733–764. Springer (2018), https://doi.org/10.1007/978-3-030-03329-3_25
11. Dai, W., Doröz, Y., Polyakov, Y., Rohloff, K., Sajjadpour, H., Savaş, E., Sunar, B.: Implementation and evaluation of a lattice-based key-policy ABE scheme. IEEE Transactions on Information Forensics and Security **13**(5), 1169–1184 (2018). https://doi.org/10.1109/TIFS.2017.2779427
12. Ding, S., Li, C., Li, H.: A novel efficient pairing-free CP-ABE based on elliptic curve cryptography for IoT. IEEE Access **6**, 27336–27345 (2018). https://doi.org/10.1109/ACCESS.2018.2836350
13. Döttling, N., Garg, S.: Identity-based encryption from the diffie-hellman assumption. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology – CRYPTO 2017. pp. 537–569. Springer International Publishing (2017)
14. Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness encryption and its applications. In: Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing. p. 467–476. STOC '13, Association for Computing Machinery (2013), https://doi.org/10.1145/2488608.2488667
15. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security. p. 89–98. CCS '06, Association for Computing Machinery (2006), https://doi.org/10.1145/1180405.1180418
16. Herranz, J.: Attribute-based encryption implies identity-based encryption. IET Information Security **11**(6), 332–337 (2017). https://doi.org/10.1049/iet-ifs.2016.0490

17. Herranz, J.: Attacking pairing-free attribute-based encryption schemes. IEEE Access **8**, 222226–222232 (2020). https://doi.org/10.1109/ACCESS.2020.3044143
18. Huang, B., Gao, J., Li, X.: Efficient lattice-based revocable attribute-based encryption against decryption key exposure for cloud file sharing. Journal of Cloud Computing **12**(1), 37 (Mar 2023). https://doi.org/10.1186/s13677-023-00414-w
19. Ka, A.K.: Easy-ABE: An easy ciphertext-policy attribute-based encryption. In: Bella, G., Doinea, M., Janicke, H. (eds.) Innovative Security Solutions for Information Technology and Communications. pp. 168–183. Springer Nature Switzerland (2023)
20. Ka, A.K.: EasyABE github (2023), https://github.com/khoureich/EasyABE
21. Ka, A.K.: CIBE github (2024), https://github.com/khoureich/Clustered-IBE
22. Ka, A.K.: M-Sel: A message selection functional encryption from simple tools. In: Manulis, M., MaimuŢ, D., Teşeleanu, G. (eds.) Innovative Security Solutions for Information Technology and Communications. pp. 79–96. Springer Nature Switzerland (2024)
23. Katsumata, S., Yamada, S.: Non-zero inner product encryption schemes from various assumptions: LWE, DDH and DCR. In: Lin, D., Sako, K. (eds.) Public-Key Cryptography – PKC 2019. pp. 158–188. Springer International Publishing (2019)
24. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Smart, N. (ed.) Advances in Cryptology – EUROCRYPT 2008. pp. 146–162. Springer Berlin Heidelberg (2008)
25. Kowalczyk, L., Wee, H.: Compact adaptively secure ABE for NC^1 from k-lin. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. pp. 3–33. Springer International Publishing (2019)
26. Li, J., Zhang, Y., Ning, J., Huang, X., Poh, G., Wang, D.: Attribute based encryption with privacy protection and accountability for cloudIoT. IEEE Transactions on Cloud Computing **10**(02), 762–773 (apr 2022)
27. Mishra, A.K., Mohapatra, Y.: Hybrid blockchain based medical data sharing with the optimized CP-ABE for e-health systems. International Journal of Information Technology **16**(1), 121–130 (Jan 2024), https://doi.org/10.1007/s41870-023-01625-9
28. Odelu, V., Das, A.K.: Design of a new CP-ABE with constant-size secret keys for lightweight devices using elliptic curve cryptography. Sec. and Commun. Netw. **9**(17), 4048–4059 (nov 2016). https://doi.org/10.1002/sec.1587
29. Odelu, V., Das, A.K., Khurram Khan, M., Choo, K.K.R., Jo, M.: Expressive CP-ABE scheme for mobile devices in IoT satisfying constant-size keys and ciphertexts. IEEE Access **5**, 3273–3283 (2017). https://doi.org/10.1109/ACCESS.2017.2669940
30. Papakonstantinou, P.A., Rackoff, C.W., Vahlis, Y.: How powerful are the DDH hard groups? Cryptology ePrint Archive, Paper 2012/653 (2012), https://eprint.iacr.org/2012/653
31. Riepel, D., Wee, H.: FABEO: Fast attribute-based encryption with optimal security. p. 2491–2504. CCS '22, Association for Computing Machinery (2022), https://doi.org/10.1145/3548606.3560699
32. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. pp. 457–473. Springer Berlin Heidelberg (2005)
33. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) Advances in Cryptology. pp. 47–53. Springer Berlin Heidelberg (1985)

34. Shi, J., Yu, Q., Yu, Y., Wang, L., Zhang, W.: Privacy protection in social applications: A ciphertext policy attribute-based encryption with keyword search. International Journal of Intelligent Systems **37**(12), 12152–12168 (2022), https://onlinelibrary.wiley.com/doi/abs/10.1002/int.23080
35. Tan, S.Y., Yeow, K.W., Hwang, S.O.: Enhancement of a lightweight attribute-based encryption scheme for the internet of things. IEEE Internet of Things Journal **6**(4), 6384–6395 (2019). https://doi.org/10.1109/JIOT.2019.2900631
36. Tseng, Y.F., Huang, J.J.: Cryptanalysis on two pairing-free ciphertext-policy attribute-based encryption schemes. In: 2020 International Computer Symposium (ICS). pp. 403–407 (2020). https://doi.org/10.1109/ICS51289.2020.00086
37. Wang, G., Wan, M., Liu, Z., Gu, D.: Fully secure lattice-based ABE from noisy linear functional encryption. In: Yu, Y., Yung, M. (eds.) Information Security and Cryptology. pp. 421–441. Springer International Publishing (2021)
38. Wang, Y., Chen, B., Li, L., Ma, Q., Li, H., He, D.: Efficient and secure ciphertext-policy attribute-based encryption without pairing for cloud-assisted smart grid. IEEE Access **8**, 40704–40713 (2020). https://doi.org/10.1109/ACCESS.2020.2976746
39. Wang, Z., Fan, X., Liu, F.H.: FE for inner products and its application to decentralized ABE. In: Lin, D., Sako, K. (eds.) Public-Key Cryptography – PKC 2019. pp. 97–127. Springer International Publishing, Cham (2019)
40. Yao, X., Chen, Z., Tian, Y.: A lightweight attribute-based encryption scheme for the internet of things. Future Generation Computer Systems **49**, 104–112 (2015). https://doi.org/10.1016/j.future.2014.10.010
41. Zhang, Y., Deng, R.H., Xu, S., Sun, J., Li, Q., Zheng, D.: Attribute-based encryption for cloud computing access control: A survey. ACM Comput. Surv. **53**(4) (aug 2020), https://doi.org/10.1145/3398036

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 [4].

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.

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. 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. 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 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

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

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

IND-CPA security. Indistinguishability under chosen plaintext attack security for CP-ABE is formally defined via the following game:

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

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

4. $\mathsf{Key\ query\ 2}$: key query 1 is repeated.

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

It is required that no attribute set $\mathbb{S}_i$ in the key query phases satisfies the access policy $\mathbb{P}$ in the challenge phase. But when the CP-ABE scheme is a secret access policy one as in our case, the restriction on the attribute sets queried by the adversary is no longer necessary since the access policy is secret and chosen by the challenger (the encryptor).

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.

IND-CPA security. Boneh et *al.* [8] 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. $\mathsf{Key\ query\ 1}$: 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 selects $b \in \{0,1\}$, 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 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 a vector $\boldsymbol{P} \in \mathbb{Z}_q^\ell$ and 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 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 The DDH based IPFE scheme of Agrawal et al.

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

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

F Proof of Theorem 2

Assume there exists an IND-CPA adversary $\mathcal{A}$ that has 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 $\{\mathsf{ID}_i\}_{i=1}^q,\ q \le \ell - 2$ 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 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.