

Anonymous Attribute-Based Signcryption: Definitions, Constructions, and Applications

Yongkang Lang¹, Fangguo Zhang^{3,4}, Zhiyuan An^{3,4}, Xinyi Huang⁵, and
Xiaofeng Chen²

¹ School of Cryptology, Xidian University, Xi'an 710071, China
lang_yongkang@163.com

² State Key Laboratory of Integrated Service Networks (ISN),
Xidian University, Xi'an, 710071, China
xfchen@xidian.edu.cn

³ School of Computer Science and Engineering,
Sun Yat-sen University, Guangzhou 510006, China
isszhfg@mail.sysu.edu.cn, anzhy@mail2.sysu.edu.cn

⁴ Guangdong Provincial Key Laboratory of Information Security,
Guangzhou 510006, China

⁵ College of Computer Science and Technology,
Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China
xyhuang81@gmail.com

Abstract. We put forward a generalization of attribute-based signcryption, called anonymous attribute-based signcryption (A²BSC). Beyond message confidentiality and ciphertext unforgeability, A²BSC further requires *ciphertext anonymity*: no information about the signcryptor's attributes or ciphertext-related attributes/policies is leaked, regardless of the decryption outcome. Specifically, we begin by establishing the syntax and security notions for A²BSC within a *unified* framework, which encompasses various variants (key-policy, ciphertext-policy, dual-policy, and a hierarchical dual-policy variant called Special A²BSC). Then, we construct a Special A²BSC scheme for *general policies* (modeled as bounded-depth Boolean circuits) from the succinct learning with errors and the basis-augmented short integer solution assumptions in the standard model, hence achieving post-quantum security. This naturally yields lattice-based instantiations of both ciphertext-policy and dual-policy A²BSC.

Beyond its independent interest, we also show the expressiveness and generality of our A²BSC by exploring its application to matchmaking encryption (ME) and arranged matchmaking encryption (AME) proposed by Ateniese et al. (Crypto '19). As a byproduct, we give generic constructions of both ME and AME for *arbitrary policies* against unbounded collusions, and strengthen the CPA-privacy of (A)ME to achieve CCA security. The latter is achieved for free in our construction, as A²BSC natively provides CCA security. Overall, our new solution enhances the diversity of approaches for building the advanced primitive (A)ME.

Keywords: Attribute-based signcryption · Anonymity · Matchmaking encryption · Lattice-based cryptography · Circuits

Table of Contents

Anonymous Attribute-Based Signcryption: Definitions, Constructions, and Applications	1
1 Introduction	3
1.1 Our Contributions	4
1.2 Technical Overview	5
2 Preliminaries	10
2.1 Arranged Matchmaking Encryption	10
2.2 Matchmaking Encryption	13
3 Anonymous Attribute-Based Signcryption	15
3.1 Syntax of A ² BSC	15
3.2 Security Definitions of A ² BSC	16
3.3 Various Variants of A ² BSC	19
4 Anonymous Attribute-Based Signcryption for Circuits from Lattices	22
4.1 High-Level Overview	22
4.2 Special A ² BSC for Circuits from Lattices	23
4.3 Special A ² BSC with Separable Decryption Keys for Circuits from Lattices	27
5 Generic Constructions of Matchmaking Encryption	28
5.1 AME from Special A ² BSC	28
5.2 ME from Special A ² BSC	30
A Cryptographic Building Blocks and Lattices Background	36
A.1 Functional Commitment with Private Opening	36
A.2 Functional Encryption	37
A.3 Lockable Obfuscation	38
A.4 Symmetric Key Encryption	38
A.5 Lattices Background	39
B Error Bound and Parameters	42
C Missing Proofs	43
C.1 Proof of Theorem 2	43
C.2 Proof of Theorem 3	46
C.3 Proof of Theorem 4	47
C.4 Proof of Theorem 5	48
C.5 Proof of Theorem 8	50
C.6 Proof of Theorem 9	55

1 Introduction

Signcryption. In the realm of public-key cryptography, message confidentiality and authenticity are two fundamental security requirements. Signcryption [9, 67] was originally introduced to optimize efficiency, achieving these dual objectives at a cost significantly lower than the traditional “sign-then-encrypt” sequential composition. Subsequently, An, Dodis and Rabin [4] broadened this paradigm by introducing the “commit-then-encrypt-and-sign” composition. They adopted the term “signcryption” to broadly denote any scheme that simultaneously achieves privacy and authenticity in the public-key setting, emphasizing that signcryption should be explored as an *independent* primitive. Beyond its substantial efficiency advantages, signcryption inherently provides several attractive security properties, including indistinguishability under chosen-ciphertext attacks (IND-CCA), unforgeability, and resistance against “recipient transfer” forgeries [9].

Attribute-Based Signcryption. As application scenarios grow in complexity, attribute-based signcryption (ABSC) [29] was introduced as a generalization of signcryption, which integrates attribute-based encryption (ABE) [34, 51] and attribute-based signature (ABS) [45] into a unified cryptographic primitive. In ABSC, a trusted authority generates a signing secret key $\text{ssk}_{\mathbf{z}}$ for a signcryptor with attribute $\mathbf{z} \in \mathcal{Z}$, and a decryption secret key $\text{dsk}_{\mathbf{y}}$ for a receiver with attribute $\mathbf{y} \in \mathcal{Y}$. If $C(\mathbf{z}) = 1$, the signcryptor can use $\text{ssk}_{\mathbf{z}}$ to produce a valid ciphertext, which is encrypted with policy $G : \mathcal{Y} \rightarrow \{0, 1\}$ chosen by the signcryptor and signed with policy $C : \mathcal{Z} \rightarrow \{0, 1\}$. During decryption, if $G(\mathbf{y}) = 1$, the receiver can not only successfully decrypt the message, but also verify its origin. The ABSC security properties are as follows:

- a) **Message Confidentiality.** A property inherited from ABE.
- b) **Ciphertext Unforgeability.** Inherited from ABS, it ensures that even if multiple signcryptors with attributes $\{\mathbf{z}_i\}_{i \in [Q]}$ collude, they cannot forge a valid ciphertext for policy C , as long as $C(\mathbf{z}_i) \neq 1$ for all $i \in [Q]$.
- c) **Signcryptor Anonymity.** Inherited from ABS, it ensures that given a valid ciphertext associated with policy C , if $C(\mathbf{z}_0) = C(\mathbf{z}_1) = 1$, then it is infeasible to distinguish whether the ciphertext originated from a signcryptor with attribute $\mathbf{z}_0$ or $\mathbf{z}_1$.

Despite these appealing properties, ABSC still faces a limitation: it only protects message confidentiality but fails to provide privacy for the ciphertext policy G (or attribute), which is transmitted alongside the ciphertext in the clear. This is particularly concerning in privacy-critical scenarios such as healthcare [40, 42, 65, 66], blockchain-based access control [25] and edge computing [61], where policies/attributes are inherently *sensitive* and must remain *hidden*. In contrast, in a simple setting, identity-based signcryption [16, 20] (a variant with weak expressiveness for policies) has achieved this security goal, namely ciphertext anonymity. Moreover, surprisingly, compared to the rapid strides recently made in ABE [2, 12, 22, 23, 37, 56–58] and ABS [26, 36, 38, 41, 43, 52] for *general policies*, foundational research on ABSC remains relatively sparse and fragmented.

At present, existing ABSC constructions [29, 48, 49, 64] suffer from limited expressiveness by only supporting the class $\mathcal{NC}^1$, unless one relies on heavier tools such as indistinguishability obfuscation (*iO*) [24]. More critically, the community appears to have paid insufficient attention to this independent primitive in its own right, leading to an oversight of its deep connections with recent advanced cryptographic primitives.

1.1 Our Contributions

In this work, we systematically study ABSC and introduce a novel primitive, dubbed *anonymous attribute-based signcryption* (A^2BSC). In a nutshell, A^2BSC not only inherits the message confidentiality and ciphertext unforgeability properties of ABSC, but also provides a new security notion, namely *ciphertext anonymity*. This guarantees that no information about the signcryptor's attributes or the ciphertext-associated attributes/policies is revealed, regardless of whether decryption succeeds. More concretely, we obtain:

- *Formal Definitions and Security Notions of A^2BSC .* We present the formal definition of A^2BSC , which generalizes ABSC by introducing anonymity. We also define three security notions for A^2BSC : message confidentiality, ciphertext unforgeability, and ciphertext anonymity. Notably, our definitions and security notions are formulated for the *full form* of A^2BSC over the relations $\mathcal{R}_{\text{sign}}$ and $\mathcal{R}_{\text{dec}}$. In particular, valid ciphertexts are generated only if the relation $\mathcal{R}_{\text{sign}}$ is satisfied, and decryption succeeds only if the relation $\mathcal{R}_{\text{dec}}$ is satisfied, where
 - for $\mathcal{R}_{\text{sign}}$, a signcryptor key is associated with both policy $V : \mathcal{V} \rightarrow \{0, 1\}$ and attribute $\mathbf{z} \in \mathcal{Z}$, which must satisfy the corresponding attribute $\mathbf{v} \in \mathcal{V}$ and policy $C : \mathcal{Z} \rightarrow \{0, 1\}$ embedded in the signature component of the valid ciphertext.
 - for $\mathcal{R}_{\text{dec}}$, a decryption key is associated with policy $F : \mathcal{X} \rightarrow \{0, 1\}$ and attribute $\mathbf{y} \in \mathcal{Y}$, which must satisfy the corresponding attribute $\mathbf{x} \in \mathcal{X}$ and policy $G : \mathcal{Y} \rightarrow \{0, 1\}$ embedded in the ciphertext.

The full form of our A^2BSC provides a unified framework that encompasses a variety of concepts. By specializing these relations $\mathcal{R}_{\text{sign}}$ and $\mathcal{R}_{\text{dec}}$, we obtain the anonymous variants of the well-known key-policy ABSC, ciphertext-policy ABSC, and dual-policy ABSC. Moreover, we present a variant of dual-policy A^2BSC with hierarchical signcryptor keys, dubbed *Special A^2BSC* (see Sect. 3.3).

- *Constructions of Special A^2BSC for Circuits from Lattices.* We construct a lattice-based *Special A^2BSC* scheme for Boolean circuits of bounded depth d , which naturally yields ciphertext-policy/dual-policy A^2BSC from lattices. In the standard model, our scheme achieves message confidentiality and ciphertext anonymity under the $\text{poly}(d, \lambda)$ -succinct learning with errors (LWE) assumption [57], and ciphertext unforgeability under the basis-augmented short integer solution (BASIS) assumption [59]. Moreover, we extend it to

Special A²BSC with separable decryption keys, which splits the decryption key into two pieces: one associated with F and the other with $\mathbf{y}$.

Application to (Arranged) Matchmaking Encryption. Beyond its independent interest, we further show the expressiveness and generality of our A²BSC by exploring its application to matchmaking encryption (ME) and arranged ME (AME) [5, 6]. Interestingly, we observe an in-depth connection between the security requirements of (A)ME (i.e., strict “attributes” *privacy* and strong mutual *authentication*) and the native guarantees (i.e., Items **a**)–**c**) provided by our A²BSC. Motivated by this, we offer a novel view of (A)ME from the A²BSC perspective. Naturally, we give generic constructions of both ME and AME for *arbitrary policies* against unbounded collusions,¹ and enhance the security definition of (A)ME to achieve CCA security. This yields the following result:

Theorem 1 (Informal). *Assuming the existence of a secure Special A²BSC (resp., Special A²BSC with separable decryption keys) scheme for arbitrary policies, there exists an AME (resp., ME) scheme for arbitrary policies.*

Crucially, this transformation inherently provides the resulting schemes with CCA security. Plugging our lattice-based Special A²BSC constructions (Sect. 4) and existing cryptographic tools, we obtain the first lattice-based CCA-secure ME and AME schemes for circuits in the *standard* model.²

On a broader scale, we believe adding to the diversity of approaches for (A)ME is valuable. As Waters [53] noted regarding CCA security, diverse research pathways not only create a clearer picture but also enable broader applications. We hope our approach, based on A²BSC, contributes to the diversity of (A)ME and, in turn, encourages further study of A²BSC as an independent primitive.

1.2 Technical Overview

In this section, we provide an overview of our techniques. We begin by formalizing the notions of A²BSC.

Formalizing Anonymous Attribute-Based Signcryption. Our full form of A²BSC allows both the signcryptor and the receiver to possess a *composite* “attribute”, which consists of a policy and an attribute. Unlike regular ABSC, our A²BSC places both the signcryptor and the receiver under a dual-policy framework, which supports *bilateral* access control. In our setting, the authority generates

- a signing secret key $\text{ssk}_{V,\mathbf{z}}$ for a signcryptor with “attribute” $(V, \mathbf{z})$ where policy $V : \mathcal{V} \rightarrow \{0, 1\}$ and attribute $\mathbf{z} \in \mathcal{Z}$;

¹ We say that a scheme is secure against unbounded collusion attacks [6] if a malicious receiver can make an unbounded number of queries to the decryption key oracles.

² While prior black-box constructions of (A)ME [5, 6, 27, 28] could yield similar results by substituting lattice-based non-interactive zero-knowledge proofs for $\mathcal{NP}$ relations ($\text{NIZK}_{\mathcal{NP}}$) in the standard model, almost all existing efficient lattice-based $\text{NIZK}_{\mathcal{NP}}$ [44, 63] are in the random oracle model.

- a decryption secret key $\text{dsk}_{F,\mathbf{y}}$ for a receiver with “attribute” $(F, \mathbf{y})$ where policy $F : \mathcal{X} \rightarrow \{0, 1\}$ and attribute $\mathbf{y} \in \mathcal{Y}$.

To encrypt a message μ , the signcryptor uses $\text{ssk}_{V,\mathbf{z}}$ to produce a *valid* ciphertext ct , which is encrypted with $(\mathbf{x} \in \mathcal{X}, G : \mathcal{Y} \rightarrow \{0, 1\})$ chosen by the signcryptor and signed with $(\mathbf{v} \in \mathcal{V}, C : \mathcal{Z} \rightarrow \{0, 1\})$. We say that a ciphertext is deemed valid if the conjunction $V(\mathbf{v}) = 1 \wedge C(\mathbf{z}) = 1$ holds. On the other side, the receiver can successfully decrypt if the conjunction $F(\mathbf{x}) = 1 \wedge G(\mathbf{y}) = 1$ holds. The security properties of A^2BSC are as follows.

- Message Confidentiality.** For any ciphertext encrypted with $(\mathbf{x}, G)$, no adversary can distinguish whether it contains μ_0 or μ_1 , unless it possesses a key $\text{dsk}_{F,\mathbf{y}}$ that satisfies $F(\mathbf{x}) = 1 \wedge G(\mathbf{y}) = 1$.
- Ciphertext Unforgeability.** It ensures that no adversary can forge a valid ciphertext signed with $(\mathbf{v}, C)$, unless it possesses a key $\text{ssk}_{V,\mathbf{z}}$ that satisfies $V(\mathbf{v}) = 1 \wedge C(\mathbf{z}) = 1$.
- Ciphertext Anonymity.** Similar to regular ABSC, for a ciphertext ct signed with $(\mathbf{v}, C)$, it provides signcryptor anonymity. In particular, if $V_0(\mathbf{v}) = V_1(\mathbf{v}) = 1$ and $C(\mathbf{z}_0) = C(\mathbf{z}_1) = 1$, the adversary cannot determine whether ct originated from a signcryptor with $(V_0, \mathbf{z}_0)$ or $(V_1, \mathbf{z}_1)$. Beyond this, it introduces two levels of ciphertext “attribute” anonymity:
 - *1-sided Ciphertext Anonymity:* No adversary can distinguish whether ct is encrypted with $(\mathbf{x}_0, G_0)$ or $(\mathbf{x}_1, G_1)$, unless it holds a decryption key that can decrypt ct .
 - *2-sided Ciphertext Anonymity:* This is a stronger security notion, guaranteeing that even if the adversary can successfully decrypt ct , it still cannot distinguish whether ct is encrypted with $(\mathbf{x}_0, G_0)$ or $(\mathbf{x}_1, G_1)$.

We construct a **Special A^2BSC** scheme for arbitrary (bounded-depth) circuit policies from lattices, and further extend it to a variant supporting separable decryption keys. Due to the heavy notation, we defer its high-level overview to Sect. 4.1.

Background on Matchmaking Encryption. Before explaining how to construct ME from Special A^2BSC , we briefly recall the background of ME [5, 6]. It enables both the sender and the receiver to specify policies that the other party must satisfy, ensuring that a ciphertext can be decrypted if and only if both parties’ policies are simultaneously satisfied. In an ME scheme, a sender encrypts a message μ under an encryption key $\text{ek}_{\mathbf{x}}$ associated with attribute $\mathbf{x} \in \mathcal{X}$, and a policy $G : \mathcal{Y} \rightarrow \{0, 1\}$ specifying which receivers may decrypt. On the receiver side, one holds a key $\text{dk}_{\mathbf{y}}$ for attribute $\mathbf{y} \in \mathcal{Y}$, along with a key dk_F for a policy $F : \mathcal{X} \rightarrow \{0, 1\}$. Decryption succeeds if $F(\mathbf{x}) = G(\mathbf{y}) = 1$ (i.e., a match occurs). Crucially, the security guarantees are privacy and authenticity:

- *Privacy* ensures that, in the case of a mismatch, the receiver learns nothing beyond the failure of decryption; in the case of a match, nothing is revealed except the message μ and the fact that a match occurred.
- *Authenticity* ensures that only holders of a valid encryption key $\text{ek}_{\mathbf{x}}$ can produce a valid ciphertext w.r.t. any policy that is satisfied by $\mathbf{x}$.

Along with ME, there are two variants: (i) AME is the same as regular ME, except that the receiver holds a key $\text{dk}_{F,\mathbf{y}}$ that combines the policy F and attribute $\mathbf{y}$; (ii) identity-based ME is a weak variant that only supports equality policies over strings.

Rethinking the AME Construction. For ease of exposition, our starting point is the relatively easy-to-construct AME. In the original work of [5, 6], a black-box construction of AME is presented by considering functional encryption (FE) [13] for the following functionality $\mathbb{F} : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}^* \cup \{\perp\}$:

$$\mathbb{F}_{(F,\mathbf{y})}(\mathbf{x}, G, \mu) = \begin{cases} \mu, & F(\mathbf{x}) = 1 \wedge G(\mathbf{y}) = 1 \\ \perp, & \text{otherwise.} \end{cases}$$

Here, $\mathbf{x}$ and G are the sender's attributes and the specified policy, respectively. For the receiver's attribute $\mathbf{y}$ and policy F , the authority issues the AME decryption key $\text{dk}_{F,\mathbf{y}} := \text{sk}_{F,\mathbf{y}}$, where $\text{sk}_{F,\mathbf{y}}$ is a functional key for $\mathbb{F}_{(F,\mathbf{y})}$ in FE. Naturally, AME privacy follows from the FE security. Without considering authenticity, AME seems readily captured by other primitives, such as predicate encryption [14, 39] or dual-policy ABE [7, 8]. However, when it comes to simultaneously achieving both privacy and authenticity, the challenge becomes non-trivial. As noted by Ateniese et al. [5, 6], constructing (A)ME is quite *difficult*. In other words, for the sender to generate a valid AME ciphertext, the following challenge must be addressed:

How can the sender $\mathbf{S}$ convince a receiver that $\mathbf{S}$ possesses an attribute $\mathbf{x}$ satisfying the policy F specified by the receiver, without revealing $\mathbf{x}$?

Note that the sender $\mathbf{S}$ does not know the receiver's policy F during encryption. A direct (albeit somewhat brute-force) approach to address this is to leverage the power of NIZK. This is exemplified by the construction of [6], which employs both $\text{NIZK}_{\mathcal{NP}}$ and signature techniques. Roughly speaking, the trust authority generates a signature σ on $\mathbf{x}$ via signature scheme, which serves as the sender's encryption key $\text{ek} = (\mathbf{x}, \sigma)$. Then, during encryption, the sender constructs the FE ciphertext ct_{FE} followed by a NIZK proof π for the following $\mathcal{NP}$ relation:

$$R := \left\{ ((\text{pk}, \text{ct}_{\text{FE}}, \text{mpk}_{\text{FE}}), (\mathbf{x}, \sigma)) : \begin{array}{l} \exists r, \mu, G \text{ s.t.} \\ \text{ct}_{\text{FE}} = \text{Enc}_{\text{FE}}(\text{mpk}_{\text{FE}}, (\mathbf{x}, G, \mu); r) \wedge \\ \text{Vrfy}(\text{pk}, \sigma, \mathbf{x}) = 1 \end{array} \right\} \quad (1)$$

where pk and Vrfy are the public key and verification algorithm of signature. Here, we adopt the term "FE-then-NIZK" to refer to the paradigm for AME. Observe that this complex relation R within NIZK is designed precisely to achieve the goal described above in *italics*. Unfortunately, despite the notable recent progress on lattice-based $\text{NIZK}_{\mathcal{NP}}$ in the standard model, it is still far from satisfactory. Known constructions [17, 47, 53, 55] require additional building blocks (e.g., correlation intractable hash functions, hidden bit generator) at a high cost. Having clarified the role of NIZK, our motivation is to explore whether this goal can be naturally captured via our A²BSC from lattices in the standard model.

Warm-up: AME from A²BSC. At first glance, the AME security goals seem to be a mixture of confidentiality, unforgeability, and anonymity. From this perspective, ME may bear certain similarities to our A²BSC. In this warm-up, we attempt to construct AME from dual-policy A²BSC, addressing privacy and authenticity in turn.

Achieving Privacy. We start by showing that AME privacy can be achieved using dual-policy A²BSC. More specifically, we treat a signing secret key $\text{ssk}_{\mathbf{x}}$ (setting $\mathbf{z} := \mathbf{x}$) of A²BSC³ as an encryption key $\text{ek}_{\mathbf{x}}$ of AME, and a decryption secret key $\text{dsk}_{F,\mathbf{y}}$ of A²BSC as a decryption key $\text{dk}_{F,\mathbf{y}}$. During encryption, the sender's attribute $\mathbf{x}$ is assigned to the first slot of dual-policy A²BSC, while the policy G is assigned to the second slot. Decryption succeeds only if a match occurs (i.e., $F(\mathbf{x}) = 1 \wedge G(\mathbf{y}) = 1$). Moreover, the ciphertext anonymity of A²BSC ensures that $\mathbf{x}$ and G are not leaked regardless of whether the matching holds.

Achieving Authenticity. We turn to the AME authenticity. In [5, 6], this property is achieved via $\text{NIZK}_{\mathcal{NP}}$. Inspired by the work of [38], we treat a witness w in NIZK as a signing secret key $\text{ssk}_{\mathbf{x}}$. To prove that w is a valid witness for the statement ϕ (i.e., $(\phi, w) \in R$ where R is an $\mathcal{NP}$ relation), we construct a (policy) circuit $C_{\phi}(w) := R(\phi, w)$ with the statement ϕ hardwired. We then use $\text{ssk}_{\mathbf{x}}$ to produce a ciphertext ct that is signed with C_{ϕ} and encrypted with $(\mathbf{x}, G)$, where the signature component is treated as the proof π . The receiver validates the proof π by checking the validity of ct . Here, the soundness of NIZK follows from the ciphertext unforgeability of A²BSC, while the zero-knowledge property follows from the ciphertext anonymity of A²BSC. Unfortunately, the circuit $C_{\phi}(w)$ cannot be used directly to capture the relation (1) used in the $\text{NIZK}_{\mathcal{NP}}$ of [5, 6]. This is because the ciphertext-related $(\mathbf{x}, G)$ must be made public as part of the statement ϕ . In our context, $(\mathbf{x}, G)$ must remain hidden for AME. Despite this obstacle, A²BSC offers a strawman approach that achieves the privacy of AME and a limited form of authenticity (where sender S can only convince the receiver that S possesses some attribute key). This observation suggests the plausibility of constructing AME from A²BSC.

Application of A²BSC to AME. Following the above discussion, let us identify what prevents this: if we were to encode (1) directly into a circuit C_{ϕ} (where Enc_{FE} is replaced by the encryption component of A²BSC), it would necessitate that $(\mathbf{x}, G)$ be exposed as part of the statement ϕ . Intuitively, to achieve privacy, the hardwired constants in C_{ϕ} must exclude $(\mathbf{x}, G)$. To this end, we consider a variant of A²BSC (denoted as Special A²BSC), which integrates hierarchical signing secret keys into the dual-policy A²BSC framework. In Special A²BSC, an authority generates a *partial* signing secret key $\text{ssk}_{\mathbf{x}}$ for a signcryptor with attribute $\mathbf{x}$. Moreover, for any $\mathbf{x}'$, the signcryptor can locally derive a *complete* signing secret key $\text{ssk}_{\mathbf{x}||\mathbf{x}'}$ from $\text{ssk}_{\mathbf{x}}$. This capability allows the signcryptor to generate valid ciphertexts containing signatures under a circuit C that satisfies $C(\mathbf{x}||\mathbf{x}') = 1$. In addition, we observe that functional commitment (FC) [59] of-

³ In this dual-policy A²BSC, the signcryptor's key is associated solely with $\mathbf{z}$, while the policy V is set as the constant function that always outputs 1 (hence omitted).

Setup : Run the Setup algorithms of A^2BSC , FE and FC respectively, and output $mpk := (mpk_{A^2BSC}, mpk_{FE}, crs_{FC})$, $msk := (msk_{A^2BSC}, msk_{FE})$.

SKGen : The authority generates a partial signing secret key ssk_x of A^2BSC and issues it to a user with attribute x as the encryption key ek_x .

RKGen : The authority generates a decryption key $dk_{F,y} := (dsk_{F,y}, sk_{Eval_F})$ and issues it to a user with (F, y) , where $dsk_{F,y}$ is the decryption secret key of A^2BSC , and sk_{Eval_F} is the functional key of FE for the circuit F_{Eval} with crs_{FC} and F hardwired as follows: $F_{Eval}[crs_{FC}, F](st) := FC.Eval(crs_{FC}, st, F)$.

Enc : On input mpk, ek_x, G and μ , do the following.

1. Run $(\sigma, st) \leftarrow FC.Commit(crs_{FC}, x)$, where $st := (x, \tilde{st})$ is a state of FC .
2. Run $ct_{FE} \leftarrow FE.Enc(mpk_{FE}, st)$.
3. Parse ek_x as ssk_x and derive a complete signing key $ssk_{x||x'}$ from ssk_x , where $x' := \tilde{st}$. Prepare a circuit $C[\sigma]$ with σ hardwired such that
$$C[\sigma](x||x') := (1 = FC.Verify(crs_{FC}, \sigma, F_x, 1, \pi_{F_x}))$$
where F_x is an equality-test function for x that outputs 1 if the input equals x and 0 otherwise, and $\pi_{F_x} \leftarrow FC.Eval(crs_{FC}, st, F_x)$. It then uses $ssk_{x||x'}$ to produce an A^2BSC ciphertext ct_1 for the message $\mu' := (\mu, ct_{FE})$, which is signed with $C[\sigma]$ and encrypted with (x, G) .
4. Output $ct = (ct_1, \sigma)$.

Dec : If the **Unsigncrypt** algorithm of A^2BSC outputs (μ, ct_{FE}) , compute an opening π_F via $FE.Dec(sk_{Eval_F}, ct_{FE})$. Output μ if $FC.Verify(crs_{FC}, \sigma, F, 1, \pi_F) = 1$ and $\perp$ otherwise.

Fig. 1. High-level construction of our AME.

fers a highly appealing property for our setting, namely *private openings*. In an FC with private opening, given a commitment σ to an attribute x along with its state $st := (x, \tilde{st})$, one can invoke the **Eval** algorithm on st to compute an opening π_F for a function F . Here, the private opening of FC ensures that σ and π_F reveal nothing about x beyond the value $F(x)$. This resolves the challenge we face. On the sender's side, by embedding σ as a hardwired constant within C , the sender implicitly proves possession of the underlying attribute x without compromising its privacy. On the receiver's side, π_F allows the receiver to verify whether x committed in σ satisfies F . Since the state st cannot be directly provided to the receiver to compute π_F , we encrypt st using FE . This allows the receiver to derive π_F from the ciphertext while st remains hidden. Rather than remaining abstract, we jump straight into our construction. Fig. 1 illustrates a high-level construction of AME from A^2BSC and several tools, including FC with private opening (Definition 14) and FE (Definition 17).

The encryption procedure can be intuitively summarized in three steps: (i) commit to attribute x to obtain the commitment σ and the state st ; (ii) encrypt the state st via FE ; (iii) construct a circuit $C[\sigma]$, and use the A^2BSC Signcrypt algorithm to generate a valid ciphertext under $C[\sigma]$, where its validity implies that sender S proves to receiver that S possesses the attribute associated with σ . In this way, the receiver

- **[First check]** checks whether the sender possesses the attribute committed to by σ through decrypting the A²BSC ciphertext;
- **[Second check]** if the above holds, the receiver further uses the FC verification algorithm to verify π_F , checking whether the attribute associated with σ satisfies F (i.e., $F(\mathbf{x}) \stackrel{?}{=} 1$).

Crucially, the binding property of FC guarantees that the attribute verified in the **first check** is consistent with the one confirmed in the **second check**. Moreover, the A²BSC anonymity, the FE security and the private opening of FC ensure that the ciphertext (ct_1, σ) does not leak any information about $\mathbf{x}$.

Application of A²BSC to ME. Our generic construction is not limited to building AME. By leveraging the modularity of our design, it can be adapted to construct ME. We stress that this does not mean AME can be turned into ME, as the two notions are incomparable [5, 6]. Instead, we observe that whenever the underlying A²BSC scheme enjoys separable decryption keys, this naturally yields an ME scheme from our generic construction (see Sect. 5.2). Simply put, the dual-policy matching condition in ME is a conjunction $F(\mathbf{x}) = 1 \wedge G(\mathbf{y}) = 1$. Since the two clauses are logically independent, the underlying decryption secret key can be split. In such a setting, the algorithm for generating $\text{dsk}_{F,\mathbf{y}}$ can be *separated* into two independent parts: computing dsk_F for F and $\text{dsk}_{\mathbf{y}}$ for $\mathbf{y}$. Only by possessing both dsk_F and $\text{dsk}_{\mathbf{y}}$ can one reconstruct $\text{dsk}_{F,\mathbf{y}}$. This can be readily achieved in the lattice-based setting using a minor lattice “trick” (see Sect. 4.3), which we do not detail here.

2 Preliminaries

Notations. For a positive integer $k \in \mathbb{N}$, we denote the set $\{1, \dots, k\}$ by $[k]$. Row vectors and matrices are denoted by bold lowercase letters (e.g., $\mathbf{v}$) and bold uppercase letters (e.g., $\mathbf{V}$), respectively. We let $\mathbf{1}_k$ denote the k -dimensional row vector with all entries being 1. For any $\mathbf{v}, \mathbf{V}$ over $\mathbb{Z}$, we use $\|\mathbf{v}\|$ and $\|\mathbf{V}\|$ to denote the maximum absolute value over all their entries. For vectors $\mathbf{u}, \mathbf{v}$ (resp. matrices $\mathbf{U}, \mathbf{V}$), we use $[\mathbf{u}|\mathbf{v}]$ (resp. $[\mathbf{U}|\mathbf{V}]$) to denote the horizontal concatenation of $\mathbf{u}$ and $\mathbf{v}$ (resp. $\mathbf{U}$ and $\mathbf{V}$). We write $v \leftarrow \mathcal{D}$ to denote sampling v from a distribution $\mathcal{D}$, and $v \leftarrow S$ to denote sampling v uniformly from a set S . Let $\otimes$ denote the Kronecker product of matrices. We use $\approx_s$ (resp. $\approx_c$) to denote statistical (resp. computational) indistinguishability. $\text{Round}_\beta(v)$ denotes a function that outputs 0 if $\|v\| < \beta$ and 1 otherwise. We let $\text{poly}(\lambda)$ and $\text{negl}(\lambda)$ denote a fixed polynomial in λ and a function that is $o(\lambda^{-c})$ for all $c \in \mathbb{N}$, respectively.

We recall the definitions of underlying building blocks, such as FC, FE, and lockable obfuscation (LO), along with lattice background, in Appendix A.

2.1 Arranged Matchmaking Encryption

Definition 1 (Arranged Matchmaking Encryption [5, 6]). Let λ be a security parameter, $\mathcal{X}_\lambda$ and $\mathcal{G}_\lambda$ be sender’s attribute and policy spaces, $\mathcal{Y}_\lambda$ and $\mathcal{F}_\lambda$

be receiver's attribute and policy spaces, and $\mathcal{M} = \mathcal{M}(\lambda)$ be a message space. An AME scheme Π_{AME} consists of the following efficient algorithms:

- $\text{Setup}(1^\lambda) \rightarrow (\text{mpk}, \text{msk})$: On input the security parameter λ , the Setup algorithm outputs the master public key mpk and the master secret key msk .
- $\text{SKGen}(\text{mpk}, \text{msk}, \mathbf{x}) \rightarrow \text{ek}_{\mathbf{x}}$: On input the master public key mpk , the master secret key msk , and an attribute vector $\mathbf{x} \in \mathcal{X}_\lambda$, the SKGen algorithm outputs an encryption key $\text{ek}_{\mathbf{x}}$.
- $\text{RKGen}(\text{mpk}, \text{msk}, F, \mathbf{y}) \rightarrow \text{dk}_{F, \mathbf{y}}$: On input the master public key mpk , the master secret key msk , a policy function $F \in \mathcal{F}_\lambda$ and an attribute vector $\mathbf{y} \in \mathcal{Y}_\lambda$, the RKGen algorithm outputs a decryption key $\text{dk}_{F, \mathbf{y}}$.
- $\text{Enc}(\text{mpk}, \text{ek}_{\mathbf{x}}, G, \mu) \rightarrow \text{ct}$: On input the master public key mpk , the encryption key $\text{ek}_{\mathbf{x}}$ for $\mathbf{x} \in \mathcal{X}_\lambda$, a policy $G \in \mathcal{G}_\lambda$ and a message $\mu \in \mathcal{M}$, the Enc algorithm outputs a ciphertext ct linked to both $\mathbf{x}$ and G .
- $\text{Dec}(\text{mpk}, \text{dk}_{F, \mathbf{y}}, \text{ct}) \rightarrow \mu \text{ or } \perp$: On input the master public key mpk , the decryption key $\text{dk}_{F, \mathbf{y}}$ for $(F \in \mathcal{F}_\lambda, \mathbf{y} \in \mathcal{Y}_\lambda)$ and a ciphertext ct , the Dec algorithm outputs a message μ or a failure symbol $\perp$.

Correctness. For all inputs $\lambda \in \mathbb{N}$, $\mathbf{x} \in \mathcal{X}_\lambda$, $F \in \mathcal{F}_\lambda$, $\mathbf{y} \in \mathcal{Y}_\lambda$, $G \in \mathcal{G}_\lambda$ and $\mu \in \mathcal{M}$, we require the following: if $F(\mathbf{x}) = 1 \wedge G(\mathbf{y}) = 1$,

$$\Pr [\text{Dec}(\text{mpk}, \text{dk}_{F, \mathbf{y}}, \text{Enc}(\text{mpk}, \text{ek}_{\mathbf{x}}, G, \mu)) = \mu] = 1 - \text{negl}(\lambda),$$

where $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$, $\text{dk}_{F, \mathbf{y}} \leftarrow \text{RKGen}(\text{mpk}, \text{msk}, F, \mathbf{y})$, and $\text{ek}_{\mathbf{x}} \leftarrow \text{SKGen}(\text{mpk}, \text{msk}, \mathbf{x})$.

Security Definition. For Π_{AME} , we focus on privacy and authenticity.

Privacy: A malicious receiver, even with adaptive access to the SKGen and RKGen oracles, can only learn a single bit indicating whether a match occurs, with no leakage of the sender's sensitive information (attributes $\mathbf{x}$, policy G). Following the refined definitions in [27, 28], we consider two levels of privacy:

- in the *mismatch* case (referred to as CPA-1-sided privacy), the receiver cannot decrypt the challenge ciphertext and cannot distinguish, except with negligible probability, ciphertexts generated from $(\mu_0, \mathbf{x}_0, G_0)$ or $(\mu_1, \mathbf{x}_1, G_1)$;
- in the *match* case (referred to as CPA-2-sided privacy), the receiver can recover the message (requiring $\mu_0 = \mu_1$) but still cannot distinguish whether the challenge ciphertext is generated from $(\mathbf{x}_0, G_0)$ or $(\mathbf{x}_1, G_1)$.

We further enhance these notions to the CCA setting. For the mismatch case (CCA-1-sided privacy), the adversary is allowed to query a decryption oracle $\mathcal{O}_{\text{Dec}}$; in the match case, however, the CCA notion is trivial, since the adversary is already permitted to decrypt the challenge ciphertext.

Authenticity: Only senders holding legitimate encryption keys $\text{ek}_{\mathbf{x}}$ can produce ciphertexts that are successfully decryptable. A malicious sender cannot forge valid ciphertexts without the corresponding attributes of the legitimate sender, and the success probability of such forgery is negligible.

$\text{EXP}_{\text{AME}, \mathcal{A}}^{\text{ATK-t-Priv}}(1^\lambda)$	$\text{EXP}_{\text{AME}, \mathcal{A}}^{\text{Auth}}(1^\lambda)$
1. $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$ 2. $(\mu_0, \mu_1, \mathbf{x}_0, \mathbf{x}_1, G_0, G_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{SKG}}, \mathcal{O}_{\text{RKG}}, \mathcal{O}_{\text{Dec}}}(1^\lambda, \text{mpk})$ 3. $b \leftarrow \{0, 1\}$ 4. $\text{ek}_{\mathbf{x}_b} \leftarrow \text{SKGen}(\text{mpk}, \text{msk}, \mathbf{x}_b)$ 5. $\text{ct} \leftarrow \text{Enc}(\text{mpk}, \text{ek}_{\mathbf{x}_b}, G_b, \mu_b)$ 6. $b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\text{SKG}}, \mathcal{O}_{\text{RKG}}, \mathcal{O}_{\text{Dec}}}(1^\lambda, \text{ct}, st)$ 7. If $b' = b$, return 1; else return 0.	1. $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$ 2. $(\text{ct}, F, \mathbf{y}) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{SKG}}, \mathcal{O}_{\text{RKG}}}(1^\lambda, \text{mpk})$ 3. $\text{dk}_{F, \mathbf{y}} \leftarrow \text{RKGen}(\text{mpk}, \text{msk}, F, \mathbf{y})$ 4. $\mu = \text{Dec}(\text{mpk}, \text{dk}_{F, \mathbf{y}}, \text{ct})$ 5. If $\forall \mathbf{x} \in \mathcal{Q}_{\mathcal{O}_{\text{SKG}}} : (F(\mathbf{x}) = 0) \wedge (\mu \neq \perp)$, return 1; else return 0.

Fig. 2. Games defining ATK- t -sided privacy (left, where $\text{ATK} \in \{\text{CPA}, \text{CCA}\}$) and authenticity (right) of AME. Oracles $\mathcal{O}_{\text{SKG}}$, $\mathcal{O}_{\text{RKG}}$, and $\mathcal{O}_{\text{Dec}}$ are implemented by $\text{SKGen}(\text{mpk}, \text{msk}, \cdot)$, $\text{RKGen}(\text{mpk}, \text{msk}, \cdot)$, and $\text{Dec}(\text{mpk}, \text{RKGen}(\text{mpk}, \text{msk}, \cdot), \cdot)$, respectively. List $\mathcal{Q}_{\mathcal{O}_{\text{SKG}}}$ stores all attributes that have been queried to $\mathcal{O}_{\text{SKG}}$. Components only needed for the security experiment of CCA-1-sided privacy are written in blue.

Definition 2 (ATK- t -sided Privacy). Let $t \in \{1, 2\}$ and $\text{ATK} \in \{\text{CPA}, \text{CCA}\}$. An AME scheme Π_{AME} is said to satisfy ATK- t -sided privacy if for all valid PPT adversaries $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that

$$\left| \Pr \left[\text{EXP}_{\text{AME}, \mathcal{A}}^{\text{ATK-t-Priv}}(1^\lambda) = 1 \right] - \frac{1}{2} \right| \leq \text{negl}(\lambda),$$

where the experiment $\text{EXP}_{\text{AME}, \mathcal{A}}^{\text{ATK-t-Priv}}(1^\lambda)$ is depicted in Fig. 2. $\mathcal{A}$ is called valid if it satisfies the following restrictions:

1. **Key Query Restriction:** For any $(F, \mathbf{y})$ queried to $\mathcal{O}_{\text{RKG}}$, the following invariant holds:

– **Case $t = 1$ (Mismatch only):**

$$\begin{aligned} & (G_0(\mathbf{y}) = G_1(\mathbf{y}) = 0) \vee (F(\mathbf{x}_0) = F(\mathbf{x}_1) = 0) \\ & \vee (G_0(\mathbf{y}) = F(\mathbf{x}_1) = 0) \vee (G_1(\mathbf{y}) = F(\mathbf{x}_0) = 0). \end{aligned} \quad (2)$$

– **Case $t = 2$ (Mismatch and match):** Either Eq. (2) holds, or

$$(G_0(\mathbf{y}) = G_1(\mathbf{y})) \wedge (F(\mathbf{x}_0) = F(\mathbf{x}_1)) \wedge (\mu_0 = \mu_1). \quad (3)$$

2. **Decryption Query Restriction:** If $\text{ATK} = \text{CCA}$, after the challenge ciphertext ct is released, $\mathcal{A}$ is prohibited from querying $\mathcal{O}_{\text{Dec}}$ on ct with any $(F, \mathbf{y})$ that satisfies the matching condition.

Next, we consider a weaker variant of privacy, namely *selective privacy*. This notion follows the selective security notion commonly used in ABE [12].

Definition 3 (Selective Privacy). We define selective privacy for AME as the same as the above privacy, except that adversary $\mathcal{A}$ is required to submit the challenge pairs $(\mathbf{x}_0, G_0)$ and $(\mathbf{x}_1, G_1)$ to the challenger before the setup phase. We define the corresponding advantage of $\mathcal{A}$ as $\text{Adv}_{\text{AME}, \mathcal{A}}^{\text{Sel-Priv}}(1^\lambda)$. If this advantage value is negligible, the AME scheme is said to satisfy selective privacy.

Remark 1. By adapting the lemma in [10, 12] to AME, any AME scheme that achieves selective privacy can also achieve privacy, although this comes at the cost of an exponential loss in parameters.

Definition 4 (Authenticity). An AME scheme Π_{AME} is said to satisfy authenticity if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that

$$\Pr \left[\text{EXP}_{\text{AME}, \mathcal{A}}^{\text{Auth}}(1^\lambda) = 1 \right] \leq \text{negl}(\lambda),$$

where the experiment $\text{EXP}_{\text{AME}, \mathcal{A}}^{\text{Auth}}(1^\lambda)$ is depicted in Fig. 2.

2.2 Matchmaking Encryption

Definition 5 (Matchmaking Encryption [5, 6]). Let λ be a security parameter, $\mathcal{X}_\lambda$ and $\mathcal{G}_\lambda$ be sender's attribute and policy spaces, $\mathcal{Y}_\lambda$ and $\mathcal{F}_\lambda$ be receiver's attribute and policy spaces, and $\mathcal{M} = \mathcal{M}(\lambda)$ be a message space. An ME scheme consists of a tuple of efficient algorithms Π_{ME} with the following syntax:

- Setup, SKGen, Enc : These are identical to those in AME.
- RKGen(mpk, msk, $\mathbf{y}$) $\rightarrow$ $\text{dk}_{\mathbf{y}}$: On input the master public key mpk, the master secret key msk and an attribute vector $\mathbf{y} \in \mathcal{Y}_\lambda$, the RKGen algorithm outputs a decryption key $\text{dk}_{\mathbf{y}}$.
- PolGen(mpk, msk, F) $\rightarrow$ dk_F : On input the master public key mpk, the master secret key msk and a policy function $F \in \mathcal{F}_\lambda$, the PolGen algorithm outputs a decryption key dk_F .
- Dec(mpk, dk_F , $\text{dk}_{\mathbf{y}}$, ct) $\rightarrow$ μ or $\perp$: On input the master public key mpk, the decryption key dk_F for $F \in \mathcal{F}_\lambda$, the decryption key $\text{dk}_{\mathbf{y}}$ for $\mathbf{y} \in \mathcal{Y}_\lambda$ and a ciphertext ct, the Dec algorithm outputs a message $\mu \in \mathcal{M}$ or $\perp$.

Correctness. For all inputs $\lambda \in \mathbb{N}$, $\mathbf{x} \in \mathcal{X}_\lambda$, $F \in \mathcal{F}_\lambda$, $\mathbf{y} \in \mathcal{Y}_\lambda$, $G \in \mathcal{G}_\lambda$ and $\mu \in \mathcal{M}$, we require the following: if $F(\mathbf{x}) = 1 \wedge G(\mathbf{y}) = 1$,

$$\Pr[\text{Dec}(\text{mpk}, \text{dk}_F, \text{dk}_{\mathbf{y}}, \text{Enc}(\text{mpk}, \text{ek}_{\mathbf{x}}, G, \mu)) = \mu] = 1 - \text{negl}(\lambda).$$

where $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$, $\text{dk}_F \leftarrow \text{PolGen}(\text{mpk}, \text{msk}, F)$, $\text{dk}_{\mathbf{y}} \leftarrow \text{RKGen}(\text{mpk}, \text{msk}, \mathbf{y})$ and $\text{ek}_{\mathbf{x}} \leftarrow \text{SKGen}(\text{mpk}, \text{msk}, \mathbf{x})$.

Security Definition. For Π_{ME} , we focus on privacy and authenticity.

Definition 6 (ATK- t -sided Privacy). Let $t \in [2]$ and $\text{ATK} \in \{\text{CPA}, \text{CCA}\}$. An ME scheme Π_{ME} is said to satisfy ATK- t -sided privacy if for all valid PPT adversary $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that

$$\left| \Pr \left[\text{EXP}_{\text{ME}, \mathcal{A}}^{\text{ATK-}t\text{-Priv}}(1^\lambda) = 1 \right] - \frac{1}{2} \right| \leq \text{negl}(\lambda),$$

$\text{EXP}_{\text{ME}, \mathcal{A}}^{\text{ATK-t-Priv}}(1^\lambda)$	$\text{EXP}_{\text{ME}, \mathcal{A}}^{\text{Auth}}(1^\lambda)$
1. $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$ 2. $(\mu_0, \mu_1, \mathbf{x}_0, \mathbf{x}_1, G_0, G_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{SKG}}, \mathcal{O}_{\text{PG}}, \mathcal{O}_{\text{RKG}}, \mathcal{O}_{\text{Dec}}}(1^\lambda, \text{mpk})$ 3. $b \leftarrow \{0, 1\}$ 4. $\text{ek}_{\mathbf{x}_b} \leftarrow \text{SKGen}(\text{mpk}, \text{msk}, \mathbf{x}_b)$ 5. $\text{ct} \leftarrow \text{Enc}(\text{mpk}, \text{ek}_{\mathbf{x}_b}, G_b, \mu_b)$ 6. $b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\text{SKG}}, \mathcal{O}_{\text{PG}}, \mathcal{O}_{\text{RKG}}, \mathcal{O}_{\text{Dec}}}(1^\lambda, \text{ct}, st)$ 7. If $b' = b$, return 1; else return 0.	1. $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$ 2. $(\text{ct}, F, \mathbf{y}) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{SKG}}, \mathcal{O}_{\text{PG}}, \mathcal{O}_{\text{RKG}}}(1^\lambda, \text{mpk})$ 3. $\text{dk}_F \leftarrow \text{PolGen}(\text{mpk}, \text{msk}, F)$ 4. $\text{dk}_{\mathbf{y}} \leftarrow \text{RKGen}(\text{mpk}, \text{msk}, \mathbf{y})$ 5. $\mu = \text{Dec}(\text{mpk}, \text{dk}_F, \text{dk}_{\mathbf{y}}, \text{ct})$ 6. If $\forall \mathbf{x} \in \mathcal{Q}_{\mathcal{O}_{\text{SKG}}} : (F(\mathbf{x}) = 0) \wedge (\mu \neq \perp)$, return 1; else return 0.

Fig. 3. Games defining ATK- t -sided privacy (left, where $\text{ATK} \in \{\text{CPA}, \text{CCA}\}$) and authenticity (right) of ME. Oracles $\mathcal{O}_{\text{SKG}}$, $\mathcal{O}_{\text{PG}}$, $\mathcal{O}_{\text{RKG}}$, and $\mathcal{O}_{\text{Dec}}$ are implemented by $\text{SKGen}(\text{mpk}, \text{msk}, \cdot)$, $\text{PG}(\text{mpk}, \text{msk}, \cdot)$, $\text{RKGen}(\text{mpk}, \text{msk}, \cdot)$, and $\text{Dec}(\text{mpk}, \text{PG}(\text{mpk}, \text{msk}, \cdot), \text{RKGen}(\text{mpk}, \text{msk}, \cdot), \cdot)$, respectively. List $\mathcal{Q}_{\mathcal{O}_{\text{SKG}}}$ stores all attributes that have been queried to $\mathcal{O}_{\text{SKG}}$. Components only needed for the security experiment of CCA-1-sided privacy are written in blue.

where the experiment $\text{EXP}_{\text{ME}, \mathcal{A}}^{\text{ATK-t-Priv}}(1^\lambda)$ is depicted in Fig. 3. $\mathcal{A}$ is called valid if it satisfies the following restrictions:

1. **Key Query Restriction:** For any F and $\mathbf{y}$ that have been queried to $\mathcal{O}_{\text{PG}}$ and $\mathcal{O}_{\text{RKG}}$, the following invariant holds:
 - **Case $t = 1$ (Mismatch only):**

$$(G_0(\mathbf{y}) = G_1(\mathbf{y}) = 0) \vee (F(\mathbf{x}_0) = F(\mathbf{x}_1) = 0) \vee (G_0(\mathbf{y}) = F(\mathbf{x}_1) = 0) \vee (G_1(\mathbf{y}) = F(\mathbf{x}_0) = 0). \quad (4)$$

- **Case $t = 2$ (Mismatch and match):** Either Eq. (4) holds, or

$$(G_0(\mathbf{y}) = G_1(\mathbf{y})) \wedge (F(\mathbf{x}_0) = F(\mathbf{x}_1)) \wedge (\mu_0 = \mu_1). \quad (5)$$

2. **Decryption Query Restriction:** If $\text{ATK} = \text{CCA}$, after the challenge ciphertext ct is released, $\mathcal{A}$ is prohibited from querying $\mathcal{O}_{\text{Dec}}$ on ct with any F and $\mathbf{y}$ that satisfy the matching condition.

Definition 7 (Selective Privacy). We define selective privacy for ME as the same as the above privacy, except that adversary $\mathcal{A}$ is required to submit the challenge pairs $(\mathbf{x}_0, G_0)$ and $(\mathbf{x}_1, G_1)$ to the challenger before the setup phase. We define the corresponding advantage of $\mathcal{A}$ as $\text{Adv}_{\text{ME}, \mathcal{A}}^{\text{Sel-Priv}}(1^\lambda)$. If this advantage value is negligible, the ME scheme is said to satisfy selective privacy.

Definition 8 (Authenticity). An ME scheme Π_{ME} is said to satisfy authenticity if for any PPT $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that

$$\Pr \left[\text{EXP}_{\text{ME}, \mathcal{A}}^{\text{Auth}}(1^\lambda) = 1 \right] \leq \text{negl}(\lambda),$$

where the experiment $\text{EXP}_{\text{ME}, \mathcal{A}}^{\text{Auth}}(1^\lambda)$ is depicted in Fig. 3.

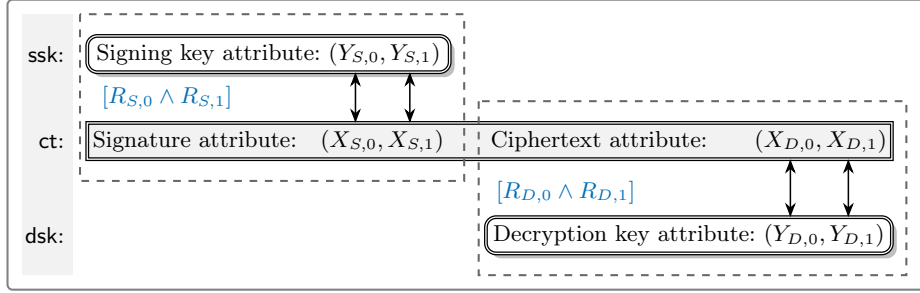


Fig. 4. Relations over multiple attribute sets in A²BSC.

3 Anonymous Attribute-Based Signcryption

3.1 Syntax of A²BSC

Below, we provide the full form of A²BSC, adopting a unified framework designed to capture diverse A²BSC variants within a single generalized model.

Notations for Composite Relations. Before defining the syntax, we introduce a notation for the conjunction of relations. Following the notations in [2, 8], let $R_L : \mathcal{X}_L \times \mathcal{Y}_L \rightarrow \{0, 1\}$ and $R_R : \mathcal{X}_R \times \mathcal{Y}_R \rightarrow \{0, 1\}$ be two relations. We define their *conjunction*, denoted as $[R_L \wedge R_R]$, to be a relation over the product space $(\mathcal{X}_L \times \mathcal{X}_R) \times (\mathcal{Y}_L \times \mathcal{Y}_R)$. Formally, for any inputs $((X_L, X_R), (Y_L, Y_R))$, where $(X_L, X_R) \in \mathcal{X}_L \times \mathcal{X}_R$ and $(Y_L, Y_R) \in \mathcal{Y}_L \times \mathcal{Y}_R$, we have

$$[R_L \wedge R_R]((X_L, X_R), (Y_L, Y_R)) = 1 \text{ iff } R_L(X_L, Y_L) = 1 \wedge R_R(X_R, Y_R) = 1.$$

In our A²BSC, we employ this notation to specify two composite relations:

1. **A signcryption relation** $\mathcal{R}_{\text{sign}} := [R_{S,0} \wedge R_{S,1}]$, determining the validity of generated ciphertexts;
2. **A decryption relation** $\mathcal{R}_{\text{dec}} := [R_{D,0} \wedge R_{D,1}]$, determining the validity of decryption capability.

Remark 2. Observe that this conjunctive formulation could be abstracted into a single, generalized relation $\mathcal{R}$. We opt for an explicit conjunction, as this approach is conceptually intuitive and directly covers the A²BSC variants we focus on in this work.

Definition 9 (Anonymous Attribute-Based Signcryption). Let λ be a security parameter, and $\mathcal{M} = \mathcal{M}(\lambda)$ be a message space. An A²BSC scheme for the relations $\mathcal{R}_{\text{sign}}$ and $\mathcal{R}_{\text{dec}}$ (as defined above and illustrated in Fig. 4) consists of the following efficient algorithms:

- $\text{Setup}(1^\lambda) \rightarrow (\text{mpk}, \text{msk})$: It takes the security parameter λ as input, and outputs the master public key mpk and the master secret key msk .

- $\text{sExtract}(\text{mpk}, \text{msk}, Y_{S,0}, Y_{S,1}) \rightarrow \text{ssk}_{Y_{S,0}, Y_{S,1}}$: It takes (mpk, msk) and a pair of signing key attributes $(Y_{S,0}, Y_{S,1}) \in \mathcal{Y}_{S,0} \times \mathcal{Y}_{S,1}$ as input, and outputs a signing secret key $\text{ssk}_{Y_{S,0}, Y_{S,1}}$.
- $\text{dExtract}(\text{mpk}, \text{msk}, Y_{D,0}, Y_{D,1}) \rightarrow \text{dsk}_{Y_{D,0}, Y_{D,1}}$: It takes (mpk, msk) and a pair of decryption key attributes $(Y_{D,0}, Y_{D,1}) \in \mathcal{Y}_{D,0} \times \mathcal{Y}_{D,1}$ as input, and outputs a decryption secret key $\text{dsk}_{Y_{D,0}, Y_{D,1}}$.
- $\text{Signcrypt}(\text{mpk}, \text{ssk}_{Y_{S,0}, Y_{S,1}}, (X_{S,0}, X_{S,1}), (X_{D,0}, X_{D,1}), \mu) \rightarrow \text{ct or } \perp$: It takes mpk , a signing secret key $\text{ssk}_{Y_{S,0}, Y_{S,1}}$, a pair of signature attribute $(X_{S,0}, X_{S,1}) \in \mathcal{X}_{S,0} \times \mathcal{X}_{S,1}$, a pair of ciphertext attributes $(X_{D,0}, X_{D,1}) \in \mathcal{X}_{D,0} \times \mathcal{X}_{D,1}$ and a message $\mu \in \mathcal{M}$ as input, and outputs a ciphertext ct or $\perp$.
- $\text{Unsigncrypt}(\text{mpk}, \text{dsk}_{Y_{D,0}, Y_{D,1}}, \text{ct}) \rightarrow \mu \text{ or } \perp$: It takes mpk , a decryption secret key $\text{dsk}_{Y_{D,0}, Y_{D,1}}$ and a ciphertext ct as input, and outputs the message μ or $\perp$.

Correctness. An A^2BSC scheme for $\mathcal{R}_{\text{sign}}$ and $\mathcal{R}_{\text{dec}}$ is correct if for all $\lambda \in \mathbb{N}$, messages $\mu \in \mathcal{M}$, and input tuples

$$\begin{aligned} (X_{S,0}, X_{S,1}, X_{D,0}, X_{D,1}) &\in \mathcal{X}_{S,0} \times \mathcal{X}_{S,1} \times \mathcal{X}_{D,0} \times \mathcal{X}_{D,1}, \\ (Y_{S,0}, Y_{S,1}, Y_{D,0}, Y_{D,1}) &\in \mathcal{Y}_{S,0} \times \mathcal{Y}_{S,1} \times \mathcal{Y}_{D,0} \times \mathcal{Y}_{D,1} \end{aligned}$$

satisfying $\mathcal{R}_{\text{sign}}((X_{S,0}, X_{S,1}), (Y_{S,0}, Y_{S,1})) = 1 = \mathcal{R}_{\text{dec}}((X_{D,0}, X_{D,1}), (Y_{D,0}, Y_{D,1}))$, it holds that

$$\Pr \left[\mu = \mu' \mid \begin{array}{l} (\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda); \\ \text{ssk}_{Y_{S,0}, Y_{S,1}} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, Y_{S,0}, Y_{S,1}); \\ \text{dsk}_{Y_{D,0}, Y_{D,1}} \leftarrow \text{dExtract}(\text{mpk}, \text{msk}, Y_{D,0}, Y_{D,1}); \\ \text{ct} \leftarrow \text{Signcrypt}(\text{mpk}, \text{ssk}_{Y_{S,0}, Y_{S,1}}, (X_{S,0}, X_{S,1}), \\ \quad \quad \quad (X_{D,0}, X_{D,1}), \mu); \\ \mu' := \text{Unsigncrypt}(\text{mpk}, \text{dsk}_{Y_{D,0}, Y_{D,1}}, \text{ct}) \end{array} \right] = 1 - \text{negl}(\lambda).$$

3.2 Security Definitions of A^2BSC

In this section, we give three security definitions for A^2BSC : *message confidentiality* (or indistinguishability under selective attribute and adaptive chosen-ciphertext attack: IND-Sel-CCA), *ciphertext unforgeability* (or existential unforgeability under co-selective attribute and adaptive chosen-message attack: EUF-CoSel), and *ciphertext anonymity* (or anonymity under selective attribute: Anon-Sel).

Definition 10 (IND-Sel-CCA). For an A^2BSC scheme Π for $[R_{S,0} \wedge R_{S,1}]$ and $[R_{D,0} \wedge R_{D,1}]$, a message space $\mathcal{M}$, an adversary $\mathcal{A}$ and a challenge $\mathcal{C}$, let us define the IND-Sel-CCA security experiment $\text{EXP}_{\text{A}^2\text{BSC}, \mathcal{A}}^{\text{IND}}(1^\lambda)$ as follows.

1. **Init:** $\mathcal{A}$ declares the challenge attributes $(X_{D,0}^*, X_{D,1}^*)$.
2. **Setup:** $\mathcal{C}$ samples $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$ and sends mpk to $\mathcal{A}$.
3. **Query Phase 1:** Throughout the experiment, $\mathcal{A}$ adaptively performs the following queries in any order, and $\mathcal{C}$ responds to these queries as follows.

- **sExtract** $\mathcal{O}_{sE}(Y_{S,0}, Y_{S,1})$: $\mathcal{A}$ chooses an input $(Y_{S,0}, Y_{S,1})$, $\mathcal{C}$ replies with $\text{sSk}_{Y_{S,0}, Y_{S,1}} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, Y_{S,0}, Y_{S,1})$.
- **dExtract** $\mathcal{O}_{dE}(Y_{D,0}, Y_{D,1})$: $\mathcal{A}$ chooses an input $(Y_{D,0}, Y_{D,1})$, it outputs $\perp$ if $R_{D,0}(X_{D,0}^*, Y_{D,0}) = R_{D,1}(X_{D,1}^*, Y_{D,1}) = 1$. Otherwise, $\mathcal{C}$ replies with $\text{dSk}_{Y_{D,0}, Y_{D,1}} \leftarrow \text{dExtract}(\text{mpk}, \text{msk}, Y_{D,0}, Y_{D,1})$.
- **Signcrypt** $\mathcal{O}_{SC}((X_{S,0}, X_{S,1}), (X_{D,0}, X_{D,1}), \mu)$: $\mathcal{C}$ samples a set of signing key attributes $(Y_{S,0}, Y_{S,1})$ s.t.

$$R_{S,0}(X_{S,0}, Y_{S,0}) = R_{S,1}(X_{S,1}, Y_{S,1}) = 1.$$

Then, $\mathcal{C}$ computes $\text{sSk}_{Y_{S,0}, Y_{S,1}} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, Y_{S,0}, Y_{S,1})$ and replies to $\mathcal{A}$ with $\text{ct} \leftarrow \text{Signcrypt}(\text{mpk}, \text{sSk}_{Y_{S,0}, Y_{S,1}}, (X_{S,0}, X_{S,1}), (X_{D,0}, X_{D,1}), \mu)$.

- **Unsigncrypt** $\mathcal{O}_{USC}(\text{ct}, (Y_{D,0}, Y_{D,1}))$: $\mathcal{A}$ chooses an input $(\text{ct}, Y_{D,0}, Y_{D,1})$, $\mathcal{C}$ computes $\text{dSk}_{Y_{D,0}, Y_{D,1}} \leftarrow \text{dExtract}(\text{mpk}, \text{msk}, Y_{D,0}, Y_{D,1})$ and replies to $\mathcal{A}$ with the output of $\text{Unsigncrypt}(\text{mpk}, \text{dSk}_{Y_{D,0}, Y_{D,1}}, \text{ct})$.
4. **Challenge**: $\mathcal{A}$ issues a message pair $(\mu_0, \mu_1) \in \mathcal{M}^2$ and a set of signature attributes $(X_{S,0}^*, X_{S,1}^*)$. $\mathcal{C}$ samples $\alpha \leftarrow \{0, 1\}$ and a set of signing key attribute $(Y_{S,0}^*, Y_{S,1}^*)$ s.t. $R_{S,0}(X_{S,0}^*, Y_{S,0}^*) = R_{S,1}(X_{S,1}^*, Y_{S,1}^*) = 1$, computes $\text{sSk}_{Y_{S,0}^*, Y_{S,1}^*} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, Y_{S,0}^*, Y_{S,1}^*)$. Finally, $\mathcal{C}$ replies to $\mathcal{A}$ with

$$\text{ct}^* \leftarrow \text{Signcrypt}(\text{mpk}, \text{sSk}_{Y_{S,0}^*, Y_{S,1}^*}, (X_{S,0}^*, X_{S,1}^*), (X_{D,0}^*, X_{D,1}^*), \mu_\alpha).$$

5. **Query Phase 2**: $\mathcal{A}$ can issue queries analogous to those in Query Phase 1, except for the following restricted queries: $\mathcal{O}_{USC}(\text{ct}^*, (Y_{D,0}, Y_{D,1}))$ for any $(Y_{D,0}, Y_{D,1})$ with $R_{D,0}(X_{D,0}^*, Y_{D,0}) = R_{D,1}(X_{D,1}^*, Y_{D,1}) = 1$.
6. **Output Phase**: The experiment output is a guessed bit α' generated by $\mathcal{A}$.

In the above experiment, the advantage $\text{Adv}_{II, \mathcal{A}}^{\text{IND}}(1^\lambda)$ of $\mathcal{A}$ is defined as

$$\text{Adv}_{II, \mathcal{A}}^{\text{IND}}(1^\lambda) := \left| \Pr \left[\text{EXP}_{\text{A}^2\text{BSC}, \mathcal{A}}^{\text{IND}}(1^\lambda) = \alpha \right] - \frac{1}{2} \right|.$$

The A^2BSC scheme II is said to satisfy IND-Sel-CCA security if for any PPT $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that $\text{Adv}_{II, \mathcal{A}}^{\text{IND}}(1^\lambda) \leq \text{negl}(\lambda)$.

EUF-CoSel [38] requires the adversary to declare the attribute information for key queries before the release of public parameters, while the challenge attribute is adaptively chosen. This differs from selective unforgeability [52], which requires the challenge attribute to be declared before the release of public parameters.

Definition 11 (EUF-CoSel). For an A^2BSC scheme II for relations $[R_{S,0} \wedge R_{S,1}]$ and $[R_{D,0} \wedge R_{D,1}]$, a message space $\mathcal{M}$, an adversary $\mathcal{A}$ and a challenge $\mathcal{C}$, let us define EUF-CoSel security game as follows.

1. **Init**: $\mathcal{A}$ sends a set of signing key attributes $\{Y_{S,0}^{(i)}, Y_{S,1}^{(i)}\}_{i \in [N]}$ and $\mathcal{U} \subseteq [N]$ to the challenger, where $N = \text{poly}(\lambda)$.
2. **Setup**: $\mathcal{C}$ samples $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$ and sends mpk to $\mathcal{A}$.
3. **Query Phase**: Throughout the experiment, $\mathcal{A}$ adaptively performs the following queries in any order, and $\mathcal{C}$ responds to these queries as follows.

- **sExtract** $\mathcal{O}_{\text{SE}}(Y_{S,0}, Y_{S,1})$: Given an input $(Y_{S,0}, Y_{S,1}) \in \{Y_{S,0}^{(i)}, Y_{S,1}^{(i)}\}_{i \in \mathcal{U}}$, $\mathcal{C}$ replies to $\mathcal{A}$ with $\text{ssk}_{Y_{S,0}, Y_{S,1}} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, Y_{S,0}, Y_{S,1})$.
 - **dExtract** $\mathcal{O}_{\text{dE}}(Y_{D,0}, Y_{D,1})$: Given an input $(Y_{D,0}, Y_{D,1})$, $\mathcal{C}$ replies with $\text{dsk}_{Y_{D,0}, Y_{D,1}} \leftarrow \text{dExtract}(\text{mpk}, \text{msk}, Y_{D,0}, Y_{D,1})$.
 - **Signcrypt** $\mathcal{O}_{\text{SC}}((X_{S,0}, X_{S,1}), (X_{D,0}, X_{D,1}), \mu)$: $\mathcal{C}$ responds to such a query as in Definition 10.
 - **Unsigncrypt** $\mathcal{O}_{\text{USC}}(\text{ct}, (Y_{D,0}, Y_{D,1}))$: $\mathcal{C}$ responds to such a query as in Definition 10.
4. **Forgery**: $\mathcal{A}$ returns a ciphertext ct^* with the signature attributes $(X_{S,0}^*, X_{S,1}^*)$ for message $\mu^* \in \mathcal{M}$, and a decryption key attribute $(Y_{D,0}^*, Y_{D,1}^*)$.

$\mathcal{A}$ wins the game if

- a) $[R_{S,0} \wedge R_{S,1}]((X_{S,0}^*, X_{S,1}^*), (Y_{S,0}^{(i)}, Y_{S,1}^{(i)})) \neq 1$ for any $i \in \mathcal{U}$;
- b) $\mathcal{A}$ did not issue a query of the form $\mathcal{O}_{\text{SC}}((X_{S,0}^*, X_{S,1}^*), (X_{D,0}, X_{D,1}), \mu^*)$ for any $(X_{D,0}, X_{D,1})$;
- c) $\mu^* \leftarrow \text{Unsigncrypt}(\text{mpk}, \text{dsk}_{Y_{D,0}^*, Y_{D,1}^*}, \text{ct}^*)$, where $\mu^* \neq \perp$ and $\text{dsk}_{Y_{D,0}^*, Y_{D,1}^*} \leftarrow \text{dExtract}(\text{mpk}, \text{msk}, Y_{D,0}^*, Y_{D,1}^*)$.

In the above experiment, the advantage of $\mathcal{A}$ is defined as

$$\text{Adv}_{\Pi, \mathcal{A}}^{\text{EUF}}(1^\lambda) := \Pr[\mathcal{A} \text{ wins}].$$

The A^2BSC scheme Π is said to satisfy EUF-CoSel security if for any PPT $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that $\text{Adv}_{\Pi, \mathcal{A}}^{\text{EUF}}(1^\lambda) = \text{negl}(\lambda)$.

Following the work of [38], we consider a weaker security notion that is similar to EUF-CoSel, except that it prohibits any signcryption query $\mathcal{O}_{\text{SC}}$.

Definition 12 (No-Signing EUF-CoSel). No-signing EUF-CoSel game modifies the EUF-CoSel game by adding the following restrictions: $\mathcal{A}$ is prohibited from making any signcryption queries and must have $\mathcal{U} \neq \emptyset$. The winning condition and the definition of advantage are unchanged. Note that this trivially satisfies the restriction in Item b). Hence, an ABSC scheme achieves no-signing EUF-CoSel security if any PPT $\mathcal{A}$ has only negligible advantage in this game.

Remark 3. Following the general transformation technique in [38], we can derive EUF-CoSel security from its no-signing-query variant. Thus, it suffices to prove that our scheme satisfies no-signing EUF-CoSel security.

Next, we consider two levels of Anon-Sel. In particular,

- 1-sided Anon-Sel: outside observers cannot infer any information regarding the message or the attributes associated with the challenge ciphertext;
- 2-sided Anon-Sel: even authorized users capable of decrypting the challenge ciphertext learn nothing other than the message and signature attributes.

Definition 13 (t -sided Anon-Sel). Let $t \in [2]$. For an A^2BSC scheme Π for relations $[R_{S,0} \wedge R_{S,1}]$ and $[R_{D,0} \wedge R_{D,1}]$, a message space $\mathcal{M}$, an adversary $\mathcal{A}$ and a challenge $\mathcal{C}$, let us define t -sided Anon-Sel security experiment $\text{EXP}_{\text{A}^2\text{BSC}, \mathcal{A}}^{t\text{-AnonSel}}(1^\lambda)$ as follows.

1. **Init:** $\mathcal{A}$ declares the challenge attributes $(X_{D,0}^{(0)}, X_{D,1}^{(0)}), (X_{D,0}^{(1)}, X_{D,1}^{(1)})$.
2. **Setup:** $\mathcal{C}$ samples $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$ and sends mpk to $\mathcal{A}$.
3. **Query Phase 1:** $\mathcal{A}$ is allowed to make adaptive queries as in Definition 10, except that queries to $\mathcal{O}_{\text{dE}}$ are subject to finer restrictions, detailed later.
4. **Challenge:** $\mathcal{A}$ issues a message μ , a set of signature attributes $(X_{S,0}^*, X_{S,1}^*)$, a set of signing key attributes $\{(Y_{S,0}^{(0)}, Y_{S,1}^{(0)}), (Y_{S,0}^{(1)}, Y_{S,1}^{(1)})\}$ such that

$$(R_{S,0}(X_{S,0}^*, Y_{S,0}^{(i)}) \wedge R_{S,1}(X_{S,1}^*, Y_{S,1}^{(i)})) = 1 \text{ for } i \in \{0, 1\}.$$

$\mathcal{C}$ samples $\alpha \leftarrow \{0, 1\}$ and computes

$$\text{ct}^* \leftarrow \text{Signcrypt}(\text{mpk}, \text{ssk}_{Y_{S,0}^{(\alpha)}, Y_{S,1}^{(\alpha)}}, (X_{S,0}^*, X_{S,1}^*), (X_{D,0}^{(\alpha)}, X_{D,1}^{(\alpha)}), \mu),$$

where $\text{ssk}_{Y_{S,0}^{(\alpha)}, Y_{S,1}^{(\alpha)}} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, (Y_{S,0}^{(\alpha)}, Y_{S,1}^{(\alpha)}))$. $\mathcal{C}$ sends ct^* to $\mathcal{A}$.

5. **Query Phase 2:** $\mathcal{A}$ can issue queries analogous to those in Query Phase 1.
6. **Output Phase:** The experiment output is a guessed bit α' generated by $\mathcal{A}$.

In the above experiment, the advantage $\text{Adv}_{\Pi, \mathcal{A}}^{\text{Anon}}(1^\lambda)$ of $\mathcal{A}$ is defined as

$$\text{Adv}_{\Pi, \mathcal{A}}^{\text{Anon}}(1^\lambda) := \left| \Pr \left[\text{EXP}_{\text{A}^2\text{BSC}, \mathcal{A}}^{\text{t-AnonSel}}(1^\lambda) = \alpha \right] - \frac{1}{2} \right|.$$

The A^2BSC scheme Π is said to satisfy t -sided Anon-Sel security if for all valid PPT $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ s.t. $\text{Adv}_{\Pi, \mathcal{A}}^{\text{Anon}}(1^\lambda) \leq \text{negl}(\lambda)$. $\mathcal{A}$ is called valid if the following invariant holds for any $(Y_{D,0}, Y_{D,1})$ queried to $\mathcal{O}_{\text{dE}}$:

1. **Condition for failure to decrypt ct^* (Case $t = 1$):**

$$\begin{aligned} & (R_{D,0}(X_{D,0}^{(0)}, Y_{D,0}) = R_{D,0}(X_{D,0}^{(1)}, Y_{D,0}) = 0) \\ & \vee (R_{D,1}(X_{D,1}^{(0)}, Y_{D,1}) = R_{D,1}(X_{D,1}^{(1)}, Y_{D,1}) = 0) \\ & \vee (R_{D,0}(X_{D,0}^{(0)}, Y_{D,0}) = R_{D,1}(X_{D,1}^{(1)}, Y_{D,1}) = 0) \\ & \vee (R_{D,0}(X_{D,0}^{(1)}, Y_{D,0}) = R_{D,1}(X_{D,1}^{(0)}, Y_{D,1}) = 0). \end{aligned} \tag{6}$$

2. **Condition for successfully decrypting ct^* (Case $t = 2$):** Either Eq. (6) holds or

$$(R_{D,0}(X_{D,0}^{(0)}, Y_{D,0}) = R_{D,0}(X_{D,0}^{(1)}, Y_{D,0})) \wedge (R_{D,1}(X_{D,1}^{(0)}, Y_{D,1}) = R_{D,1}(X_{D,1}^{(1)}, Y_{D,1})).$$

3.3 Various Variants of A^2BSC

Here, we give various variants of A^2BSC by specifying the relations for Definition 9, namely key-policy A^2BSC (KP- A^2BSC), ciphertext-policy A^2BSC (CP- A^2BSC), dual policy A^2BSC (DP- A^2BSC), and Special A^2BSC .

KP- A^2BSC for Circuits. To define KP- A^2BSC for circuits considered in this paper, we set

- $\mathcal{X}_{D,0} = \{0,1\}^{\ell_0}$ and $\mathcal{Y}_{D,0}$ as a circuit class $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_0 := \ell_0(\lambda)$ and depth $d_0 := d_0(\lambda)$.
- $\mathcal{X}_{D,1}$ as a circuit class $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda \in \mathbb{N}}$ which consists of circuits that output 1 for any input of length $\ell_1 := \ell_1(\lambda)$, and $\mathcal{Y}_{D,1} = \{0,1\}^{\ell_1}$.
- $\mathcal{X}_{S,0} = \{0,1\}^{\ell_a}$ and $\mathcal{Y}_{S,0}$ as a circuit class $\mathcal{V} = \{\mathcal{V}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_a := \ell_a(\lambda)$ and depth $d_a := d_a(\lambda)$.
- $\mathcal{X}_{S,1}$ as a circuit class $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ which consists of circuits that output 1 for any input of length $\ell_b := \ell_b(\lambda)$, and $\mathcal{Y}_{S,1} = \{0,1\}^{\ell_b}$.

For $(\mathbf{x} \in \{0,1\}^{\ell_0}, F \in \mathcal{F}), (\mathbf{y} \in \{0,1\}^{\ell_1}, G \in \mathcal{G}), (\mathbf{v} \in \{0,1\}^{\ell_a}, V \in \mathcal{V}), (\mathbf{z} \in \{0,1\}^{\ell_b}, C \in \mathcal{C})$, we define the relations

$$[R_{D,0} \wedge R_{D,1}]((\mathbf{x}, F), (G, \mathbf{y})) = F(\mathbf{x}) \text{ and } [R_{S,0} \wedge R_{S,1}]((\mathbf{v}, V), (C, \mathbf{z})) = V(\mathbf{v}).$$

Here, we omit the relations (depicted in gray) that always hold, for simplicity.

CP-A²BSC for Circuits. To define CP-A²BSC for circuits considered in this paper, we set

- $\mathcal{X}_{D,0} = \{0,1\}^{\ell_0}$, and $\mathcal{Y}_{D,0}$ as a circuit class $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ which consists of circuits that output 1 for any input of length $\ell_0 := \ell_0(\lambda)$.
- $\mathcal{X}_{D,1}$ as a circuit class $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_1 := \ell_1(\lambda)$ and depth $d_1 := d_1(\lambda)$, and $\mathcal{Y}_{D,1} = \{0,1\}^{\ell_1}$.
- $\mathcal{X}_{S,0} = \{0,1\}^{\ell_a}$ and $\mathcal{Y}_{S,0}$ as a circuit class $\mathcal{V} = \{\mathcal{V}_\lambda\}_{\lambda \in \mathbb{N}}$ which consists of circuits that output 1 for any input of length $\ell_a := \ell_a(\lambda)$.
- $\mathcal{X}_{S,1}$ as a circuit class $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_b := \ell_b(\lambda)$ and depth $d_b := d_b(\lambda)$, and $\mathcal{Y}_{S,1} = \{0,1\}^{\ell_b}$.

For $(\mathbf{x} \in \{0,1\}^{\ell_0}, F \in \mathcal{F}), (\mathbf{y} \in \{0,1\}^{\ell_1}, G \in \mathcal{G}), (\mathbf{v} \in \{0,1\}^{\ell_a}, V \in \mathcal{V}), (\mathbf{z} \in \{0,1\}^{\ell_b}, C \in \mathcal{C})$, we define the relations

$$[R_{D,0} \wedge R_{D,1}]((\mathbf{x}, F), (G, \mathbf{y})) = G(\mathbf{y}) \text{ and } [R_{S,0} \wedge R_{S,1}]((\mathbf{v}, V), (C, \mathbf{z})) = C(\mathbf{z}).$$

Here, we omit the relations (depicted in gray) that always hold, for simplicity.

DP-A²BSC for Circuits. To define DP-A²BSC for circuits considered in this paper, we set

- $\mathcal{X}_{D,0} = \{0,1\}^{\ell_0}$ and $\mathcal{Y}_{D,0}$ as a circuit class $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_0 := \ell_0(\lambda)$ and depth $d_0 := d_0(\lambda)$.
- $\mathcal{X}_{D,1}$ as a circuit class $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_1 := \ell_1(\lambda)$ and depth $d_1 := d_1(\lambda)$, and $\mathcal{Y}_{D,1} = \{0,1\}^{\ell_1}$.
- $\mathcal{X}_{S,0} = \{0,1\}^{\ell_a}$ and $\mathcal{Y}_{S,0}$ as a circuit class $\mathcal{V} = \{\mathcal{V}_\lambda\}_{\lambda \in \mathbb{N}}$ which consists of circuits that output 1 for any input of length $\ell_a := \ell_a(\lambda)$.
- $\mathcal{X}_{S,1}$ as a circuit class $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_b := \ell_b(\lambda)$ and depth $d_b := d_b(\lambda)$, and $\mathcal{Y}_{S,1} = \{0,1\}^{\ell_b}$.

For $(\mathbf{x} \in \{0,1\}^{\ell_0}, F \in \mathcal{F}), (\mathbf{y} \in \{0,1\}^{\ell_1}, G \in \mathcal{G}), (\mathbf{v} \in \{0,1\}^{\ell_a}, V \in \mathcal{V}), (\mathbf{z} \in \{0,1\}^{\ell_b}, C \in \mathcal{C})$, we define the relations

$$\begin{aligned} [R_{D,0} \wedge R_{D,1}]((\mathbf{x}, F), (G, \mathbf{y})) &= F(\mathbf{x}) \wedge G(\mathbf{y}), \\ [R_{S,0} \wedge R_{S,1}]((\mathbf{v}, V), (C, \mathbf{z})) &= C(\mathbf{z}). \end{aligned}$$

Here, we omit the relations (depicted in gray) that always hold, for simplicity.

Special A²BSC for Circuits. We consider a signing secret key with hierarchical structure for DP-A²BSC. Specifically, the signing key attribute $Y_{S,1} \in \{0,1\}^{\ell_b}$ is partitioned into a base layer $Y_{S,1,0} \in \{0,1\}^{\ell_{b,0}}$ and an extension layer $Y_{S,1,1} \in \{0,1\}^{\ell_{b,1}}$, where $\ell_b = \ell_{b,0} + \ell_{b,1}$. In Special A²BSC, the signcryptor first obtains a $\text{ssk}_{Y_{S,1,0}} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, Y_{S,1,0})$ from the authority. Subsequently, the signcryptor can locally derive a complete signing secret key $\text{ssk}_{Y_{S,1,0} \| Y_{S,1,1}}$ for any $Y_{S,1,1} \in \{0,1\}^{\ell_{b,1}}$. Using $\text{ssk}_{Y_{S,1,0} \| Y_{S,1,1}}$, the signcryptor can produce valid ciphertexts w.r.t. any $X_{S,1} \in \mathcal{X}_{S,1}$ (such that $R_{S,1}(X_{S,1}, Y_{S,1,0} \| Y_{S,1,1}) = 1$). Essentially, this hierarchy endows the signcryptor with delegation capabilities, allowing for the arbitrary extension of the remaining attribute components to generate signatures for $X_{S,1}$. The hierarchical algorithm is defined as follows.

- $\text{Derive}(\text{mpk}, \text{ssk}_{Y_{S,1,0}}, Y_{S,1,1}) \rightarrow \text{ssk}_{Y_{S,1,0} \| Y_{S,1,1}}$: On input the master public key mpk , a partial signing secret key $\text{ssk}_{Y_{S,1,0}}$, and an attribute $Y_{S,1,1}$, the algorithm outputs a complete signing secret key $\text{ssk}_{Y_{S,1,0} \| Y_{S,1,1}}$.

To define Special A²BSC for circuits considered in this paper, we set

- $\mathcal{X}_{D,0} = \{0,1\}^{\ell_0}$ and $\mathcal{Y}_{D,0}$ as a circuit class $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_0 := \ell_0(\lambda)$ and depth $d_0 := d_0(\lambda)$.
- $\mathcal{X}_{D,1}$ as a circuit class $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_1 := \ell_1(\lambda)$ and depth $d_1 := d_1(\lambda)$, and $\mathcal{Y}_{D,1} = \{0,1\}^{\ell_1}$.
- $\mathcal{X}_{S,0} = \{0,1\}^{\ell_a}$ and $\mathcal{Y}_{S,0}$ as a circuit class $\mathcal{V} = \{\mathcal{V}_\lambda\}_{\lambda \in \mathbb{N}}$ which consists of circuits that output 1 for any input of length $\ell_a := \ell_a(\lambda)$.
- $\mathcal{X}_{S,1}$ as a circuit class $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ that consists of circuits with input length $\ell_b := \ell_b(\lambda)$ and depth $d_b := d_b(\lambda)$, and $\mathcal{Y}_{S,1} = \{0,1\}^{\ell_b}$ where $\ell_b = \ell_{b,0} + \ell_{b,1}$.

For $(\mathbf{x} \in \{0,1\}^{\ell_0}, F \in \mathcal{F}), (\mathbf{y} \in \{0,1\}^{\ell_1}, G \in \mathcal{G}), ([\mathbf{z}_0 \| \mathbf{z}_1] \in \{0,1\}^{\ell_b}, C \in \mathcal{C})$, we define the relations

$$[R_{D,0} \wedge R_{D,1}]((\mathbf{x}, F), (G, \mathbf{y})) = F(\mathbf{x}) \wedge G(\mathbf{y}) \text{ and } R_{S,1}(C, \mathbf{z}_0 \| \mathbf{z}_1) = C(\mathbf{z}_0 \| \mathbf{z}_1).$$

The security definition of Special A²BSC is identical to that of DP-A²BSC, except for the following two aspects:

- For a sExtract query, the input is $\mathbf{z}_0 \in \{0,1\}^{\ell_{b,0}}$, and the output is $\text{ssk}_{\mathbf{z}_0} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, \mathbf{z}_0)$.
- For a Signcrypt query, the input is $(C \in \mathcal{C}, \mathbf{x} \in \{0,1\}^{\ell_0}, G \in \mathcal{G}, \mu \in \mathcal{M})$. It samples a set of signing key attribute $[\mathbf{z}_0 \| \mathbf{z}_1]$ such that $C(\mathbf{z}_0 \| \mathbf{z}_1) = 1$. The output is $\text{ct} \leftarrow \text{Signcrypt}(\text{mpk}, \text{ssk}_{\mathbf{z}_0 \| \mathbf{z}_1}, C, (\mathbf{x}, G), \mu)$, where $\text{ssk}_{\mathbf{z}_0 \| \mathbf{z}_1} \leftarrow \text{Derive}(\text{mpk}, \text{ssk}_{\mathbf{z}_0}, \mathbf{z}_1)$ and $\text{ssk}_{\mathbf{z}_0} \leftarrow \text{sExtract}(\text{mpk}, \text{msk}, \mathbf{z}_0)$.

Remark 4. In the above model, the adversary does not need to query the complete signing secret key, as it can derive the complete signing secret key by itself from the result of the `sExtract` query.

4 Anonymous Attribute-Based Signcryption for Circuits from Lattices

In this section, we present a **Special A²BSC** construction based on the lattice assumptions **BASIS** and succinct **LWE**. Then, we show how to upgrade it to **Special A²BSC** with separable decryption keys. Using these lattice-based constructions together with appropriate building blocks, we can instantiate the generic constructions of Sect. 5 from lattice assumptions.

4.1 High-Level Overview

In this section, we provide a high-level overview for constructing **Special A²BSC** from lattices. Our construction takes the **ABS** scheme from [38] and the **ABE** schemes from [58] as the starting points.

Hierarchical ABS. In **ABS** [38], the signing key for attribute $\mathbf{z}_0 \in \{0, 1\}^{\ell_0}$ is a low-norm matrix $\mathbf{R}_L$ such that $\mathbf{A}_s \cdot \mathbf{R}_L = \mathbf{B}_{\ell_0} - \mathbf{z}_0 \otimes \mathbf{G}$. For a policy C such that $C(\mathbf{z}_0) = 1$, we have $\mathbf{B}_C = \mathbf{A}_s \mathbf{R}_L \cdot \mathbf{H}_{C, \mathbf{z}_0} + \mathbf{G}$, where $\mathbf{B}_C$ and $\mathbf{H}_{C, \mathbf{z}_0}$ are calculated from lattice evaluation (Lemma 4). It then computes $\mathbf{T}_{[\mathbf{A}_s \| \mathbf{B}_C]}$ for $[\mathbf{A}_s \| \mathbf{B}_C]$ from $\mathbf{R}_L \mathbf{H}_{C, \mathbf{z}_0}$ via `ExtendLeft` (Lemma 3), thereby generating a signature under policy C . We observe that, by leveraging function commitments as in [59], one can construct $\mathbf{A}_s \cdot \mathbf{R}_R = \mathbf{B}_{\ell_1} - \mathbf{z}_1 \otimes \mathbf{G}$ for any $\mathbf{z}_1 \in \{0, 1\}^{\ell_1}$, where $\mathbf{R}_R, \mathbf{B}_{\ell_1}$ are constructed from the state and commitment of function commitments. By concatenating these, we obtain

$$\mathbf{A}_s \cdot [\mathbf{R}_L \| \mathbf{R}_R] = [\mathbf{B}_{\ell_0} \| \mathbf{B}_{\ell_1}] - [\mathbf{z}_0 \| \mathbf{z}_1] \otimes \mathbf{G}.$$

For a policy function C with an input length of $\ell_0 + \ell_1$, if $C(\mathbf{z}_0 \| \mathbf{z}_1) = 1$, we similarly compute $\mathbf{T}_{[\mathbf{A}_s \| \mathbf{B}_C]}$ from $[\mathbf{R}_L \| \mathbf{R}_R] \mathbf{H}_{C, [\mathbf{z}_0 \| \mathbf{z}_1]}$ via `ExtendLeft`. It then generates a valid signature for C . This realizes a hierarchical signing key structure: the *first layer* is issued by the authority for partial attributes $\mathbf{z}_0$, while the user can further delegate keys for any attribute $\mathbf{z}_1$ (*second layer*). Thus, the user can generate valid signatures under policy C if $C(\mathbf{z}_0 \| \mathbf{z}_1) = 1$.

Dual-Policy ABE. We adapt the **ABE** schemes from [58], which provide both **KP-ABE** and **CP-ABE** schemes for circuits from $(2m^2, \tau_0)$ -succinct **LWE**, a falsifiable assumption of recent interest. We conveniently fuse **KP-ABE** and **CP-ABE** to construct a dual-policy **ABE**,⁴ which is enabled by the algebraic consistency of their keys.

⁴ Distinct from the Boolean logic used in **ABS**, we follow the convention of lattice-based **ABE** where $F(\mathbf{x}) = G(\mathbf{y}) = 0$ implies that the decryption condition is satisfied.

Special A²BSC. In rough terms, our A²BSC framework follows the design principles of the CHK paradigm [11, 18, 62], achieving CCA security via one-time signatures (OTS) and the delegability of the above dual-policy ABE. Meanwhile, we use the above hierarchical ABS to sign both the OTS verification key and the message. Finally, to achieve ciphertext anonymity, we wrap the decryption mechanism of ABE using LWE-based LO (Definition 19) [33]. The security of LO ensures that the ciphertext-related attribute and policy information in the dual-policy ABE remains hidden. This yields our construction satisfying 1-sided Anon-Sel (Definition 13). Note that achieving 2-sided Anon-Sel for A²BSC is beyond the reach of LO, as this stronger security implies $i\mathcal{O}$ [27, 28].⁵

4.2 Special A²BSC for Circuits from Lattices

Building Blocks. Our construction requires the following basic building blocks.

- An OTS scheme $\text{OTS} = (\text{OTS.KGen}, \text{OTS.Sign}, \text{OTS.Vrfy})$, where the verification key ovk output by OTS.KGen satisfies $\text{ovk} \in \{0, 1\}^{L-|\mu|}$ for $L = L(\lambda)$.
- A LO scheme $\text{LO} = (\text{LO.Obf}, \text{LO.Eval})$ with lock space $\{0, 1\}^\kappa$ and input $\{0, 1\}^{u+|\mu|}$, where $\kappa = \kappa(\lambda)$, and $u = u(\lambda)$ denotes the upper-bound on the length of the signature that is generated by Signcrypt .

For simplicity, we assume the circuit classes $\mathcal{C}, \mathcal{G}$ are identical to $\mathcal{F} = \{F : \{0, 1\}^\ell \rightarrow \{0, 1\}\}$, and set its circuit depth to d , where $\ell = \ell_0 + \ell_1$. We construct a Special A²BSC scheme for circuit classes $\mathcal{C}, \mathcal{F}, \mathcal{G}$ as follows:

- **Setup**(1^λ): Define the universe of signing attributes as $\mathcal{L} = \{\mathbf{z}_0^{(i)}\}_{i \in [N]}$, where $\mathbf{z}_0^{(i)} \in \{0, 1\}^{\ell_0}$ and $N = N(\lambda)$. Sample

$$\begin{aligned} (\mathbf{A}_r, \mathbf{T}_{\mathbf{A}_r}) &\leftarrow \text{TrapGen}(1^n, 1^m, q), \mathbf{W} \leftarrow \mathbb{Z}_q^{2m^2 n \times m}, \\ \mathbf{T} &\leftarrow \text{SamplePre}([\mathbf{I}_{2m^2} \otimes \mathbf{A}_r \| \mathbf{W}], \mathbf{I}_{2m^2} \otimes \mathbf{T}_{\mathbf{A}_r}, \mathbf{I}_{2m^2} \otimes \mathbf{G}, \tau_0), \\ \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3 &\leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{P} \leftarrow \mathbb{Z}_q^{n \times \kappa}. \end{aligned}$$

Pick $\vec{\mathbf{B}} := [\mathbf{B}_{\ell_0}^{(1)} \| \dots \| \mathbf{B}_{\ell_0}^{(N)}] \leftarrow \mathbb{Z}_q^{n \times m' \ell_0 N}$, $\vec{\mathbf{M}} := [\mathbf{M}_0 \| \dots \| \mathbf{M}_L] \leftarrow \mathbb{Z}_q^{n \times m' (L+1)}$. Sample $(\mathbf{A}_s, \mathbf{T}_{\mathbf{A}_s}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$, an invertible matrix $\mathbf{V}_i \leftarrow \mathbb{Z}_q^{n \times n}$ for $i \in [\ell_1]$. It then computes $\mathbf{T}'_i = \mathbf{T}_{\mathbf{A}_s} \mathbf{G}^{-1} (\mathbf{V}_i^{-1} \mathbf{G}) \in \mathbb{Z}_q^{m \times m'}$ for $i \in [\ell_1]$ and samples $\mathbf{T}' \leftarrow \text{SamplePre}(\mathbf{C}_{\ell_1}, \mathbf{T}_{\mathbf{C}_{\ell_1}}, \mathbf{G}_{n\ell_1}, \tau_0)$ where

$$\mathbf{C}_{\ell_1} = \left[\text{diag}(\mathbf{V}_1 \mathbf{A}_s, \dots, \mathbf{V}_{\ell_1} \mathbf{A}_s) \| -\mathbf{1}_{\ell_1}^\top \otimes \mathbf{G} \right], \mathbf{T}_{\mathbf{C}_{\ell_1}} = \left[\frac{\text{diag}(\mathbf{T}'_1, \dots, \mathbf{T}'_{\ell_1})}{\mathbf{0}^{m' \times m' \ell_1}} \right].$$

Output $\text{mpk} := (\text{pp}_{\text{com}}, \mathbf{A}_s, \{\mathbf{A}_i\}_{i \in [3]}, \vec{\mathbf{B}}, \vec{\mathbf{M}}, \mathbf{P}, \mathbf{T}', \{\mathbf{V}_i\}_{i \in [\ell_1]})$ and $\text{msk} := (\mathbf{T}_{\mathbf{A}_s}, \mathbf{T}_{\mathbf{A}_r})$, where $\text{pp}_{\text{com}} := (\mathbf{A}_r, \mathbf{W}, \mathbf{T})$.

⁵ Generally, ABE decryption algorithm can be modeled as a circuit C_{dec} with $(\mathbf{x}, G)$ hardwired. Meanwhile, 2-sided Anon-Sel requires that $(\mathbf{x}, G)$ remains hidden even from users who successfully decrypt the ciphertext. Given two pairs $(\mathbf{x}_0, G_0), (\mathbf{x}_1, G_1)$ that yield the same decryption outcome, any user who evaluates C_{dec} using his/her key cannot distinguish which pair was hardwired. This indistinguishability between two functionally equivalent circuits implies $i\mathcal{O}$.

- $\text{sExtract}(\text{mpk}, \text{msk}, \mathbf{z}_0)$: Identify the index $i \in [N]$ such that $\mathbf{z}_0 = \mathbf{z}_0^{(i)} \in \mathcal{L}$. Sample $\mathbf{R}_L^{(i)} \leftarrow \text{SamplePre}(\mathbf{A}_s, \mathbf{T}_{\mathbf{A}_s}, \mathbf{B}_{\ell_0}^{(i)} - \mathbf{z}_0 \otimes \mathbf{G}, \tau_1)$. Note that $\mathbf{B}_{\ell_0}^{(i)} = \mathbf{A}_s \mathbf{R}_L^{(i)} + \mathbf{z}_0 \otimes \mathbf{G}$. Output $\text{ssk}_{\mathbf{z}_0} := \mathbf{R}_L^{(i)}$.
- $\text{Derive}(\text{mpk}, \text{ssk}_{\mathbf{z}_0}, \mathbf{z}_1)$: Let $\mathbf{z}_1 \in \{0, 1\}^{\ell_1}$. Sample

$$\begin{pmatrix} \mathbf{R}_1 \\ \vdots \\ \mathbf{R}_{\ell_1} \\ \mathbf{Q} \end{pmatrix} \leftarrow \text{SamplePre} \left(\mathbf{C}_{\ell_1}, \mathbf{T}', \begin{pmatrix} -\mathbf{z}_1[1] \mathbf{V}_1 \mathbf{G} \\ \vdots \\ -\mathbf{z}_1[\ell_1] \mathbf{V}_{\ell_1} \mathbf{G} \end{pmatrix}, \tau_1 \right).$$

Let $\mathbf{R}_R := [\mathbf{R}_1 || \dots || \mathbf{R}_{\ell_1}]$ and $\mathbf{B}_{\ell_1} := [\mathbf{V}_1^{-1} \mathbf{G} \mathbf{Q} || \dots || \mathbf{V}_{\ell_1}^{-1} \mathbf{G} \mathbf{Q}] \in \mathbb{Z}_q^{n \times m' \ell_1}$. Note that $\mathbf{A}_s \mathbf{R}_R = \mathbf{B}_{\ell_1} - \mathbf{z}_1 \otimes \mathbf{G}$. Output $\text{ssk}_{\mathbf{z}_0 || \mathbf{z}_1} := ([\mathbf{R}_L^{(i)} || \mathbf{R}_R], \mathbf{Q})$.

- $\text{dExtract}(\text{mpk}, \text{msk}, F, \mathbf{y})$: Compute

$$\begin{aligned} \mathbf{P}_\ell^{(1)} &:= \text{Ver}^{\text{vc}}(\text{pp}_{\text{com}}, 1^\ell), & \mathbf{B}_1 &:= -\mathbf{A}_1 \mathbf{P}_\ell^{(1)}, & \mathbf{B}_F &:= \text{EvalF}(\mathbf{B}_1, F), \\ \mathbf{P}_{y,d}^{(2)} &:= \text{Ver}^c(\text{pp}_{\text{com}}, \mathbf{y}, 1^d), & \mathbf{B}_y &:= -\mathbf{A}_2 \mathbf{P}_{y,d}^{(2)}. \end{aligned}$$

Sample $\mathbf{D} \leftarrow \text{SamplePre}([\mathbf{A}_r || \mathbf{B}_F || \mathbf{B}_y], \mathbf{T}_{\mathbf{A}_r}, \mathbf{G}, \tau_1)$. Output $\text{dsk}_{F,y} := \mathbf{D}$.

- $\text{Signcrypt}(\text{mpk}, \text{ssk}_{\mathbf{z}_0 || \mathbf{z}_1}, C, (\mathbf{x}, G), \mu)$: If $C(\mathbf{z}_0 || \mathbf{z}_1) = 0$, output $\perp$. Otherwise, do the following.

1. Sample $(\text{ovk}, \text{osk}) \leftarrow \text{OTS.KGen}(1^\lambda)$.
2. *Signature components*: Compute the message encoding matrix $\mathbf{M} := \mathbf{M}_0 + \sum_{k \in [L]} \mathbf{m}[k] \cdot \mathbf{M}_k$, where $\mathbf{m} := [\mu || \text{ovk}] \in \{0, 1\}^L$. Compute

$$\begin{aligned} \mathbf{B}_C^{(i)} &:= \text{EvalF}([\mathbf{B}_{\ell_0}^{(i)} || \mathbf{B}_{\ell_1}], C) \\ \mathbf{H}_{C, [\mathbf{z}_0 || \mathbf{z}_1]} &:= \text{EvalFX}([\mathbf{B}_{\ell_0}^{(i)} || \mathbf{B}_{\ell_1}], C, [\mathbf{z}_0 || \mathbf{z}_1]). \end{aligned}$$

Note that $\mathbf{A}_s \cdot [\mathbf{R}_L^{(i)} || \mathbf{R}_R] = [\mathbf{B}_{\ell_0}^{(i)} || \mathbf{B}_{\ell_1}] - [\mathbf{z}_0 || \mathbf{z}_1] \otimes \mathbf{G}$. Since $C(\mathbf{z}_0 || \mathbf{z}_1) = 1$, we have $\mathbf{B}_C^{(i)} = \mathbf{A}_s [\mathbf{R}_L^{(i)} || \mathbf{R}_R] \cdot \mathbf{H}_{C, [\mathbf{z}_0 || \mathbf{z}_1]} + \mathbf{G}$. It then computes $\mathbf{T}_{[\mathbf{A}_s || \mathbf{B}_C^{(i)}]}$ from $[\mathbf{R}_L^{(i)} || \mathbf{R}_R] \mathbf{H}_{C, [\mathbf{z}_0 || \mathbf{z}_1]}$ via Lemma 3. Sample

$$\mathbf{u} \leftarrow \text{SamplePre}([\mathbf{A}_s || \mathbf{B}_C^{(1)} || \dots || \mathbf{B}_C^{(N)} || \mathbf{M}], \mathbf{T}_{[\mathbf{A}_s || \mathbf{B}_C^{(i)}]}, \mathbf{0}, \tau_2).$$

3. *Attribute-policy encoding component*: Choose a lock string $\delta \leftarrow \{0, 1\}^\kappa$. Compute $\mathbf{C}_x := \text{Com}^{\text{vc}}(\text{pp}_{\text{com}}, \mathbf{x})$, $\mathbf{C}_G := \text{Com}^c(\text{pp}_{\text{com}}, G)$, and $\mathbf{C}_{\text{ovk}} := \text{Com}^{\text{vc}}(\text{pp}_{\text{com}}, \text{ovk})$. Sample

$$\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z}, \chi}^m, \mathbf{e}_1 \leftarrow \mathcal{D}_{\mathbb{Z}, \chi'}^m, \mathbf{e}_2 \leftarrow \mathcal{D}_{\mathbb{Z}, \chi'}^m, \mathbf{e}_3 \leftarrow \mathcal{D}_{\mathbb{Z}, \chi'}^m, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}, \chi'}^\kappa.$$

It then computes $\tilde{\text{ct}}$ as follows

$$\overbrace{(\mathbf{s} \mathbf{A}_r + \mathbf{e}_0)}^{c_0}, \overbrace{(\mathbf{s}(\mathbf{A}_1 + \mathbf{C}_x) + \mathbf{e}_1)}^{c_1}, \overbrace{(\mathbf{s}(\mathbf{A}_2 + \mathbf{C}_G) + \mathbf{e}_2)}^{c_2}, \overbrace{(\mathbf{s}(\mathbf{A}_3 + \mathbf{C}_{\text{ovk}}) + \mathbf{e}_3)}^{c_3}, \overbrace{(\mathbf{s} \mathbf{P} + \mathbf{e} + \lfloor q/2 \rfloor \cdot \delta)}^c.$$

Subroutine($\text{dsk}_{F,Y,F_{\text{ovk}}}, \tilde{\text{ct}}$) $\rightarrow \delta$ or $\perp$: Parse $\text{dsk}_{F,Y,F_{\text{ovk}}}$ as $\mathbf{D}'$. Compute

1. $\mathbf{P}_\ell^{(1)} := \text{Ver}^{\text{vc}}(\text{pp}_{\text{com}}, 1^\ell), \mathbf{Z}_x := \text{Open}^{\text{vc}}(\text{pp}_{\text{com}}, \mathbf{x}),$
 $\mathbf{B}_1 := -\mathbf{A}_1 \mathbf{P}_\ell^{(1)}, \mathbf{H}_{F,x} := \text{EvalFX}(\mathbf{B}_1, F, \mathbf{x}).$
2. $\mathbf{P}_{y,d}^{(2)} := \text{Ver}^c(\text{pp}_{\text{com}}, \mathbf{y}, 1^d), \mathbf{Z}_{G,y} := \text{Open}^c(\text{pp}_{\text{com}}, G, \mathbf{y}), \mathbf{B}_y := -\mathbf{A}_2 \mathbf{P}_{y,d}^{(2)}.$
3. $\mathbf{P}_{\text{ovk}}^{(3)} := \text{Ver}^{\text{vc}}(\text{pp}_{\text{com}}, 1^{|\text{ovk}|}), \mathbf{Z}_{\text{ovk}} := \text{Open}^{\text{vc}}(\text{pp}_{\text{com}}, \text{ovk}),$
 $\mathbf{B}_3 := -\mathbf{A}_3 \mathbf{P}_{\text{ovk}}^{(3)}, \mathbf{H}_{F_{\text{ovk}},\text{ovk}} := \text{EvalFX}(\mathbf{B}_3, F_{\text{ovk}}, \text{ovk}).$
4. $\mathbf{d}_1 := [\mathbf{c}_0 \parallel \mathbf{c}_1] \cdot \begin{pmatrix} -\mathbf{Z}_x \\ -\mathbf{P}_\ell^{(1)} \end{pmatrix} \cdot \mathbf{H}_{F,x},$
 $\mathbf{d}_2 := [\mathbf{c}_0 \parallel \mathbf{c}_2] \cdot \begin{pmatrix} -\mathbf{Z}_{G,y} \\ -\mathbf{P}_{y,d}^{(2)} \end{pmatrix},$
 $\mathbf{d}_3 := [\mathbf{c}_0 \parallel \mathbf{c}_3] \cdot \begin{pmatrix} -\mathbf{Z}_{\text{ovk}} \\ -\mathbf{P}_{\text{ovk}}^{(3)} \end{pmatrix} \cdot \mathbf{H}_{F_{\text{ovk}},\text{ovk}}.$

Output $\text{Round}_\beta(\mathbf{c} - [\mathbf{c}_0 \parallel \mathbf{d}_1 \parallel \mathbf{d}_2 \parallel \mathbf{d}_3] \cdot \mathbf{D}').$

Fig. 5. Description of the circuit Subroutine. F_{ovk} is an equality-test function for ovk that outputs 0 if the input equals ovk and 1 otherwise.

4. Anonymization: Prepare a circuit Subroutine as defined in Fig. 5 and run

$$\tilde{P} \leftarrow \text{LO.Obf}(1^\lambda, \text{Subroutine}(\cdot, \tilde{\text{ct}}), \text{bin}_q(\mathbf{u}) \parallel \mu, \delta).^6$$

5. Run $\sigma \leftarrow \text{OTS.Sign}_{\text{osk}}(\tilde{P} \parallel \mathbf{Q})$. Output $\text{ct} := (\text{ovk}, \tilde{P}, \mathbf{Q}, \sigma)$ with C .
 – Unsigncrypt($\text{mpk}, \text{ct}, \text{dsk}_{F,Y}$) $\rightarrow \mu / \perp$: If $0 \leftarrow \text{OTS.Vrfy}_{\text{ovk}}((\tilde{P} \parallel \mathbf{Q}), \sigma)$, output $\perp$. Otherwise, do the following:
1. Derive the decryption key for ovk: Compute $\mathbf{P}_{\text{ovk}}^{(3)} := \text{Ver}^{\text{vc}}(\text{pp}_{\text{com}}, 1^{|\text{ovk}|}),$
 $\mathbf{B}_3 := -\mathbf{A}_3 \mathbf{P}_{\text{ovk}}^{(3)}, \mathbf{B}_{F_{\text{ovk}}} := \text{EvalF}(\mathbf{B}_3, F_{\text{ovk}})$. Sample
 $\mathbf{D}' \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F \parallel \mathbf{B}_y \parallel \mathbf{B}_{F_{\text{ovk}}}], \mathbf{D}, \mathbf{P}, \tau_2).$
 2. Recover message and signature: Compute $(\mathbf{u}, \mu) \leftarrow \text{LO.Eval}(\mathbf{D}', \tilde{P})$.
 3. Verify: Compute $\mathbf{M} := \mathbf{M}_0 + \sum_{k \in [L]} \mathbf{m}[k] \cdot \mathbf{M}_k$ and

$$\mathbf{B}_{\ell_1} = [\mathbf{V}_1^{-1} \mathbf{G} \mathbf{Q} \parallel \dots \parallel \mathbf{V}_{\ell_1}^{-1} \mathbf{G} \mathbf{Q}], \quad \mathbf{B}_C^{(i)} := \text{EvalF}([\mathbf{B}_{\ell_0}^{(i)} \parallel \mathbf{B}_{\ell_1}], C).$$

If $\|\mathbf{u}\| \leq \tau_2 \sqrt{m + (N+1)m'}$ and $[\mathbf{A}_s \parallel \mathbf{B}_C^{(1)} \parallel \dots \parallel \mathbf{B}_C^{(N)} \parallel \mathbf{M}] \cdot \mathbf{u}^\top = \mathbf{0}^\top$, output μ ; otherwise, output $\perp$.

Correctness. The following is a step-by-step process to verify the correctness.

Step 1 : Let $\mathbf{P}_\ell^{(1)} := \text{Ver}^{\text{vc}}(\text{pp}_{\text{com}}, 1^\ell), \mathbf{Z}_x := \text{Open}^{\text{vc}}(\text{pp}_{\text{com}}, \mathbf{x})$. By Lemma 5, we have $\mathbf{C}_x \cdot \mathbf{P}_\ell^{(1)} = -\mathbf{A}_r \cdot \mathbf{Z}_x + \mathbf{x} \otimes \mathbf{G}$. Observe that $\mathbf{B}_1 := -\mathbf{A}_1 \mathbf{P}_\ell^{(1)}$, we have

$$[\mathbf{A}_r \parallel \mathbf{A}_1 + \mathbf{C}_x] \cdot \begin{pmatrix} -\mathbf{Z}_x \\ -\mathbf{P}_\ell^{(1)} \end{pmatrix} = -\mathbf{A}_1 \mathbf{P}_\ell^{(1)} \underbrace{-\mathbf{A}_r \mathbf{Z}_x - \mathbf{C}_x \mathbf{P}_\ell^{(1)}}_{:= -\mathbf{x} \otimes \mathbf{G}} = \mathbf{B}_1 - \mathbf{x} \otimes \mathbf{G}.$$

⁶ For simplicity, Obf takes $(\text{bin}_q(\mathbf{u}), \mu)$ directly as an input. This can be replaced by a symmetric key K , with $(\text{bin}_q(\mathbf{u}), \mu)$ encrypted using symmetric key encryption (SKE).

Then, $\mathbf{d}_1$ is computed as

$$[\mathbf{c}_0 \| \mathbf{c}_1] \cdot \begin{pmatrix} -\mathbf{Z}_{\mathbf{x}} \\ -\mathbf{P}_{\ell}^{(1)} \end{pmatrix} \cdot \mathbf{H}_{F,\mathbf{x}} = \mathbf{sB}_F + \mathbf{e}'_1. \quad (7)$$

Here, we use the key equation (Lemma 4)

$$(\mathbf{B}_1 - \mathbf{x} \otimes \mathbf{G}) \cdot \mathbf{H}_{F,\mathbf{x}} = \mathbf{B}_F - F(\mathbf{x})\mathbf{G} \text{ with } F(\mathbf{x}) = 0. \quad \textcolor{red}{4}$$

Step 2 : Let $\mathbf{P}_{y,d}^{(2)} := \text{Ver}^c(\text{pp}_{\text{com}}, \mathbf{y}, 1^d)$, $\mathbf{Z}_{G,y} := \text{Open}^c(\text{pp}_{\text{com}}, G, \mathbf{y})$. By Lemma 6, we have $\mathbf{C}_G \cdot \mathbf{P}_{y,d}^{(2)} = G(\mathbf{y})\mathbf{G} - \mathbf{A}_r \mathbf{Z}_{G,y}$. Observe that $\mathbf{B}_y := -\mathbf{A}_2 \mathbf{P}_{y,d}^{(2)}$, we have

$$[\mathbf{A}_r \| \mathbf{A}_2 + \mathbf{C}_G] \cdot \begin{pmatrix} -\mathbf{Z}_{G,y} \\ -\mathbf{P}_{y,d}^{(2)} \end{pmatrix} = -\mathbf{A}_2 \mathbf{P}_{y,d}^{(2)} \underbrace{-\mathbf{A}_r \mathbf{Z}_{G,y} - \mathbf{C}_G \mathbf{P}_{y,d}^{(2)}}_{:= -G(\mathbf{y})\mathbf{G}} = \mathbf{B}_y - G(\mathbf{y})\mathbf{G}.$$

Then, for $G(\mathbf{y}) = 0$,⁴ $\mathbf{d}_2$ is computed as

$$[\mathbf{c}_0 \| \mathbf{c}_2] \cdot \begin{pmatrix} -\mathbf{Z}_{G,y} \\ -\mathbf{P}_{y,d}^{(2)} \end{pmatrix} = \mathbf{sB}_y + \mathbf{e}'_2. \quad (8)$$

Step 3 : Similar to Step 1, for $F_{\text{ovk}}(\text{ovk}) = 0$, $\mathbf{d}_3$ is computed as

$$[\mathbf{c}_0 \| \mathbf{c}_3] \cdot \begin{pmatrix} -\mathbf{Z}_{\text{ovk}} \\ -\mathbf{P}_{\text{ovk}}^{(3)} \end{pmatrix} \cdot \mathbf{H}_{F_{\text{ovk}}, \text{ovk}} = \mathbf{sB}_{F_{\text{ovk}}} + \mathbf{e}'_3. \quad (9)$$

Step 4 : Connecting (7), (8), and (9), we deduce that

$$\begin{aligned} \mathbf{c} - [\mathbf{c}_0 \| \mathbf{d}_1 \| \mathbf{d}_2 \| \mathbf{d}_3] \cdot \mathbf{D}' &= \mathbf{sP} + \lfloor \frac{q}{2} \rfloor \boldsymbol{\delta} - \mathbf{s}[\mathbf{A}_r \| \mathbf{B}_F \| \mathbf{B}_y \| \mathbf{B}_{F_{\text{ovk}}}] \cdot \mathbf{D}' + \mathbf{e}_{\text{total}} \\ &= \lfloor \frac{q}{2} \rfloor \boldsymbol{\delta} + \mathbf{e}_{\text{total}}, \end{aligned}$$

where the total error term $\mathbf{e}_{\text{total}} := \mathbf{e} - [\mathbf{e}_0 \| \mathbf{e}'_1 \| \mathbf{e}'_2 \| \mathbf{e}'_3] \cdot \mathbf{D}'$. From the above argument, we have $\boldsymbol{\delta} \leftarrow \text{Subroutine}(\mathbf{D}', \text{ct})$. By the correctness of LO (Definition 19), we recover $(\mathbf{u}, \mu) \leftarrow \text{LO.Eval}(\mathbf{D}', \tilde{P})$.

Step 5 : Next, we verify the correctness of the signature. Given $\mathbf{A}_s \mathbf{R}_{\mathbf{L}}^{(i)} = \mathbf{B}_{\ell_0}^{(i)} - \mathbf{z}_0 \otimes \mathbf{G}$ and $\mathbf{A}_s \mathbf{R}_{\mathbf{R}} = \mathbf{B}_{\ell_1} - \mathbf{z}_1 \otimes \mathbf{G}$, if $C(\mathbf{z}_0 \| \mathbf{z}_1) = 1$, then

$$\begin{aligned} \mathbf{A}_s \cdot [\mathbf{R}_{\mathbf{L}}^{(i)} \| \mathbf{R}_{\mathbf{R}}] \cdot \mathbf{H}_{C, [\mathbf{z}_0 \| \mathbf{z}_1]} &= ([\mathbf{B}_{\ell_0}^{(i)} \| \mathbf{B}_{\ell_1}] - [\mathbf{z}_0 \| \mathbf{z}_1] \otimes \mathbf{G}) \cdot \mathbf{H}_{C, [\mathbf{z}_0 \| \mathbf{z}_1]} \\ &= \mathbf{B}_C^{(i)} - \mathbf{G}. \end{aligned}$$

Therefore, we sample $\mathbf{T}_{[\mathbf{A}_s \| \mathbf{B}_C^{(i)}]}$ from $[\mathbf{R}_{\mathbf{L}}^{(i)} \| \mathbf{R}_{\mathbf{R}}] \mathbf{H}_{C, [\mathbf{z}_0 \| \mathbf{z}_1]}$ using Item 3 of Lemma 3. With this trapdoor, we can sample a low-norm vector $\mathbf{u}$ as a signature such that $[\mathbf{A}_s \| \mathbf{B}_C^{(1)} \| \dots \| \mathbf{B}_C^{(N)} \| \mathbf{M}] \cdot \mathbf{u}^\top = \mathbf{0}^\top$. Furthermore, by the definition of lattice trapdoors, we have $\|\mathbf{u}\| \leq \tau_2 \sqrt{m + (N+1)m'}$.

Due to space constraints, we defer error bounds and parameters to Appendix B.

Security Proof. We establish the security of our scheme via the following theorems, with the full proofs deferred to Appendices C.1–C.3.

Theorem 2. *Our Special A²BSC scheme satisfies IND-Sel-CCA security (Definition 10), assuming $(2m^2, \tau_0)$ -succinct LWE (Assumption 4) is hard, Obf is secure, and OTS is secure.*

Theorem 3. *Our Special A²BSC scheme satisfies no-signing EUF-CoSel security (Definition 12), assuming BASIS _{$n, m, q, \beta_{\text{SIS}}, \tau_1, \ell_1$} (Assumption 2) is hard.*

Theorem 4. *Our Special A²BSC scheme satisfies 1-sided Anon-Sel (Definition 13), assuming $(2m^2, \tau_0)$ -succinct LWE (Assumption 4) is hard, Obf is secure, and OTS is secure.*

4.3 Special A²BSC with Separable Decryption Keys for Circuits from Lattices

Our Special A²BSC with separable decryption keys is identical to the Special A²BSC scheme in Sect. 4.2, except that we modify the algorithms as follows:

- $\text{dExtract}_0(\text{mpk}, \text{msk}, F)$: Compute

$$\mathbf{P}_\ell^{(1)} := \text{Ver}^{\text{vc}}(\text{pp}_{\text{com}}, 1^\ell), \mathbf{B}_1 := -\mathbf{A}_1 \mathbf{P}_\ell^{(1)}, \mathbf{B}_F := \text{EvalF}(\mathbf{B}_1, F).$$

Sample $\mathbf{D}_0 \leftarrow \text{SamplePre}([\mathbf{A}_r \| \mathbf{B}_F], \mathbf{T}_{\mathbf{A}_r}, \mathbf{P}_0, \tau_1)$, where $\mathbf{P}_0 \leftarrow \mathbb{Z}_q^{n \times m}$ is the additional random matrix introduced in mpk . Output $\text{dsk}_F := \mathbf{D}_0$.

- $\text{dExtract}_1(\text{mpk}, \text{msk}, \mathbf{y})$: Compute $\mathbf{P}_{y,d}^{(2)} := \text{Ver}^c(\text{pp}_{\text{com}}, \mathbf{y}, 1^d)$, $\mathbf{B}_y := -\mathbf{A}_2 \mathbf{P}_{y,d}^{(2)}$ and $\mathbf{P}_1 := \mathbf{G} - \mathbf{P}_0$. Sample $\mathbf{D}_1 \leftarrow \text{SamplePre}([\mathbf{A}_r \| \mathbf{B}_y], \mathbf{T}_{\mathbf{A}_r}, \mathbf{P}_1, \tau_1)$. Output $\text{dsk}_y := \mathbf{D}_1$.

- $\text{Unsigncrypt}(\text{mpk}, \text{ct}, \text{dsk}_F, \text{dsk}_y) \rightarrow \mu / \perp$: If $0 \leftarrow \text{OTS.Vrfy}_{\text{ovk}}((\tilde{P} \| \mathbf{Q}), \sigma)$, output $\perp$. Otherwise, do the following:

1. Derive the decryption key for ovk: Parse $\mathbf{D}_0 := \begin{pmatrix} \mathbf{D}_{0,\text{top}} \\ \mathbf{D}_{0,\text{bot}} \end{pmatrix}$, $\mathbf{D}_1 := \begin{pmatrix} \mathbf{D}_{1,\text{top}} \\ \mathbf{D}_{1,\text{bot}} \end{pmatrix}$, where $(\mathbf{D}_{i,\text{top}}, \mathbf{D}_{i,\text{bot}}) \in \mathbb{Z}^{m \times m} \times \mathbb{Z}^{m \times m}$ for each $i \in \{0, 1\}$. Compute

$$\mathbf{D} := \begin{pmatrix} \mathbf{D}_{0,\text{top}} + \mathbf{D}_{1,\text{top}} \\ \mathbf{D}_{0,\text{bot}} \\ \mathbf{D}_{1,\text{bot}} \end{pmatrix}.$$

Note that $\mathbf{D}$ is a trapdoor for $[\mathbf{A}_r \| \mathbf{B}_F \| \mathbf{B}_y]$ s.t. $[\mathbf{A}_r \| \mathbf{B}_F \| \mathbf{B}_y] \cdot \mathbf{D} = \mathbf{G}$.

It then computes $\mathbf{P}_{\text{ovk}}^{(3)} := \text{Ver}^{\text{vc}}(\text{pp}_{\text{com}}, 1^{|\text{ovk}|})$, $\mathbf{B}_3 := -\mathbf{A}_3 \mathbf{P}_{\text{ovk}}^{(3)}$, $\mathbf{B}_{F_{\text{ovk}}} := \text{EvalF}(\mathbf{B}_3, F_{\text{ovk}})$ and samples

$$\mathbf{D}' \leftarrow \text{SamplePre}([\mathbf{A}_r \| \mathbf{B}_F \| \mathbf{B}_y \| \mathbf{B}_{F_{\text{ovk}}}], \mathbf{D}, \mathbf{P}, \tau_2),$$

$\vdots$

Security Proof. We establish the IND-Sel-CCA security of our scheme via the following theorem, with the full proofs deferred to Appendix C.4.

Theorem 5. *Our Special A²BSC with separable keys satisfies IND-Sel-CCA security (Definition 10), assuming $(2m^2, \tau_0)$ -succinct LWE (Assumption 4) is hard, Obf is secure, and OTS is secure.*

Observe that the construction in Sect. 4.2 modifies only the structure of the decryption secret key, which does not affect its unforgeability. Thus, the proof strategy employed in Theorem 3 carries over directly, establishing the following theorem.

Theorem 6. *Our Special A²BSC scheme with separable keys satisfies no-signing EUF-CoSel security (Definition 12), assuming $\text{BASIS}_{n,m,q,\beta_{\text{SIS}},\tau_1,\ell_1}$ (Assumption 2) is hard.*

The following theorem follows readily from the proof strategy of Theorem 4.

Theorem 7. *Our Special A²BSC scheme with separable keys satisfies 1-sided Anon-Sel (Definition 13), assuming $(2m^2, \tau_0)$ -succinct LWE (Assumption 4) is hard, Obf is secure, and OTS is secure.*

5 Generic Constructions of Matchmaking Encryption

In this section, we present two generic constructions from A²BSC: one for AME and the other for ME.

5.1 AME from Special A²BSC

Building Blocks. Our construction requires the following building blocks.

- An FC scheme $\Pi_{\text{FC}} = \text{FC}(\text{Setup}, \text{Commit}, \text{Eval}, \text{Verify})$ for circuit class $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$, where $\mathcal{F}_\lambda$ is defined as $\mathcal{F}_\lambda := \{F : \{0, 1\}^{\ell(\lambda)} \rightarrow \{0, 1\}\}$.
- A Special A²BSC scheme $\Pi_{\text{A}^2\text{BSC}} = \text{A}^2\text{BSC}(\text{Setup}, \text{sExtract}, \text{dExtract}, \text{Derive}, \text{Signcrypt}, \text{Unsigncrypt})$ for circuit classes $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$, $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ and $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda \in \mathbb{N}}$, where $\mathcal{C}_\lambda, \mathcal{F}_\lambda, \mathcal{G}_\lambda$ are defined as

$$\mathcal{C}_\lambda := \{C[\sigma] \mid \sigma \in \{0, 1\}^{\zeta(\lambda)}\}, \mathcal{F}_\lambda = \mathcal{G}_\lambda := \{G : \{0, 1\}^{\ell(\lambda)} \rightarrow \{0, 1\}\}.$$

Here, the circuit $C[\sigma]$ with input space $\{0, 1\}^{\kappa(\lambda)}$ is formally defined in Fig. 6. $\zeta(\lambda)$ denotes the upper bounds on the length of σ output by FC.Commit .

- An FE scheme $\Pi_{\text{FE}} = \text{FE}(\text{Setup}, \text{KeyGen}, \text{Enc}, \text{Dec})$ for circuit class $\mathcal{F}_{\text{Eval}}$, where $F_{\text{Eval}} \in \mathcal{F}_{\text{Eval}}$ is defined as Eq. (10).

In the following, we describe our AME construction.

Setup(1^λ): On input the security parameter λ , it first runs $(\text{mpk}_0, \text{msk}_0) \leftarrow \text{A}^2\text{BSC.Setup}(1^\lambda)$, $(\text{mpk}_1, \text{msk}_1) \leftarrow \text{FE.Setup}(1^\lambda)$, $\text{crs} \leftarrow \text{FC.Setup}(1^\lambda)$. Output $\text{mpk} := (\text{mpk}_0, \text{mpk}_1, \text{crs})$, $\text{msk} := (\text{msk}_0, \text{msk}_1)$.

SKGen($\text{mpk}, \text{msk}, \mathbf{x}$): On input (mpk, msk) and attributes $\mathbf{x} \in \{0, 1\}^{\ell(\lambda)}$, it computes $\text{ssk}_{\mathbf{x}} \leftarrow \text{A}^2\text{BSC.sExtract}(\text{mpk}_0, \text{msk}_0, \mathbf{x})$ and outputs $\text{ek}_{\mathbf{x}} := \text{ssk}_{\mathbf{x}}$.

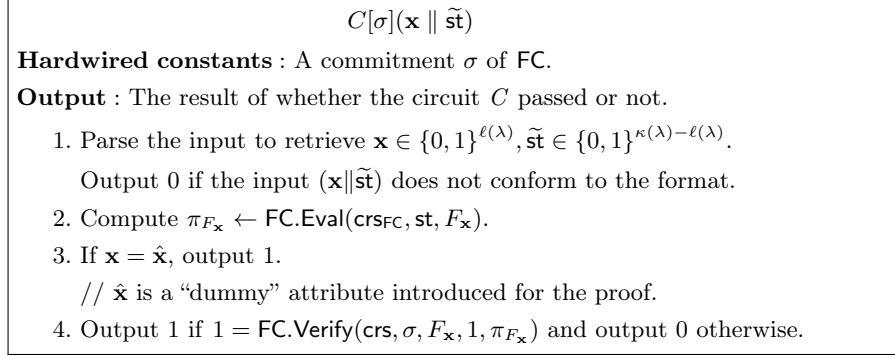


Fig. 6. Description of the circuit $C[\sigma]$. $F_{\mathbf{x}}$ is an equality-test function for $\mathbf{x}$ that outputs 1 if the input equals $\mathbf{x}$ and 0 otherwise.

RKGen(mpk, msk, F , $\mathbf{y}$) : On input (mpk, msk), a function $F \in \mathcal{F}$, and attributes $\mathbf{y} \in \{0, 1\}^{\ell(\lambda)}$, it computes $\text{dsk}_{F, \mathbf{y}} \leftarrow \text{A}^2\text{BSC.dExtract}(\text{mpk}_0, \text{msk}_0, F, \mathbf{y})$ and $\text{sk}_{\text{Eval}_F} \leftarrow \text{FE.KeyGen}(\text{mpk}_1, \text{msk}_1, F_{\text{Eval}})$, where the circuit F_{Eval} with crs and F hardwired is defined as follows:

$$F_{\text{Eval}}[\text{crs}, F](\mathcal{T}, \text{st}) = \begin{cases} \text{FC.Eval}(\text{crs}, \text{st}, F), & \text{if } \mathcal{T} = 0 \\ \pi_F, & \text{if } \mathcal{T} = 1 \end{cases} \quad (10)$$

The latter case of F_{Eval} only appears in the security proof, where π_F is parsed directly from st. Finally, output $\text{dk}_{F, \mathbf{y}} := (\text{dsk}_{F, \mathbf{y}}, \text{sk}_{\text{Eval}_F})$.

Enc(mpk, $\text{ek}_{\mathbf{x}}$, G , μ) : On input the master public key mpk, an encryption key $\text{ek}_{\mathbf{x}}$, a policy function $G \in \mathcal{G}$ and a message $\mu \in \mathcal{M}$, do the following.

1. Compute $(\sigma, \text{st}) \leftarrow \text{FC.Commit}(\text{crs}, \mathbf{x})$, where st is composed of $(\mathbf{x}, \tilde{\mathbf{st}})$.
2. Compute $\text{ct}_{\text{FE}} \leftarrow \text{FE.Enc}(\text{mpk}_1, (\mathcal{T}, \text{st}))$, where the tag $\mathcal{T} := 0$.
3. Parse $\text{ek}_{\mathbf{x}}$ as $\text{ssk}_{\mathbf{x}}$. Run $\text{ssk}_{\mathbf{x} \parallel \tilde{\mathbf{st}}} \leftarrow \text{A}^2\text{BSC.Derive}(\text{mpk}_0, \text{ssk}_{\mathbf{x}}, \tilde{\mathbf{st}})$ and

$$\text{ct}_1 \leftarrow \text{A}^2\text{BSC.Signcrypt}(\text{mpk}_0, \text{ssk}_{\mathbf{x} \parallel \tilde{\mathbf{st}}}, C[\sigma], (\mathbf{x}, G), \mu')$$

where $\mu' := (\mu, \text{ct}_{\text{FE}})$, and the circuit $C[\sigma] \in \mathcal{C}$ is defined in Fig. 6.

4. Output $\text{ct} := (\text{ct}_1, \sigma)$.

Dec(mpk, $\text{dk}_{F, \mathbf{y}}$, ct) : On input the master public key mpk, a decryption key $\text{dk}_{F, \mathbf{y}} := (\text{dsk}_{F, \mathbf{y}}, \text{sk}_{\text{Eval}_F})$, and a ciphertext ct, it proceeds as follows:

1. Compute $\mu' \leftarrow \text{A}^2\text{BSC.Unsigncrypt}(\text{mpk}_0, \text{ct}_1, \text{dsk}_{F, \mathbf{y}})$. If $\mu' \neq \perp$, parse μ' as $(\mu, \text{ct}_{\text{FE}})$ and compute an opening $\pi_F \leftarrow \text{FE.Dec}(\text{sk}_{\text{Eval}_F}, \text{ct}_{\text{FE}})$. Otherwise, output $\perp$.
2. If $\text{FC.Verify}(\text{crs}, \sigma, F, 1, \pi_F) = 1$, output μ . Otherwise, output $\perp$.

Correctness. Since $C[\sigma](\mathbf{x} \parallel \tilde{\mathbf{st}}) = 1$, ct_1 is a valid A^2BSC ciphertext. If $F(\mathbf{x}) = G(\mathbf{y}) = 1$, the correctness of Special A^2BSC further guarantees that the message μ' is recovered. By the correctness of the FE scheme, we obtain an

opening π_F for policy F . Then, we have $\text{FC.Verify}(\text{crs}, \sigma, F, 1, \pi_F) = 1$, which follows from $F(\mathbf{x}) = 1$ and the correctness of the FC scheme.

Security Proof. We establish the security of our construction via the following theorems, with the full proofs deferred to Appendices C.5 and C.6.

Theorem 8. *Let $t \in [2]$. If the FE scheme satisfies security (Definition 18), the FC scheme supports private openings (Definition 16), the Special A²BSC scheme satisfies IND-Sel-CCA (Definition 10) and t -sided Anon-Sel (Definition 13), then our AME construction satisfies CCA- t -sided selective privacy (Definition 3).*

Theorem 9. *If the Special A²BSC scheme satisfies no-signing EUF-CoSel (Definition 12), and the FC scheme is binding (Definition 15), then our AME construction achieves authenticity (Definition 4).*

5.2 ME from Special A²BSC

In this section, our building blocks for ME are identical to those in Sect. 5.1, except that we consider an attribute-separable decryption key for Special A²BSC (see 4.3). In particular, $\text{dsk}_{F,\mathbf{y}} \leftarrow \text{A}^2\text{BSC.dExtract}(\text{mpk}, \text{msk}, F, \mathbf{y})$ can be separated into two parts, one is $\text{dsk}_F \leftarrow \text{A}^2\text{BSC.dExtract}_0(\text{mpk}, \text{msk}, F)$ with F , and the other is $\text{dsk}_{\mathbf{y}} \leftarrow \text{A}^2\text{BSC.dExtract}_1(\text{mpk}, \text{msk}, \mathbf{y})$ with $\mathbf{y}$. Then, $\text{dsk}_{F,\mathbf{y}}$ can be reconstructed by combining dsk_F and $\text{dsk}_{\mathbf{y}}$. As a result, our ME construction is similar to that in Sect. 5.1, except that we modify these algorithms as follows:

$\text{RKGen}(\text{mpk}, \text{msk}, \mathbf{y})$: On input the master public-secret key pair (mpk, msk) and attributes $\mathbf{y} \in \{0, 1\}^{\ell(\lambda)}$, it outputs $\text{dsk}_{\mathbf{y}} \leftarrow \text{A}^2\text{BSC.dExtract}_1(\text{mpk}_0, \text{msk}_0, \mathbf{y})$.
 $\text{PolGen}(\text{mpk}, \text{msk}, F)$: On input the master public-secret key pair (mpk, msk) and a policy function $F \in \mathcal{F}$, it runs $\text{dsk}_F \leftarrow \text{A}^2\text{BSC.dExtract}_0(\text{mpk}_0, \text{msk}_0, F)$, $\text{sk}_{\text{Eval}_F} \leftarrow \text{FE.KeyGen}(\text{mpk}_1, \text{msk}_1, F_{\text{Eval}})$, where the circuit F_{Eval} , with crs and F hardwired, is defined as in (10). Output $\text{dk}_F := (\text{dsk}_F, \text{sk}_{\text{Eval}_F})$.
 $\text{Dec}(\text{mpk}, \text{dk}_F, \text{dk}_{\mathbf{y}}, \text{ct})$: On input the master public key mpk , the keys $\text{dk}_F := (\text{dsk}_F, \text{sk}_{\text{Eval}_F})$ and $\text{dk}_{\mathbf{y}} := \text{dsk}_{\mathbf{y}}$, and a ciphertext ct , it proceeds as follows:
 1. Compute $\mu' \leftarrow \text{A}^2\text{BSC.Unsigncrypt}(\text{mpk}_0, \text{ct}_1, (\text{dsk}_F, \text{dsk}_{\mathbf{y}}))$. If $\mu' \neq \perp$, parse μ' as $(\mu, \text{ct}_{\text{FE}})$ and compute an opening $\pi_F \leftarrow \text{FE.Dec}(\text{sk}_{\text{Eval}_F}, \text{ct}_{\text{FE}})$. Otherwise, output $\perp$.
 2. If $\text{FC.Verify}(\text{crs}, \sigma, F, 1, \pi_F) = 1$, output μ . Otherwise, output $\perp$.

Security Proof. Regarding security, we establish it via the following theorems. Their proof strategies are similar to those of Theorems 8 and 9.

Theorem 10. *Let $t \in [2]$. If the Special A²BSC scheme with separable keys satisfies IND-Sel-CCA (Definition 10) and t -sided Anon-Sel (Definition 13), the FE scheme is secure (Definition 18), and the FC scheme supports private openings (Definition 16), then our ME construction satisfies CCA- t -sided selective privacy (Definition 7).*

Theorem 11. *If the Special A²BSC scheme with separable keys satisfies EUF-CoSel (Definition 11), and the FC scheme is binding (Definition 15), then our ME construction satisfies authenticity (Definition 8).*

Acknowledgements. This work was supported in part by National Cryptologic Science Fund of China under Grant (2025NCSF01011), in part by the National Natural Science Foundation of China under Grant (62432012, 62425205, 62572495), and in part by the Key Research and Development Program of Shaanxi under Grant (2024GX-ZDCYL-01-08) and the National Key R&D Program of China (2025YFA1017100).

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
2. Agrawal, S., Kumari, S., Yamada, S.: Attribute based encryption for turing machines from lattices. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part III. LNCS, vol. 14922, pp. 352–386. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-68382-4_11
3. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Miller, G.L. (ed.) STOC. pp. 99–108. ACM (1996). <https://doi.org/10.1145/237814.237838>
4. An, J.H., Dodis, Y., Rabin, T.: On the security of joint signature and encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 83–107. Springer (2002). https://doi.org/10.1007/3-540-46035-7_6
5. Ateniese, G., Francati, D., Nuñez, D., Venturi, D.: Match me if you can: Match-making encryption and its applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 701–731. Springer (2019). https://doi.org/https://doi.org/10.1007/978-3-030-26951-7_24
6. Ateniese, G., Francati, D., Nuñez, D., Venturi, D.: Match me if you can: Match-making encryption and its applications. J. Cryptol. **34**(3), 16 (2021). <https://doi.org/10.1007/S00145-021-09381-4>
7. Attrapadung, N., Imai, H.: Dual-policy attribute based encryption. In: Abdalla, M., Pointcheval, D., Fouque, P., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 168–185. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01957-9_11
8. Attrapadung, N., Yamada, S.: Duality in ABE: converting attribute based encryption for dual predicate and dual policy via computational encodings. In: Nyberg, K. (ed.) CT-RSA 2015. LNCS, vol. 9048, pp. 87–105. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16715-2_5
9. Baek, J., Steinfeld, R., Zheng, Y.: Formal proofs for the security of signcryption. J. Cryptol. **20**(2), 203–235 (2007). <https://doi.org/10.1007/S00145-007-0211-0>
10. Boneh, D., Boyen, X.: Efficient selective identity-based encryption without random oracles. J. Cryptol. **24**(4), 659–693 (2011). <https://doi.org/10.1007/S00145-010-9078-6>
11. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. SIAM Journal on Computing **36**(5), 1301–1328 (2007). <https://doi.org/10.1137/S009753970544713X>

12. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_30
13. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer (2011). https://doi.org/10.1007/978-3-642-19571-6_16
14. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer (2007). https://doi.org/10.1007/978-3-540-70936-7_29
15. Bootle, J., Delaplace, C., Espitau, T., Fouque, P., Tibouchi, M.: LWE without modular reduction and improved side-channel attacks against BLISS. In: Peyrin, T., Galbraith, S.D. (eds.) ASIACRYPT 2018, Part I. LNCS, vol. 11272, pp. 494–524. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03326-2_17
16. Boyen, X.: Multipurpose identity-based signcryption (A swiss army knife for identity-based cryptography). In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer (2003). https://doi.org/10.1007/978-3-540-45146-4_23
17. Branco, P., Choudhuri, A.R., Döttling, N., Jain, A., Malavolta, G., Srinivasan, A.: Black-box non-interactive zero knowledge from vector trapdoor hash. In: Fehr, S., Fouque, P. (eds.) EUROCRYPT 2025, Proceedings, Part IV. pp. 64–92. LNCS, Springer (2025). https://doi.org/10.1007/978-3-031-91134-7_3
18. Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer (2004). https://doi.org/10.1007/978-3-540-24676-3_13
19. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_27
20. Chen, L., Malone-Lee, J.: Improved identity-based signcryption. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 362–379. Springer (2005). https://doi.org/10.1007/978-3-540-30580-4_25
21. Chen, Y., Genise, N., Mukherjee, P.: Approximate trapdoors for lattices and smaller hash-and-sign signatures. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019, Part III. LNCS, vol. 11923, pp. 3–32. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34618-8_1
22. Cini, V., Wee, H.: ABE for circuits with poly(λ)-sized keys from LWE. In: FOCS. pp. 435–446. IEEE (2023). <https://doi.org/10.1109/FOCS57990.2023.00032>
23. Cini, V., Wee, H.: Unbounded ABE for circuits from lwe, revisited. In: Chung, K., Sasaki, Y. (eds.) ASIACRYPT 2024, Part IV. LNCS, vol. 15487, pp. 238–267. Springer (2024). https://doi.org/10.1007/978-981-96-0894-2_8
24. Datta, P., Dutta, R., Mukhopadhyay, S.: Functional signcryption: Notion, construction, and applications. In: Au, M.H., Miyaji, A. (eds.) ProvSec 2015. pp. 268–288. LNCS, Springer (2015). https://doi.org/10.1007/978-3-319-26059-4_15
25. Du, R., Zhang, T.: Blockchain-based ciphertext policy-hiding access control scheme. In: Li, F., Liang, K., Lin, Z., Katsikas, S.K. (eds.) Security and Privacy in Communication Networks, SecureComm 2022. pp. 272–289. LNICST, Springer (2022). https://doi.org/10.1007/978-3-031-25538-0_15
26. El Kaafarani, A., Katsumata, S.: Attribute-based signatures for unbounded circuits in the ROM and efficient instantiations from lattices. In: Abdalla, M., Dahab,

- R. (eds.) PKC 2018, Proceedings, Part II. pp. 89–119. LNCS, Springer (2018). https://doi.org/10.1007/978-3-319-76581-5_4
27. Francati, D., Friolo, D., Malavolta, G., Venturi, D.: Multi-key and multi-input predicate encryption from learning with errors. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part III. LNCS, vol. 14006, pp. 573–604. Springer (2023). https://doi.org/10.1007/978-3-031-30620-4_19
28. Francati, D., Friolo, D., Malavolta, G., Venturi, D.: Multi-key and multi-input predicate encryption (for conjunctions) from learning with errors. *J. Cryptol.* **37**(3), 24 (2024). <https://doi.org/10.1007/S00145-024-09504-7>
29. Gagné, M., Narayan, S., Safavi-Naini, R.: Threshold attribute-based signcryption. In: Garay, J.A., Prisco, R.D. (eds.) SCN 2010, Proceedings. LNCS, vol. 6280, pp. 154–171. Springer (2010). https://doi.org/10.1007/978-3-642-15317-4_11
30. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Dwork, C. (ed.) STOC. pp. 197–206. ACM (2008). <https://doi.org/10.1145/1374376.1374407>
31. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_5
32. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) STOC 2013. pp. 555–564. ACM (2013). <https://doi.org/10.1145/2488608.2488678>
33. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: Umans, C. (ed.) FOCS 2017. pp. 612–621. IEEE Computer Society (2017). <https://doi.org/10.1109/FOCS.2017.62>
34. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., di Vimercati, S.D.C. (eds.) CCS 2006. pp. 89–98. ACM (2006). <https://doi.org/10.1145/1180405.1180418>
35. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. *SIAM J. Comput.* **28**(4), 1364–1396 (1999). <https://doi.org/10.1137/S0097539793244708>
36. Hayashi, R., Sakai, Y., Yamada, S.: Attribute-based signatures for circuits with optimal parameter size from standard assumptions. In: Bai, S., Persichetti, E. (eds.) PKC 2026, Part III. LNCS, vol. 16553, pp. 466–495. Springer (2026). https://doi.org/10.1007/978-3-032-26737-5_15
37. Hsieh, Y., Lin, H., Luo, J.: Attribute-based encryption for circuits of unbounded depth from lattices. In: FOCS 2023. pp. 415–434. IEEE (2023). <https://doi.org/10.1109/FOCS57990.2023.00031>
38. Katsumata, S., Yamada, S.: Group signatures without NIZK: from lattices in the standard model. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 312–344. Springer (2019). https://doi.org/10.1007/978-3-030-17659-4_11
39. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. *J. Cryptol.* **26**(2), 191–224 (2013). <https://doi.org/10.1007/S00145-012-9119-4>
40. Lai, J., Deng, R.H., Li, Y.: Expressive CP-ABE with partially hidden access structures. In: Youm, H.Y., Won, Y. (eds.) ASIACCS '12. pp. 18–19. ACM (2012). <https://doi.org/10.1145/2414456.2414465>

41. Ling, S., Nguyen, K., Phan, D.H., Tang, K.H., Wang, H., Xu, Y.: Fully dynamic attribute-based signatures for circuits from codes. In: Tang, Q., Teague, V. (eds.) PKC 2024, Proceedings, Part I. pp. 37–73. LNCS, Springer (2024). https://doi.org/10.1007/978-3-031-57718-5_2
42. Liu, J., Huang, X., Liu, J.K.: Secure sharing of personal health records in cloud computing: Ciphertext-policy attribute-based signcryption. *Future Gener. Comput. Syst.* **52**, 67–76 (2015). <https://doi.org/10.1016/J.FUTURE.2014.10.014>
43. Luo, F., Al-Kuwari, S.M.: Attribute-based signatures from lattices: unbounded attributes and semi-adaptive security. *Des. Codes Cryptogr.* **90**(5), 1157–1177 (2022). <https://doi.org/10.1007/S10623-022-01027-1>
44. Lyubashevsky, V., Nguyen, N.K., Plançon, M.: Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part II. pp. 71–101. LNCS, Springer (2022). https://doi.org/10.1007/978-3-031-15979-4_3
45. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-based signatures. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 376–392. Springer (2011). https://doi.org/10.1007/978-3-642-19074-2_24
46. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41
47. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 89–114. Springer (2019). https://doi.org/10.1007/978-3-030-26948-7_4
48. Rao, Y.S., Dutta, R.: Expressive bandwidth-efficient attribute based signature and signcryption in standard model. In: Susilo, W., Mu, Y. (eds.) ACISP 2014. pp. 209–225. LNCS, Springer (2014). https://doi.org/10.1007/978-3-319-08344-5_14
49. Rao, Y.S., Dutta, R.: Efficient attribute-based signature and signcryption realizing expressive access structures. *Int. J. Inf. Sec.* **15**(1), 81–109 (2016). <https://doi.org/10.1007/S10207-015-0289-6>
50. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) STOC. pp. 84–93. ACM (2005). <https://doi.org/10.1145/1060590.1060603>
51. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27
52. Tsabary, R.: An equivalence between attribute-based signatures and homomorphic signatures, and new constructions for both. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. pp. 489–518. LNCS, Springer (2017). https://doi.org/10.1007/978-3-319-70503-3_16
53. Waters, B.: A new approach for non-interactive zero-knowledge from learning with errors. In: Mohar, B., Shinkar, I., O’Donnell, R. (eds.) STOC 2024. pp. 399–410. ACM (2024). <https://doi.org/10.1145/3618260.3649683>
54. Waters, B., Wee, H., Wu, D.J.: Multi-authority ABE from lattices without random oracles. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part I. LNCS, vol. 13747, pp. 651–679. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22318-1_23
55. Waters, B., Wee, H., Wu, D.J.: New techniques for preimage sampling: Improved nizks and more from LWE. In: Fehr, S., Fouque, P. (eds.) EUROCRYPT 2025,

- Proceedings, Part IV. pp. 3–33. LNCS, Springer (2025). https://doi.org/10.1007/978-3-031-91134-7_1
56. Wee, H.: Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 217–241. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-07085-3_8
 57. Wee, H.: Circuit ABE with $\text{poly}(\text{depth}, \lambda)$ -sized ciphertexts and keys from lattices. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part III. LNCS, vol. 14922, pp. 178–209. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-68382-4_6
 58. Wee, H.: Almost optimal KP and CP-ABE for circuits from succinct LWE. In: Fehr, S., Fouque, P. (eds.) EUROCRYPT 2025, Part III. LNCS, vol. 15603, pp. 34–62. Springer (2025). https://doi.org/10.1007/978-3-031-91131-6_2
 59. Wee, H., Wu, D.J.: Succinct vector, polynomial, and functional commitments from lattices. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part III. LNCS, vol. 14006, pp. 385–416. Springer (2023). https://doi.org/10.1007/978-3-031-30620-4_13
 60. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: Umans, C. (ed.) FOCS 2017. pp. 600–611. IEEE Computer Society (2017). <https://doi.org/10.1109/FOCS.2017.61>
 61. Xiong, H., Zhao, Y., Peng, L., Zhang, H., Yeh, K.: Partially policy-hidden attribute-based broadcast encryption with secure delegation in edge computing. *Future Gener. Comput. Syst.* **97**, 453–461 (2019). <https://doi.org/10.1016/J.FUTURE.2019.03.008>
 62. Yamada, S., Attrapadung, N., Hanaoka, G., Kunihiro, N.: Generic constructions for chosen-ciphertext secure attribute based encryption. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 71–89. Springer (2011). https://doi.org/10.1007/978-3-642-19379-8_5
 63. Yang, R., Au, M.H., Zhang, Z., Xu, Q., Yu, Z., Whyte, W.: Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Proceedings, Part I. pp. 147–175. LNCS, Springer (2019). https://doi.org/10.1007/978-3-030-26948-7_6
 64. Ye, Y., Zhang, L., You, W., Mu, Y.: Secure decentralized access control policy for data sharing in smart grid. In: INFOCOM Workshops 2021. pp. 1–6. IEEE (2021). <https://doi.org/10.1109/INFOCOMWKSHPS51825.2021.9484474>
 65. 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) (2020). <https://doi.org/10.1145/3398036>
 66. Zhang, Y., Zheng, D., Deng, R.H.: Security and privacy in smart health: Efficient policy-hiding attribute-based access control. *IEEE Internet Things J.* **5**(3), 2130–2145 (2018). <https://doi.org/10.1109/JIOT.2018.2825289>
 67. Zheng, Y.: Digital signcryption or how to achieve $\text{cost}(\text{signature} \ \& \ \text{encryption}) \ll \text{cost}(\text{signature}) + \text{cost}(\text{encryption})$. In: Jr., B.S.K. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer (1997). <https://doi.org/10.1007/BFB0052234>

Supplementary Material

A Cryptographic Building Blocks and Lattices Background

A.1 Functional Commitment with Private Opening

Definition 14 (Functional Commitment with Private Opening [59]).

Let λ be a security parameter, $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of functions, where $\mathcal{F}_\lambda$ is defined as $\mathcal{F}_\lambda := \{F : \{0,1\}^\ell \rightarrow \{0,1\}\}$ for $\ell = \ell(\lambda)$. An FC scheme consists of a tuple of efficient algorithms Π_{FC} with the following syntax:

- $\text{Setup}(1^\lambda) \rightarrow \text{crs}$: The algorithm takes the security parameter λ as input and outputs a common reference string crs .
- $\text{Commit}(\text{crs}, \mathbf{x}) \rightarrow (\sigma, \text{st})$: The algorithm takes the common reference string crs and an input $\mathbf{x} \in \{0,1\}^\ell$ as input, and outputs a commitment σ and a state st .
- $\text{Eval}(\text{crs}, \text{st}, F) \rightarrow \pi_F$: The algorithm takes the common reference string crs , a commitment state st and a function $F \in \mathcal{F}$ as input, and outputs an opening π_F .
- $\text{Verify}(\text{crs}, \sigma, F, v, \pi) \rightarrow \{0,1\}$: The algorithm takes the common reference string crs , a commitment σ , a function $F \in \mathcal{F}$, a value $v \in \{0,1\}$ and an opening π , and outputs a bit $b \in \{0,1\}$.

Correctness. For all inputs $\lambda \in \mathbb{N}$, $\mathbf{x} \in \{0,1\}^\ell$, $F \in \mathcal{F}$, we require the following:

$$\Pr \left[\text{Verify}(\text{crs}, \sigma, F, F(\mathbf{x}), \pi_F) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda); \\ (\sigma, \text{st}) \leftarrow \text{Commit}(\text{crs}, \mathbf{x}); \\ \pi_F \leftarrow \text{Eval}(\text{crs}, \text{st}, F) \end{array} \right] = 1 - \text{negl}(\lambda).$$

Definition 15 (Binding). An FC scheme Π_{FC} is said to satisfy binding if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that

$$\Pr [\text{Verify}(\text{crs}, \sigma, F, 0, \pi_0) = 1 = \text{Verify}(\text{crs}, \sigma, F, 1, \pi_1)] = \text{negl}(\lambda),$$

where $\text{crs} \leftarrow \text{Setup}(1^\lambda)$ and $(\sigma, F, \pi_0, \pi_1) \leftarrow \mathcal{A}(1^\lambda, \text{crs})$.

Definition 16 (Private Openings). An FC scheme Π_{FC} is said to satisfy statistical (resp., computational) private openings if for any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, there exists an efficient simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ such that the experiments $\text{EXP}_{\text{FC}, \mathcal{A}}^{\text{Real}}(1^\lambda)$ and $\text{EXP}_{\text{FC}, \mathcal{A}, \mathcal{S}}^{\text{Ideal}}(1^\lambda)$ are statistically (resp., computationally) indistinguishable, where $\text{EXP}_{\text{FC}, \mathcal{A}}^{\text{Real}}(1^\lambda)$ and $\text{EXP}_{\text{FC}, \mathcal{A}, \mathcal{S}}^{\text{Ideal}}(1^\lambda)$ are defined

as follows:

$\text{EXP}_{\text{FC},\mathcal{A}}^{\text{Real}}(1^\lambda)$	$\text{EXP}_{\text{FC},\mathcal{A},\mathcal{S}}^{\text{Ideal}}(1^\lambda)$
1. $\text{crs} \leftarrow \text{Setup}(1^\lambda)$	1. $(\text{crs}, \sigma, \text{st}) \leftarrow \mathcal{S}_0(1^\lambda)$
2. $\mathbf{x} \leftarrow \mathcal{A}_0(1^\lambda, \text{crs})$, where $\mathbf{x} \in \{0, 1\}^\ell$	2. $\mathbf{x} \leftarrow \mathcal{A}_0(1^\lambda, \text{crs})$, where $\mathbf{x} \in \{0, 1\}^\ell$
3. $(\sigma, \text{st}) \leftarrow \text{Commit}(\text{crs}, \mathbf{x})$	3. —
4. $F \leftarrow \mathcal{A}_1(1^\lambda, \sigma)$	4. $F \leftarrow \mathcal{A}_1(1^\lambda, \sigma)$
5. $\pi_F \leftarrow \text{Eval}(\text{crs}, \text{st}, F)$	5. $\pi_F \leftarrow \mathcal{S}_1(\text{crs}, \text{st}, F, F(\mathbf{x}))$
6. $b \leftarrow \mathcal{A}_2(1^\lambda, \pi_F)$	6. $b \leftarrow \mathcal{A}_2(1^\lambda, \pi_F)$
7. Output $b \in \{0, 1\}$	7. Output $b \in \{0, 1\}$

A.2 Functional Encryption

Definition 17 (Functional Encryption [13, 32]). Let λ be a security parameter, $\mathcal{X}_\lambda$ and $\mathcal{Y}_\lambda$ be two finite sets. Define $\{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ as a class of functions with $\mathcal{F}_\lambda = \{F : \mathcal{X}_\lambda \rightarrow \mathcal{Y}_\lambda\}$. An FE scheme consists of a tuple of efficient algorithms Π_{FE} with the following syntax:

- $\text{Setup}(1^\lambda) \rightarrow (\text{mpk}, \text{msk})$: On input the security parameter λ , the Setup algorithm outputs the master public key mpk and the master secret key msk .
- $\text{KeyGen}(\text{mpk}, \text{msk}, F) \rightarrow \text{sk}_F$: On input the master public key mpk , the master secret key msk and a function $F \in \mathcal{F}_\lambda$, the KeyGen algorithm outputs a function secret key sk_F .
- $\text{Enc}(\text{mpk}, x) \rightarrow \text{ct}$: On input the master public key mpk and a message $x \in \mathcal{X}_\lambda$, the Enc algorithm outputs a ciphertext ct .
- $\text{Dec}(\text{sk}_F, \text{ct}) \rightarrow y$: On input a function secret key sk_F and a ciphertext ct , the Dec algorithm outputs a string $y \in \mathcal{Y}_\lambda$.

Correctness. For all inputs $\lambda \in \mathbb{N}$, $x \in \mathcal{X}_\lambda$, $F \in \mathcal{F}_\lambda$, we require the following:

$$\Pr[\text{Dec}(\text{sk}_F, \text{Enc}(\text{mpk}, x)) = F(x)] = 1 - \text{negl}(\lambda).$$

where $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$ and $\text{sk}_F \leftarrow \text{KeyGen}(\text{mpk}, \text{msk}, F)$.

Definition 18 (Security). An FE scheme Π_{FE} is said to be secure if for any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there exists a negligible function $\text{negl}(\cdot)$ such that

$$\left| \Pr[\text{EXP}_{\text{FE},\mathcal{A}}^{(0)}(1^\lambda) = 1] - \Pr[\text{EXP}_{\text{FE},\mathcal{A}}^{(1)}(1^\lambda) = 1] \right| = \text{negl}(\lambda),$$

where for each $b \in \{0, 1\}$, $\text{EXP}_{\text{FE},\mathcal{A}}^{(b)}(1^\lambda)$ is defined as follows:

1. $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$
2. $(x_0, x_1) \leftarrow \mathcal{A}_0^{\mathcal{O}_{\text{KGen}}}(1^\lambda, \text{mpk})$ // $\mathcal{A}$ outputs a pair of messages (x_0, x_1)
3. $\text{ct} \leftarrow \text{Enc}(\text{mpk}, x_b)$
4. $b' \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{KGen}}}(1^\lambda, \text{ct})$ // $\mathcal{A}$ outputs a guess b'
5. Output $b' \in \{0, 1\}$

where Oracles $\mathcal{O}_{\text{KGen}}$ is implemented by $\text{KeyGen}(\text{mpk}, \text{msk}, \cdot)$. $\mathcal{A}$ is called valid if for any F that has been queried to $\mathcal{O}_{\text{KGen}}$, we have $F(x_0) = F(x_1)$.

A.3 Lockable Obfuscation

Below, we recall the notion of LO introduced in [33, 60], which is based on the standard LWE assumption. Intuitively, LO ensures that if the lock string is randomly chosen and not revealed to the adversary, then the adversary learns nothing about the program or the message beyond their sizes.

Definition 19 (Lockable Obfuscation). Let λ be a security parameter, and let $\mathcal{C}_{n,s,d}(\lambda)$ denote the class of circuits of input size $n = n(\lambda)$, output size $s = s(\lambda)$, and depth $d = d(\lambda)$. Let $\mathcal{M} = \mathcal{M}(\lambda)$ be a message space. An LO scheme consists of a tuple of efficient algorithms Π_{LO} with the following syntax:

- $\text{Obf}(1^\lambda, P, \mu, \delta) \rightarrow \tilde{P}$: The algorithm takes the security parameter λ , a program $P \in \mathcal{C}_{n,s,d}(\lambda)$, a message $\mu \in \mathcal{M}$, a lock string $\delta \in \{0,1\}^s$ as input, and outputs a program $\tilde{P}$.
- $\text{Eval}(x, \tilde{P}) \rightarrow \mu$: The algorithm takes a string $x \in \{0,1\}^n$ and a program $\tilde{P}$, and outputs a message $\mu \in \mathcal{M} \cup \{\perp\}$.

Semi-statistical Correctness. An LO scheme Π_{LO} for $\mathcal{C}_{n,s,d}(\lambda)$ is said to satisfy semi-statistical correctness if it satisfies the following properties:

1. For all inputs $\lambda \in \mathbb{N}$, $x \in \{0,1\}^n$, $P \in \mathcal{C}_{n,s,d}(\lambda)$, $\mu \in \mathcal{M}$, if $P(x) = \delta$, we require the following:

$$\Pr [\text{Eval}(x, \text{Obf}(1^\lambda, P, \mu, \delta)) = \mu] = 1.$$

2. For all inputs $\lambda \in \mathbb{N}$, $x \in \{0,1\}^n$, $P \in \mathcal{C}_{n,s,d}(\lambda)$, $\mu \in \mathcal{M}$, if $P(x) \neq \delta$, we require the following:

$$\Pr [\text{Eval}(x, \text{Obf}(1^\lambda, P, \mu, \delta)) = \mu] \leq \text{negl}(\lambda).$$

Definition 20 (Security). An LO scheme Π_{LO} is said to be secure if for any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there exists an efficient simulator Sim such that the experiments $\text{EXP}_{\text{LO},\mathcal{A}}^{\text{Real}}(1^\lambda)$ and $\text{EXP}_{\text{LO},\mathcal{A},\text{Sim}}^{\text{Ideal}}(1^\lambda)$ are computationally indistinguishable, where $\text{EXP}_{\text{LO},\mathcal{A}}^{\text{Real}}$ and $\text{EXP}_{\text{LO},\mathcal{A},\text{Sim}}^{\text{Ideal}}$ are defined as follows:

$\text{EXP}_{\text{LO},\mathcal{A}}^{\text{Real}}(1^\lambda)$	$\text{EXP}_{\text{LO},\mathcal{A},\text{Sim}}^{\text{Ideal}}(1^\lambda)$
1. $(P, \mu, st) \leftarrow \mathcal{A}_0(1^\lambda)$	1. $(P, \mu, st) \leftarrow \mathcal{A}_0(1^\lambda)$
2. $\delta \leftarrow \{0,1\}^s$	2. $\delta \leftarrow \{0,1\}^s$
3. $\tilde{P} \leftarrow \text{Obf}(1^\lambda, P, \mu, \delta)$	3. $\tilde{P} \leftarrow \text{Sim}(1^\lambda, 1^{ P }, 1^{ \mu })$
4. $b \leftarrow \mathcal{A}_1(1^\lambda, \tilde{P})$	4. $b \leftarrow \mathcal{A}_1(1^\lambda, \tilde{P})$
5. Output $b \in \{0,1\}$	5. Output $b \in \{0,1\}$

A.4 Symmetric Key Encryption

Definition 21 (Symmetric Key Encryption). Let λ be a security parameter and $\mathcal{M} = \mathcal{M}(\lambda)$ be a message space. An SKE scheme consists of a tuple of efficient algorithms Π_{SKE} with the following syntax:

- $\text{Setup}(1^\lambda) \rightarrow \text{pp}$: The algorithm takes the security parameter λ as input and outputs a public parameter pp .
- $\text{Gen}(\text{pp}) \rightarrow \text{K}$: The algorithm takes the public parameter pp as input and outputs a symmetric key K .
- $\text{Enc}(\text{K}, \mu) \rightarrow \text{ct}$: The algorithm takes a symmetric key K and a message $\mu \in \mathcal{M}$ as input, and outputs a ciphertext ct .
- $\text{Dec}(\text{K}, \text{ct}) \rightarrow \mu$: The algorithm takes a symmetric key K and a ciphertext ct as input, and outputs a message μ .

Correctness. For all inputs $\lambda \in \mathbb{N}$, $\mu \in \mathcal{M}$, we require the following:

$$\Pr \left[\text{Dec}(\text{K}, \text{ct}) = \mu \mid \begin{array}{l} \text{pp} \leftarrow \text{Setup}(1^\lambda); \\ \text{K} \leftarrow \text{Gen}(\text{pp}); \\ \text{ct} \leftarrow \text{Enc}(\text{K}, \mu) \end{array} \right] = 1 - \text{negl}(\lambda).$$

Definition 22 (IND-CPA). An SKE scheme Π_{SKE} is said to satisfy IND-CPA security if for any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there exists an efficient simulator $\mathcal{S} := \text{CTSamp}$ such that the experiments $\text{EXP}_{\text{SKE}, \mathcal{A}}^{\text{Real}}(1^\lambda)$ and $\text{EXP}_{\text{SKE}, \mathcal{A}, \mathcal{S}}^{\text{Ideal}}(1^\lambda)$ are computationally indistinguishable, where $\text{EXP}_{\text{SKE}, \mathcal{A}}^{\text{Real}}$ and $\text{EXP}_{\text{SKE}, \mathcal{A}, \mathcal{S}}^{\text{Ideal}}$ are defined as follows:

$\text{EXP}_{\text{SKE}, \mathcal{A}}^{\text{Real}}(1^\lambda)$	$\text{EXP}_{\text{SKE}, \mathcal{A}, \mathcal{S}}^{\text{Ideal}}(1^\lambda)$
1. $\text{pp} \leftarrow \text{Setup}(1^\lambda)$ 2. $\text{K} \leftarrow \text{Gen}(\text{pp})$ 3. $\mu \leftarrow \mathcal{A}_0^{\mathcal{O}_{\text{Enc}}}(1^\lambda, \text{pp})$ //where $\mathcal{O}_{\text{Enc}}$ is implemented by $\text{Enc}(\text{K}, \cdot)$ 4. $\text{ct} \leftarrow \text{Enc}(\text{K}, \mu)$ 5. $b \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{Enc}}}(1^\lambda, \text{ct})$ 6. Output $b \in \{0, 1\}$	1. $\text{pp} \leftarrow \text{Setup}(1^\lambda)$ 2. $\text{K} \leftarrow \text{Gen}(\text{pp})$ 3. $\mu \leftarrow \mathcal{A}_0^{\mathcal{O}_{\text{Enc}}}(1^\lambda, \text{pp})$ //where $\mathcal{O}_{\text{Enc}}$ is implemented by $\text{Enc}(\text{K}, \cdot)$ 4. $\text{ct} \leftarrow \text{CTSamp}(\text{pp})$ 5. $b \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{Enc}}}(1^\lambda, \text{ct})$ 6. Output $b \in \{0, 1\}$

A.5 Lattices Background

We first recall the discrete Gaussian distributions over $\mathbb{Z}$.

Definition 23 (Discrete Gaussians Distributions [30]). The discrete Gaussian function over $\mathbb{R}$ with parameter $\chi > 0$ is given by $\rho_\chi(z) := \exp\left(-\pi \frac{|z|^2}{\chi^2}\right)$ with support $\mathbb{R}$. The discrete Gaussian distribution over $\mathbb{Z}$ with parameter χ is given by $\mathcal{D}_{\mathbb{Z}, \chi}(z) := \frac{\rho_\chi(z)}{\sum_{z' \in \mathbb{Z}} \rho_\chi(z')}$ with support $\mathbb{Z}$.

Assumption 1 (SIS [3]). Let λ be a security parameter. For any PPT adversary $\mathcal{A}$, given integers $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, $\beta = \beta(\lambda)$, the $\text{SIS}_{n, m, q, \beta}$ assumption states that

$$\Pr [0 < \|\mathbf{z}\| \leq \beta \wedge \mathbf{A} \cdot \mathbf{z}^\top = \mathbf{0} \mid \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{z} \leftarrow \mathcal{A}(1^\lambda, \mathbf{A})] \leq \text{negl}(\lambda).$$

We rely on the basis-augmented SIS assumption with structured matrices [59]. Unless otherwise noted, BASIS refers to this variant.

Assumption 2 (Basis-Augmented SIS [59]). Let λ be a security parameter. For any PPT adversary $\mathcal{A}$, given integers $n = n(\lambda), m = m(\lambda), q = q(\lambda), \beta = \beta(\lambda), \tau = \tau(\lambda)$ and $\ell = \ell(\lambda)$, the $\text{BASIS}_{n,m,q,\beta,\tau,\ell}$ assumption states that

$$\Pr \left[0 < \|\mathbf{z}\| \leq \beta \wedge \mathbf{A} \cdot \mathbf{z}^\top = \mathbf{0} \left| \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, (\mathbf{C}, \text{aux}) \leftarrow \text{Samp}(1^\lambda, \mathbf{A}), \\ \mathbf{T} \leftarrow \mathbf{C}_\tau^{-1}(\mathbf{G}_{n'}), \\ \mathbf{z} \leftarrow \mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{C}, \mathbf{T}, \text{aux}) \end{array} \right. \right] \leq \text{negl}(\lambda)$$

where the PPT algorithm Samp samples $\mathbf{V}_i \leftarrow \mathbb{Z}_q^{n \times n}$ for all $i \in [\ell]$ and outputs

$$\mathbf{C} = [\text{diag}(\mathbf{V}_1 \mathbf{A}, \dots, \mathbf{V}_\ell \mathbf{A}) \parallel -\mathbf{1}_\ell^\top \otimes \mathbf{G}] \in \mathbb{Z}_q^{n' \times m'}, \text{aux} = (\mathbf{V}_1, \dots, \mathbf{V}_\ell).$$

Assumption 3 (LWE [50]). Let λ be a security parameter. For any PPT adversary $\mathcal{A}$, given integers $n = n(\lambda), m = m(\lambda), q = q(\lambda) > 2, \chi = \chi(\lambda)$, the $\text{LWE}_{n,m,q,\chi}$ assumption states that

$$\left| \Pr \left[\begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \\ \mathbf{c} = \mathbf{s}\mathbf{A} + \mathbf{e} : \\ \mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{c}) = 1 \end{array} \right] - \Pr \left[\begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \\ \mathbf{c} \leftarrow \mathbb{Z}_q^m : \\ \mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{c}) = 1 \end{array} \right] \right| \leq \text{negl}(\lambda).$$

Assumption 4 (ℓ -succinct LWE [57]). Let λ be a security parameter. For any PPT adversary $\mathcal{A}$, given integers $n = n(\lambda), m = m(\lambda), q = q(\lambda) > 2, \chi = \chi(\lambda)$, the (ℓ, τ) -succinct LWE assumption states that

$$\left| \Pr \left[\begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \\ \mathbf{c} = \mathbf{s}\mathbf{A} + \mathbf{e}, \\ \mathbf{W} \leftarrow \mathbb{Z}_q^{\ell n \times m}, \\ \mathbf{T} \leftarrow [\mathbf{I}_\ell \otimes \mathbf{A} \parallel \mathbf{W}]^{-1}(\mathbf{I}_\ell \otimes \mathbf{G}, \tau) : \\ \mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{c}, \mathbf{W}, \mathbf{T}) = 1 \end{array} \right] - \Pr \left[\begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \\ \mathbf{c} \leftarrow \mathbb{Z}_q^m, \\ \mathbf{W} \leftarrow \mathbb{Z}_q^{\ell n \times m}, \\ \mathbf{T} \leftarrow [\mathbf{I}_\ell \otimes \mathbf{A} \parallel \mathbf{W}]^{-1}(\mathbf{I}_\ell \otimes \mathbf{G}, \tau) : \\ \mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{c}, \mathbf{W}, \mathbf{T}) = 1 \end{array} \right] \right| \leq \text{negl}(\lambda).$$

Lemma 1 (Leftover Hash Lemma [35, 59]). Fix some $n, m, q \in \mathbb{N}$, if $m \geq 2n \log q$, the leftover hash lemma states that for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{x} \leftarrow \{0, 1\}^m$ and $\mathbf{y} \leftarrow \mathbb{Z}_q^n$, we have $(\mathbf{A}, \mathbf{A}\mathbf{x}^\top) \approx_s (\mathbf{A}, \mathbf{y}^\top)$.

Lemma 2 (Smudging Lemma [15, 54]). Let λ be a security parameter. Choose any $\varepsilon \in \mathbb{Z}$ such that $\|\varepsilon\| \leq B$. Assume that $\chi \geq B \cdot \lambda^{\omega(1)}$. Then the statistical distance between the distributions $\{e : e \leftarrow \mathcal{D}_{\mathbb{Z},\chi}\}$ and $\{e + \varepsilon : e \leftarrow \mathcal{D}_{\mathbb{Z},\chi}\}$ is $\text{negl}(\lambda)$.

The Gadget Matrix [46]. Let $n, q \in \mathbb{Z}^+$ and $m = n \cdot \lceil \log q \rceil$. The gadget vector and matrix are defined as

$$\mathbf{g}_q = (2^0, 2^1, \dots, 2^{\lceil \log q \rceil - 1}) \in \mathbb{Z}_q^{\lceil \log q \rceil} \quad \text{and} \quad \mathbf{G}_{n,q} = \mathbf{I}_n \otimes \mathbf{g}_q \in \mathbb{Z}_q^{n \times m}.$$

Given $\mathbf{p} \in \mathbb{Z}_q^n$, $\mathbf{G}_{n,q}^{-1}(\mathbf{p})$ is defined as the m -bit vector $(\text{bin}_q(\mathbf{p}[1]), \dots, \text{bin}_q(\mathbf{p}[n]))$, where $\text{bin}_q(a)$ represents converting a value $a \in \mathbb{Z}_q$ into its binary representation. It holds that $\mathbf{G}_{n,q} \cdot \mathbf{G}_{n,q}^{-1}(\mathbf{p}) = \mathbf{p} \pmod q$. This notation applies column-wise when extended to matrices, ensuring that $\mathbf{G}_{n,q} \cdot \mathbf{G}_{n,q}^{-1}(\mathbf{P}) = \mathbf{P} \pmod q$ for any $\mathbf{P} \in \mathbb{Z}_q^{n \times k}$ and $k \in \mathbb{N}$.

Lattice Trapdoors. Let $n, m, q \in \mathbb{N}$ and consider a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. Given any $\mathbf{P} \in \mathbb{Z}_q^{n \times k}$, we use $\mathbf{A}^{-1}(\mathbf{P}, \tau)$ to denote the random variable whose distribution is the discrete Gaussian $\mathcal{D}_{\mathbb{Z}, \tau}^{m \times k}$ conditioned on $\mathbf{A} \cdot \mathbf{A}^{-1}(\mathbf{P}, \tau) = \mathbf{P} \pmod{q}$. If $\mathbf{K} \leftarrow \mathbf{A}^{-1}(\mathbf{P}, \tau)$, then $\|\mathbf{K}\| \leq \tau\sqrt{mk}$ with probability at least $1 - O(2^{-m})$. For any $\mathbf{P}$, a τ -trapdoor $\mathbf{T}_\mathbf{A}$ for $\mathbf{A}$ is a procedure that can sample from the distribution $\mathbf{A}^{-1}(\mathbf{P}, \tau)$ in time $\text{poly}(n, m, k, \log q)$. In a long sequence of works [1, 19, 21, 30, 46], the following properties have been established.

Lemma 3 (Properties of Trapdoors). *There exist the following PPT algorithms:*

1. $\text{TrapGen}(1^n, 1^m, q)$ that outputs $(\mathbf{A}, \mathbf{T}_\mathbf{A})$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ is 2^{-n} -close to uniform for some $m = O(n \log q)$ and $\mathbf{T}_\mathbf{A}$ is a τ -trapdoor for $\mathbf{A}$ with $\tau = \omega(\sqrt{n \log q \log m})$.
2. $\text{SamplePre}(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{P}, \tau)$ that outputs a sample $\mathbf{R}$ from $\mathbf{A}^{-1}(\mathbf{P}, \tau)$ for any $\mathbf{P} \in \mathbb{Z}_q^{n \times k}$.
3. $\text{ExtendLeft}(\mathbf{A}, \mathbf{G}, \mathbf{T}_\mathbf{G}, \mathbf{R})$ that for all $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{R} \in \mathbb{Z}^{m \times m}$, outputs a τ -trapdoor $\mathbf{T}$ for $[\mathbf{A} \parallel \mathbf{A}\mathbf{R} + \mathbf{G}]$, where $\tau = m \cdot \omega(\sqrt{\log m}) \cdot \|\mathbf{R}\|$.

In addition, the above algorithms have the following properties:

- For $\forall k = \text{poly}(\lambda)$, $\mathbf{R}_0 \leftarrow \mathcal{D}_{\mathbb{Z}, \tau}^{m \times k}$, $\mathbf{B}_0 := \mathbf{A}\mathbf{R}_0$, $\mathbf{R}_1 \leftarrow \text{SamplePre}(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}_1, \tau)$ and $\mathbf{B}_1 \leftarrow \mathbb{Z}_q^{n \times k}$, we have

$$(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}_0, \mathbf{R}_0) \approx_s (\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}_1, \mathbf{R}_1).$$

- Given a τ -trapdoor $\mathbf{T}_\mathbf{A}$ for $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, one can obtain a τ -trapdoor $\mathbf{T}_{[\mathbf{A} \parallel \mathbf{B}]}$ for $[\mathbf{A} \parallel \mathbf{B}]$, where $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times m'}$. For simplicity, we write $\mathbf{T}_\mathbf{A}$ as a trapdoor for $[\mathbf{A} \parallel \mathbf{B}]$.

Lattice Evaluation. We review two fundamental algorithms for homomorphic computation on matrices [12, 31].

Lemma 4 (EvalF, EvalFX). *Given inputs $\ell, n, q \in \mathbb{N}$, $m = O(n \log q)$ and a function $F : \{0, 1\}^\ell \rightarrow \{0, 1\}$, where F is computable by circuits of depth d and size s , there exists a pair of efficient algorithms (EvalF, EvalFX) with the following properties.*

- $\text{EvalF}(\mathbf{A}, F) \rightarrow \mathbf{A}_F$: Taking a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m\ell}$ and a function F as inputs, it outputs $\mathbf{A}_F \in \mathbb{Z}_q^{n \times m}$.
- $\text{EvalFX}(\mathbf{A}, F, \mathbf{x}) \rightarrow \mathbf{H}_{F, \mathbf{x}}$: Taking a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m\ell}$, a function F and $\mathbf{x} \in \mathbb{Z}^\ell$ as inputs, it outputs $\mathbf{H}_{F, \mathbf{x}} \in \mathbb{Z}^{m\ell \times m}$ with $\|\mathbf{H}_{F, \mathbf{x}}\| = m^{O(d)}s$.
- It holds that $(\mathbf{A} - \mathbf{x} \otimes \mathbf{G}_{n, q}) \cdot \mathbf{H}_{F, \mathbf{x}} = \mathbf{A}_F - F(\mathbf{x}) \cdot \mathbf{G}_{n, q} \pmod{q}$.

Succinct Commitments. Below we recall a succinct commitment scheme recently introduced by Wee [58], which is used to handle the relation between attributes and policies under succinct LWE.

Lemma 5 (Commitment to Vectors). *Given $n, m, q \in \mathbb{N}$ where $m \geq 2n \log q$, there exist the following efficient and deterministic algorithms:*

- $\text{Com}^{\text{vc}}(\text{pp}, \mathbf{x}) \rightarrow \mathbf{C}$: Taking public parameter pp and a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$ as inputs, it outputs $\mathbf{C} \in \mathbb{Z}_q^{n \times m}$.
- $\text{Ver}^{\text{vc}}(\text{pp}, 1^\ell) \rightarrow \mathbf{P}_\ell$: Taking public parameter pp and a length parameter $\ell \in \mathbb{N}$ as inputs, it outputs $\mathbf{P}_\ell \in \mathbb{Z}_q^{m \times m^\ell}$.
- $\text{Open}^{\text{vc}}(\text{pp}, \mathbf{x}) \rightarrow \mathbf{Z}$: Taking public parameter pp and a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$ as inputs, it outputs $\mathbf{Z} \in \mathbb{Z}_q^{m \times m^\ell}$.

In addition, for all $\text{pp} := (\mathbf{A}, \mathbf{W}, \mathbf{T})$ s.t. $[\mathbf{I}_{2m^2} \otimes \mathbf{A} \parallel \mathbf{W}] \cdot \mathbf{T} = \mathbf{I}_{2m^2} \otimes \mathbf{G}$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{W} \in \mathbb{Z}_q^{2m^2 n \times m}$, $\mathbf{T} \in \mathbb{Z}_q^{(2m^2+1)m \times 2m^3}$, the above algorithms have the following properties:

$$\mathbf{C} \cdot \mathbf{P}_\ell = \mathbf{x} \otimes \mathbf{G} - \mathbf{A} \cdot \mathbf{Z},$$

where $\|\mathbf{P}_\ell\| \leq O(\|\mathbf{T}\| \cdot m^4 \log q)$ and $\|\mathbf{Z}\| \leq O(\|\mathbf{T}\| \cdot m^7 \log q \cdot \log \ell)$.

Lemma 6 (Commitment to Circuits). *Let $\mathcal{F}_{d,\ell,s}(\lambda)$ denote the class of circuits of depth $d = d(\lambda)$, input size $\ell = \ell(\lambda)$, and size $s = s(\lambda)$. Given $n, m, q \in \mathbb{N}$ where $m \geq 2n \log q$, there exist the following efficient algorithms:*

- $\text{Com}^c(\text{pp}, F)$: Taking public parameter pp and a function $F \in \mathcal{F}_{d,\ell,s}$ as inputs, it outputs $\mathbf{C}_F \in \mathbb{Z}_q^{n \times m}$.
- $\text{Ver}^c(\text{pp}, \mathbf{x}, 1^d) \rightarrow \mathbf{P}_{x,d}$: Taking public parameter pp , an input $\mathbf{x} \in \{0, 1\}^\ell$ and a depth parameter d as inputs, it outputs $\mathbf{P}_{x,d} \in \mathbb{Z}_q^{m \times m}$.
- $\text{Open}^c(\text{pp}, F, \mathbf{x}) \rightarrow \mathbf{Z}_{F,x}$: Taking public parameter pp , a function $F \in \mathcal{F}_{d,\ell,s}$ and an input $\mathbf{x} \in \{0, 1\}^\ell$ as inputs, it outputs $\mathbf{Z}_{F,x} \in \mathbb{Z}_q^{m \times m}$.

In addition, for all $\text{pp} := (\mathbf{A}, \mathbf{W}, \mathbf{T})$ set as above, the above algorithms have the following properties:

$$\mathbf{C}_F \cdot \mathbf{P}_{x,d} = F(\mathbf{x})\mathbf{G} - \mathbf{A} \cdot \mathbf{Z}_{F,x},$$

where $\|\mathbf{P}_{x,d}\| \leq O(\|\mathbf{T}\| \cdot m^9)$ and $\|\mathbf{Z}_{F,x}\| \leq \ell \cdot O(\|\mathbf{T}\| \cdot m^{12})^d$.

B Error Bound and Parameters

Error Bound. The error term is bounded as follows. Observe that

$$\begin{aligned} \|\mathbf{e}'_1\| &= \left\| [\mathbf{e}_0 \parallel \mathbf{e}_1] \cdot \begin{pmatrix} -\mathbf{Z}_{\mathbf{x}} \\ -\mathbf{P}_{\ell}^{(1)} \end{pmatrix} \cdot \mathbf{H}_{F,\mathbf{x}} \right\| \leq (\chi + \chi') \tau_0 m^{O(d)} s \cdot \text{poly}(\lambda, m), \\ \|\mathbf{e}'_2\| &= \left\| [\mathbf{e}_0 \parallel \mathbf{e}_2] \cdot \begin{pmatrix} -\mathbf{Z}_{G,y} \\ -\mathbf{P}_{y,d}^{(2)} \end{pmatrix} \right\| \leq (\chi + \chi') \ell (\tau_0 \cdot m)^{O(d)} \cdot \text{poly}(\lambda, m), \\ \|\mathbf{e}'_3\| &= \left\| [\mathbf{e}_0 \parallel \mathbf{e}_3] \cdot \begin{pmatrix} -\mathbf{Z}_{\text{ovk}} \\ -\mathbf{P}_{\text{ovk}}^{(3)} \end{pmatrix} \cdot \mathbf{H}_{F_{\text{ovk}}, \text{ovk}} \right\| \leq (\chi + \chi') \tau_0 m^{O(d)} s \cdot \text{poly}(\lambda, m). \end{aligned}$$

Then, we obtain a bound for the error term as β , namely

$$\|\mathbf{e}_{\text{total}}\| = \|\mathbf{e} - [\mathbf{e}_0 \parallel \mathbf{e}'_1 \parallel \mathbf{e}'_2 \parallel \mathbf{e}'_3] \cdot \mathbf{D}'\| \leq (\chi + \chi') \ell (\tau_0 \cdot m)^{O(d)} s \tau_2 \cdot \text{poly}(\lambda, m) \leq \beta.$$

Parameters. We set the parameters as follows.

$$\begin{aligned} n &= \text{poly}(\lambda, d), m = O(n \log q), m' = n \lceil \log q \rceil, \\ \tau_0 &= \text{poly}(\lambda, m), \tau_1 \geq \ell(\tau_0 \cdot m)^{O(d)} s \cdot \text{poly}(m, \lambda), \tau_2 \geq \tau_1 \cdot \text{poly}(\lambda, m), \\ \chi &= \text{poly}(\lambda, n), \chi' \geq \chi \tau_0 \lambda^{\omega(1)}, q/4 \geq (\chi + \chi') \tau_1 \tau_2 \cdot \text{poly}(\lambda, m). \end{aligned}$$

C Missing Proofs

C.1 Proof of Theorem 2

Proof. Before presenting the hybrid experiments, we first define the event **Forge** and bound its probability. Let $\text{ct}^* = (\text{ovk}^*, \tilde{P}^*, \mathbf{Q}^*, \sigma^*)$ be the challenge ciphertext generated by the challenger $\mathcal{C}$. We define **Forge** as: the adversary $\mathcal{A}$ submits a ciphertext $(\text{ovk}^*, \tilde{P}, \mathbf{Q}, \sigma)$ to its decryption oracle $\mathcal{O}_{\text{USC}}$ such that $(\text{ovk}^*, \tilde{P}, \mathbf{Q}, \sigma) \neq (\text{ovk}^*, \tilde{P}^*, \mathbf{Q}^*, \sigma^*)$ and $1 \leftarrow \text{OTS.Vrfy}_{\text{ovk}^*}((\tilde{P} \parallel \mathbf{Q}), \sigma)$. Clearly, we can use $\mathcal{A}$ to break the OTS with probability $\Pr[\text{Forge}]$. Since the OTS scheme is strongly unforgeable, it must hold that $\Pr[\text{Forge}] = \text{negl}(\lambda)$. We then define the hybrid experiments as follows. $\mathcal{A}$ begins by selecting a challenge ciphertext attribute $(\mathbf{x}^*, G^*)$.

Hyb₀ In this hybrid, the challenge ciphertext ct^* is generated honestly, as in the game given in Definition 10.

Hyb₁ This is the same as **Hyb₀**, except that $\mathcal{C}$ samples $\mathbf{P}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3$ as follows: sample $\mathbf{U}_0 \leftarrow \{0, 1\}^{m \times \kappa}, \mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3 \leftarrow \{0, 1\}^{m \times m}$ and set

$$\begin{aligned} \mathbf{P} &:= \mathbf{A}_r \mathbf{U}_0, & \mathbf{A}_1 &:= \mathbf{A}_r \mathbf{U}_1 - \mathbf{C}_{\mathbf{x}^*}, \\ \mathbf{A}_2 &:= \mathbf{A}_r \mathbf{U}_2 - \mathbf{C}_{G^*}, & \mathbf{A}_3 &:= \mathbf{A}_r \mathbf{U}_3 - \mathbf{C}_{\text{ovk}^*}, \end{aligned}$$

where ovk^* is the verification key of OTS selected from the challenge ciphertext (the key generation for OTS can be pre-computed by $\mathcal{C}$ at the initial stage). By the left-over hash lemma (Lemma 1), we have **Hyb₀** $\approx_s$ **Hyb₁**.

Hyb₂ This is the same as **Hyb₁**, except that $\mathcal{C}$ computes $\tilde{\text{ct}}$ during the challenge phase as follows:

$$\mathbf{c}_1 := \mathbf{c}_0 \mathbf{U}_1 + \mathbf{e}_1, \mathbf{c}_2 := \mathbf{c}_0 \mathbf{U}_2 + \mathbf{e}_2, \mathbf{c}_3 := \mathbf{c}_0 \mathbf{U}_3 + \mathbf{e}_3, \mathbf{c} := \mathbf{c}_0 \mathbf{U}_0 + \mathbf{e}.$$

Observe that

$$\begin{aligned} \mathbf{c}_0 \mathbf{U}_1 + \mathbf{e}_1 &= (\mathbf{s} \mathbf{A}_r + \mathbf{e}_0) \mathbf{U}_1 + \mathbf{e}_1 \approx_s \mathbf{s} \mathbf{A}_r \mathbf{U}_1 + \mathbf{e}_1 = \mathbf{s}(\mathbf{A}_1 + \mathbf{C}_{\mathbf{x}^*}) + \mathbf{e}_1, \\ \mathbf{c}_0 \mathbf{U}_2 + \mathbf{e}_2 &= (\mathbf{s} \mathbf{A}_r + \mathbf{e}_0) \mathbf{U}_2 + \mathbf{e}_2 \approx_s \mathbf{s} \mathbf{A}_r \mathbf{U}_2 + \mathbf{e}_2 = \mathbf{s}(\mathbf{A}_2 + \mathbf{C}_{G^*}) + \mathbf{e}_2, \\ \mathbf{c}_0 \mathbf{U}_3 + \mathbf{e}_3 &= (\mathbf{s} \mathbf{A}_r + \mathbf{e}_0) \mathbf{U}_3 + \mathbf{e}_3 \approx_s \mathbf{s} \mathbf{A}_r \mathbf{U}_3 + \mathbf{e}_3 = \mathbf{s}(\mathbf{A}_3 + \mathbf{C}_{\text{ovk}^*}) + \mathbf{e}_3, \\ \mathbf{c}_0 \mathbf{U}_0 + \mathbf{e} &= (\mathbf{s} \mathbf{A}_r + \mathbf{e}_0) \mathbf{U}_0 + \mathbf{e} \approx_s \mathbf{s} \mathbf{A}_r \mathbf{U}_0 + \mathbf{e} = \mathbf{s} \mathbf{P} + \mathbf{e}, \end{aligned}$$

where the $\approx_s$ relies on noise smudging (Lemma 2), namely

$$\mathbf{e}_1 \approx_s \mathbf{e}_1 + \mathbf{e}_0 \mathbf{U}_1, \mathbf{e}_2 \approx_s \mathbf{e}_2 + \mathbf{e}_0 \mathbf{U}_2, \mathbf{e}_3 \approx_s \mathbf{e}_3 + \mathbf{e}_0 \mathbf{U}_3, \mathbf{e} \approx_s \mathbf{e} + \mathbf{e}_0 \mathbf{U}_0,$$

which in turn follows from $\chi' \geq \chi \tau_0 \lambda^{\omega(1)}$. We have **Hyb₁** $\approx_s$ **Hyb₂**.

Hyb₃ This is the same as **Hyb₂**, except that $\mathcal{C}$ performs the following in the dExtract algorithm:

- a) if $F(\mathbf{x}^*) = 1$, compute $\mathbf{H}_{F, \mathbf{x}^*} := \text{EvalFX}(\mathbf{B}_1, F, \mathbf{x}^*)$ and sample $\mathbf{D} \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F \parallel \mathbf{B}_y], \begin{pmatrix} (\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)}) \mathbf{H}_{F, \mathbf{x}^*} \\ \mathbf{0} \\ \mathbf{I}_m \end{pmatrix}, \mathbf{G}, \tau_1)$ instead of $\mathbf{D} \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F \parallel \mathbf{B}_y], \mathbf{T}_{\mathbf{A}_r}, \mathbf{G}, \tau_1)$. This is due to the fact that

$$[\mathbf{A}_r \parallel \overbrace{(\mathbf{A}_1 + \mathbf{C}_{\mathbf{x}^*})}^{\mathbf{A}_r \mathbf{U}_1}] \cdot \begin{pmatrix} -\mathbf{Z}_{\mathbf{x}^*} \\ -\mathbf{P}_\ell^{(1)} \end{pmatrix} \cdot \mathbf{H}_{F, \mathbf{x}^*} = \mathbf{A}_r \cdot (-(\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)})) \cdot \mathbf{H}_{F, \mathbf{x}^*} \\ = \mathbf{B}_F - F(\mathbf{x}^*) \mathbf{G}.$$

Thus, we have $[\mathbf{A}_r \parallel \mathbf{B}_F] \cdot \begin{pmatrix} (\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)}) \mathbf{H}_{F, \mathbf{x}^*} \\ \mathbf{0} \\ \mathbf{I}_m \end{pmatrix} = F(\mathbf{x}^*) \mathbf{G}$ as $F(\mathbf{x}^*) = 1$, where $\|(\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)}) \mathbf{H}_{F, \mathbf{x}^*}\| = \tau_0 \cdot m^{O(d)} s \cdot \text{poly}(m, \lambda)$.

- b) if $G^*(\mathbf{y}) = 1$, compute $\mathbf{Z}_{G^*, \mathbf{y}} := \text{Open}^c(\text{pp}_{\text{com}}, G^*, \mathbf{y})$ and sample $\mathbf{D} \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F \parallel \mathbf{B}_y], \begin{pmatrix} \mathbf{Z}_{G^*, \mathbf{y}} + \mathbf{U}_2 \mathbf{P}_{y,d}^{(2)} \\ \mathbf{0} \\ \mathbf{I}_m \end{pmatrix}, \mathbf{G}, \tau_1)$ instead of $\mathbf{D} \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F \parallel \mathbf{B}_y], \mathbf{T}_{\mathbf{A}_r}, \mathbf{G}, \tau_1)$. This is due to the fact that

$$[\mathbf{A}_r \parallel \overbrace{(\mathbf{A}_2 + \mathbf{C}_{G^*})}^{\mathbf{A}_r \mathbf{U}_2}] \cdot \begin{pmatrix} -\mathbf{Z}_{G^*, \mathbf{y}} \\ -\mathbf{P}_{y,d}^{(2)} \end{pmatrix} = \mathbf{A}_r \cdot (-(\mathbf{Z}_{G^*, \mathbf{y}} + \mathbf{U}_2 \mathbf{P}_{y,d}^{(2)})) \\ = \mathbf{B}_y - G^*(\mathbf{y}) \mathbf{G}.$$

Thus, we have $[\mathbf{A}_r \parallel \mathbf{B}_y] \cdot \begin{pmatrix} \mathbf{Z}_{G^*, \mathbf{y}} + \mathbf{U}_2 \mathbf{P}_{y,d}^{(2)} \\ \mathbf{0} \\ \mathbf{I}_m \end{pmatrix} = G^*(\mathbf{y}) \mathbf{G}$ as $G^*(\mathbf{y}) = 1$, where $\|\mathbf{Z}_{G^*, \mathbf{y}} + \mathbf{U}_2 \mathbf{P}_{y,d}^{(2)}\| = \ell(\tau_0 \cdot m)^{O(d)} \cdot \text{poly}(m, \lambda)$.

By the trapdoor sampling lemma, we have **Hyb₂** $\approx_s$ **Hyb₃**.

Hyb₄ This is the same as **Hyb₃**, except that when $\mathcal{A}$ submits a query to the decryption oracle $\mathcal{O}_{\text{USC}}(\text{ct}, (F, \mathbf{y}))$ with $\text{ct} = (\text{ovk}, \tilde{P}, \mathbf{Q}, \sigma)$, $\mathcal{C}$ responds as follows:

- a) if $\text{OTS.Vrfy}_{\text{ovk}}((\tilde{P} \parallel \mathbf{Q}), \sigma) \rightarrow 0$, return $\perp$ directly.
b) if the OTS verification is valid and $\text{ovk} = \text{ovk}^*$ (i.e., the forgery event **Forge** occurs), abort the simulation and return a random bit.
c) if the OTS verification is valid and $\text{ovk} \neq \text{ovk}^*$ (i.e., $F_{\text{ovk}^*}(\text{ovk}) = 1$), proceed as follows: compute $\mathbf{H}_{F_{\text{ovk}^*}, \text{ovk}} = \text{EvalFX}(\mathbf{B}_3, F_{\text{ovk}^*}, \text{ovk})$, sample

$$\mathbf{D}' \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F \parallel \mathbf{B}_y \parallel \mathbf{B}_{F_{\text{ovk}^*}}], \begin{pmatrix} (\mathbf{Z}_{\text{ovk}} + \mathbf{U}_3 \mathbf{P}_{\text{ovk}}^{(3)}) \mathbf{H}_{F_{\text{ovk}^*}, \text{ovk}} \\ \mathbf{0} \\ \mathbf{I}_m \end{pmatrix}, \mathbf{P}, \tau_1)$$

instead of $\mathbf{D}' \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F \parallel \mathbf{B}_y \parallel \mathbf{B}_{F_{\text{ovk}}}], \begin{pmatrix} \mathbf{P} \\ \mathbf{0} \end{pmatrix}, \mathbf{P}, \tau_1)$. This is due to the fact that

$$[\mathbf{A}_r \parallel \underbrace{(\mathbf{A}_3 + \mathbf{C}_{\text{ovk}})}_{\mathbf{A}_r \mathbf{U}_3}] \cdot \begin{pmatrix} -\mathbf{Z}_{\text{ovk}} \\ -\mathbf{P}_{\text{ovk}}^{(3)} \end{pmatrix} \cdot \mathbf{H}_{F_{\text{ovk}^*}, \text{ovk}} = \mathbf{A}_r \cdot (-(\mathbf{Z}_{\text{ovk}} + \mathbf{U}_3 \mathbf{P}_{\text{ovk}}^{(3)})) \cdot \mathbf{H}_{F_{\text{ovk}^*}, \text{ovk}} \\ = \mathbf{B}_{F_{\text{ovk}^*}} - F_{\text{ovk}^*}(\text{ovk}) \mathbf{G}$$

Thus, we have $[\mathbf{A}_r \parallel \mathbf{B}_{F_{\text{ovk}^*}}] \cdot \begin{pmatrix} (\mathbf{Z}_{\text{ovk}} + \mathbf{U}_3 \mathbf{P}_{\text{ovk}}^{(3)}) \mathbf{H}_{F_{\text{ovk}^*, \text{ovk}}} \\ \mathbf{I}_m \end{pmatrix} = F_{\text{ovk}^*}(\text{ovk}) \mathbf{G}$ as $F_{\text{ovk}^*}(\text{ovk}) = 1$, where $\|(\mathbf{Z}_{\text{ovk}} + \mathbf{U}_3 \mathbf{P}_{\text{ovk}}^{(3)}) \mathbf{H}_{F_{\text{ovk}^*, \text{ovk}}}\| = \tau_0 \cdot m^{O(d)} s \cdot \text{poly}(m, \lambda)$. Finally, substitute $\mathbf{D}'$ into the **Unsigncrypt** algorithm, and then return the output to $\mathcal{A}$.

By the trapdoor sampling lemma (Lemma 3), we have $\mathbf{Hyb}_3 \approx_s \mathbf{Hyb}_4$.

Hyb₅ This is the same as **Hyb₄**, except that $\mathcal{C}$ samples $\mathbf{c}_0 \leftarrow \mathbb{Z}_q^m$. By the $(2m^2, \tau_0)$ -succinct LWE assumption, we have $\mathbf{Hyb}_4 \approx_c \mathbf{Hyb}_5$.

Hyb₆ This is the same as **Hyb₅**, except that $\mathcal{C}$ samples $\mathbf{c} \leftarrow \mathbb{Z}_q^s$. By the left-over hash lemma, $(\mathbf{A}_r, \mathbf{c}_0, \mathbf{A}_r \mathbf{U}_0, \mathbf{c}_0 \mathbf{U}_0)$ is statistically indistinguishable from random. Thus, we have $\mathbf{Hyb}_5 \approx_s \mathbf{Hyb}_6$. Observe that in this hybrid world, even if the lock $\delta \leftarrow \{0, 1\}^\kappa$ is replaced with $\mathbf{0}$, this swap remains indistinguishable from $\mathcal{A}$'s view.

Hyb₇ This is the same as **Hyb₆**, except that $\mathcal{C}$ no longer computes $\tilde{P} \leftarrow \text{LO.Obf}(1^\lambda, \text{Subroutine}(\cdot, \tilde{\text{ct}}), \text{bin}_q(\mathbf{u}) \parallel \mu, \delta)$ honestly during the challenge phase. Instead, it uses the simulator to simulate

$$\tilde{P} \leftarrow \text{Sim} \left(1^\lambda, 1^{|\text{Subroutine}(\cdot, \tilde{\text{ct}})|}, 1^{u+|\mu|} \right).$$

We claim that $\mathbf{Hyb}_6 \approx_c \mathbf{Hyb}_7$. We construct an algorithm $\mathcal{B}$: if there exists a PPT adversary $\mathcal{A}$ that can distinguish **Hyb₆** from **Hyb₇** with non-negligible probability, then $\mathcal{B}$ can break the LO security. $\mathcal{B}$ is defined as follows:

1. Initially, receive the challenge ciphertext attribute $(\mathbf{x}^*, G^*)$ chosen by $\mathcal{A}$.
2. Generate mpk as in **Hyb₆**, and send it to $\mathcal{A}$.
3. $\mathcal{A}$ is allowed to access oracles $\mathcal{O}_{\text{SE}}, \mathcal{O}_{\text{dE}}, \mathcal{O}_{\text{SC}}, \mathcal{O}_{\text{USC}}$, where $\mathcal{B}$ responds in the same manner as in **Hyb₆**.
4. Receive the challenge (μ_0, μ_1) from $\mathcal{A}$.
5. Generate $\tilde{\text{ct}}$ as in **Hyb₆** and send it to the LO challenger.
6. Receive the obfuscated program $\tilde{P}^*$ from the LO challenger.
7. Send the challenge ciphertext $\text{ct}^* := (\text{ovk}^*, \tilde{P}^*, \mathbf{Q}^*, \sigma^*)$ to $\mathcal{A}$, where $\tilde{P}^*$ is from the LO challenger whereas all other components of ct^* are computed by $\mathcal{B}$ as in **Hyb₆**.
8. Continue responding to subsequent queries from $\mathcal{A}$ as in Step 3.
9. Finally, return $\mathcal{A}$'s guess.

If $\mathcal{A}$ guesses that $\mathcal{B}$ simulates **Hyb₆**, the obfuscated program $\tilde{P}^*$ from the LO challenger is the obfuscated Subroutine circuit; if $\mathcal{A}$ guesses that $\mathcal{B}$ simulates **Hyb₇**, $\tilde{P}^*$ from the LO challenger is sampled via **Sim** (simulator). Thus, if $\mathcal{A}$ can distinguish **Hyb₆** from **Hyb₇** with non-negligible probability, then $\mathcal{B}$ can also break the LO security with non-negligible probability.

In **Hyb₇**, since the challenge bit α is not involved, the advantage $\text{Adv}_{\Pi, \mathcal{A}}^{\text{IND}}(1^\lambda)$ of $\mathcal{A}$ is negligible. Note that throughout the game, $\mathcal{O}_{\text{SE}}, \mathcal{O}_{\text{SC}}$ respond to queries honestly, and are independent of the challenge bit. This concludes the proof. $\square$

C.2 Proof of Theorem 3

Proof. We show that if there exists any PPT adversary $\mathcal{A}$ with advantage ϵ in attacking no-signing EUF-CoSel of our A²BSC, then a PPT simulator $\mathcal{S}$ can be constructed that solves the BASIS problem with probability at least $\epsilon - \text{negl}(\lambda)$. Thus, under the hardness assumption of the BASIS problem, it follows that this advantage ϵ is negligible. The proof proceeds by analyzing the following sequence of games. Let Forge_i represents the success probability of $\mathcal{A}$ in Game_i .

Game₀ This is the same as the game specified in Definition 12. Thus, we have $\Pr[\text{Forge}_0] = \epsilon$.

Game₁ This is the same as **Game₀**, except that we sample $\vec{\mathbf{M}} = [\mathbf{M}_0 || \dots || \mathbf{M}_L]$, $\vec{\mathbf{B}} = [\mathbf{B}_{\ell_0}^{(1)} || \dots || \mathbf{B}_{\ell_0}^{(N)}]$ as follows:

1. Sample $\mathbf{U}_0 \leftarrow \mathcal{D}_{\mathbb{Z}, \tau_1}^{m \times m'}$, $\mathbf{U}_k \leftarrow \mathcal{D}_{\mathbb{Z}, \tau_1}^{m \times m'}$ for all $k \in [L]$, and set $\mathbf{M}_0 := \mathbf{A}_s \mathbf{U}_0$, $\mathbf{M}_k := \mathbf{A}_s \mathbf{U}_k$ for $k \in [L]$.
2. Sample $\bar{\mathbf{U}}_i \leftarrow \mathcal{D}_{\mathbb{Z}, \tau_1}^{m \times m' \ell_0}$ for $i \in [N]$, and set

$$\mathbf{B}_{\ell_0}^{(i)} = \begin{cases} \mathbf{A}_s \bar{\mathbf{U}}_i + \mathbf{z}_0^{(i)} \otimes \mathbf{G}, & \text{if } i \in \mathcal{U} \\ \mathbf{A}_s \bar{\mathbf{U}}_i + \mathbf{z}_0^{(i_{\min})} \otimes \mathbf{G}, & \text{if } i \notin \mathcal{U} \end{cases}$$

where $i_{\min} := \min\{i | i \in \mathcal{U}\}$.

In this game, $\mathcal{C}$ responds to $\mathcal{A}$'s signing attribute key query with $\{\text{ssk}_{\mathbf{z}_0^{(i)}} := \bar{\mathbf{U}}_i\}_{i \in \mathcal{U}}$. **Game₀** $\approx_s$ **Game₁** follows readily from Lemma 3. Thus, we have $\Pr[\text{Forge}_1] \geq \Pr[\text{Forge}_0] - \text{negl}(\lambda)$.

Game₂ This is the same as **Game₁**, except that the matrix $\mathbf{A}_s$ is sampled uniformly from $\mathbb{Z}_q^{n \times m}$, instead of being generated via the trapdoor algorithm $(\mathbf{A}_s, \mathbf{T}_{\mathbf{A}_s}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$. **Game₁** $\approx_s$ **Game₂** follows from Item 1 of Lemma 3. Thus, we have $\Pr[\text{Forge}_2] \geq \Pr[\text{Forge}_1] - \text{negl}(\lambda)$.

Reduction. We replace the challenger in **Game₂** with a PPT algorithm $\mathcal{S}$ that solves the BASIS problem with probability $\Pr[\text{Forge}_2] - \text{negl}(\lambda)$. By the above analysis, this probability is at least $\epsilon - \text{negl}(\lambda)$. $\mathcal{S}$ is defined as follows:

1. **Init:** $\mathcal{S}$ receives an instance $(\mathbf{A}_s, \mathbf{C}_{\ell_1}, \mathbf{T}', \text{aux} = (\mathbf{V}_1, \dots, \mathbf{V}_{\ell_1}))$ of the BASIS problem. Then, it runs $\mathcal{A}$ with λ as input and obtains $(\{\mathbf{z}_0^{(i)}\}_{i \in [N]}, \mathcal{U} \subseteq [N])$ from $\mathcal{A}$.
2. **Setup:** $\mathcal{S}$ constructs the master public key mpk by embedding the problem instance, where $\vec{\mathbf{M}}, \vec{\mathbf{B}}$ are sampled as in **Game₁**, and the remaining components follow the real world. Output mpk to $\mathcal{A}$.
3. **sExtract Query** $\mathcal{O}_{\text{SE}}(\mathbf{z}_0^{(i)})$: $\mathcal{S}$ replies with $\text{ssk}_{\mathbf{z}_0^{(i)}} := \bar{\mathbf{U}}_i$, where $\bar{\mathbf{U}}_i$ is sampled as in **Game₁**.
4. **dExtract Query** $\mathcal{O}_{\text{DE}}(F, \mathbf{y})$: $\mathcal{S}$ honestly replies with $\text{dsk}_{F, \mathbf{y}} \leftarrow \text{dExtract}(\text{mpk}, \text{msk}, F, \mathbf{y})$.
5. **Unsigncrypt Query** $\mathcal{O}_{\text{USC}}(\text{ct}, (F, \mathbf{y}))$: $\mathcal{S}$ responds to queries honestly as in the real world.

6. **Forgery:** $\mathcal{A}$ returns a forged ciphertext $\text{ct}^* := (\text{ovk}^*, \tilde{P}^*, \mathbf{Q}^*, \sigma^*)$ under the tuple $(C^*, (\mathbf{x}^*, G^*))$ and message μ^* . $\mathcal{S}$ first honestly executes the `Unsigncrypt` algorithm to recover a signature $\mathbf{u}$ on $\mathbf{m}^* = [\mu^* \parallel \text{ovk}^*]$ under C^* . Then, $\mathcal{S}$ checks whether $\mathcal{A}$ wins the game: if $\mathcal{A}$ fails, $\mathcal{S}$ aborts immediately; otherwise, $\mathcal{S}$ computes $\mathbf{S}_i := [\bar{\mathbf{U}}_i \parallel \mathbf{R}_R] \mathbf{H}_{C^*, [\mathbf{z}_0^{(i)} \parallel \mathbf{z}_1]}$ for $i \in [N]$ and outputs

$$\mathbf{d}^\top := \left[\mathbf{I}_m \parallel \mathbf{S}_1 \parallel \dots \parallel \mathbf{S}_N \parallel \left(\mathbf{U}_0 + \sum_{k \in [L]} \mathbf{m}^*[k] \mathbf{U}_k \right) \right] \cdot \mathbf{u}^\top$$

as the solution to the $\text{BASIS}_{n,m,q,\beta_{\text{SIS}},\tau_1,\ell_1}$ problem.

Analysis. We show that algorithm $\mathcal{S}$ can solve the BASIS problem with a probability, where the gap between this probability and $\Pr[\text{Forge}_2]$ is negligible. Clearly, $\mathcal{S}$ perfectly simulates Game_2 . In Game_2 , observe that $C^*(\mathbf{z}_0^{(i)} \parallel \mathbf{z}_1) = 0$ for all $i \in \mathcal{U}$, and similarly $C^*(\mathbf{z}_0^{(i_{\min})} \parallel \mathbf{z}_1) = 0$ for $i \in [N] \setminus \mathcal{U}$. Hence, we set

$$\mathbf{S}_i := \begin{cases} [\bar{\mathbf{U}}_i \parallel \mathbf{R}_R] \mathbf{H}_{C^*, [\mathbf{z}_0^{(i)} \parallel \mathbf{z}_1]} & \text{if } i \in \mathcal{U}, \\ [\bar{\mathbf{U}}_i \parallel \mathbf{R}_R] \mathbf{H}_{C^*, [\mathbf{z}_0^{(i_{\min})} \parallel \mathbf{z}_1]} & \text{if } i \notin \mathcal{U}. \end{cases}$$

This yields $\mathbf{B}_{C^*}^{(i)} = \mathbf{A}_s \mathbf{S}_i$ for all $i \in [N]$. Furthermore, for $\mathbf{m}^* \in \{0, 1\}^L$, we have

$$\mathbf{M}^* = \mathbf{M}_0 + \sum_{k \in [L]} \mathbf{m}^*[k] \mathbf{M}_k = \mathbf{A}_s \left(\mathbf{U}_0 + \sum_{k \in [L]} \mathbf{m}^*[k] \mathbf{U}_k \right).$$

Assuming that $\mathcal{A}$ produces a valid forgery $\mathbf{u}$, we have $[\mathbf{A}_s \parallel \mathbf{B}_{C^*}^{(1)} \parallel \dots \parallel \mathbf{B}_{C^*}^{(N)} \parallel \mathbf{M}^*] \cdot \mathbf{u}^\top = \mathbf{0}^\top$ and $\|\mathbf{u}\| \leq \tau_2 \sqrt{m + (N+1)m'}$. This implies that we get

$$\begin{aligned} & [\mathbf{A}_s \parallel \mathbf{B}_{C^*}^{(1)} \parallel \dots \parallel \mathbf{B}_{C^*}^{(N)} \parallel \mathbf{M}^*] \cdot \mathbf{u}^\top \\ &= \mathbf{A}_s \cdot \underbrace{\left[\mathbf{I}_m \parallel \mathbf{S}_1 \parallel \dots \parallel \mathbf{S}_N \parallel \left(\mathbf{U}_0 + \sum_{k \in [L]} \mathbf{m}^*[k] \mathbf{U}_k \right) \right]}_{\mathbf{d}^\top} \cdot \mathbf{u}^\top \\ &= \mathbf{0}^\top, \end{aligned}$$

where $\|\mathbf{d}\| \leq (m + (N+1)m') \cdot \|\mathbf{S}_i\| \cdot \|\mathbf{u}\| \leq (m + (N+1)m')^{3/2} \tau_2 m^{O(d)} s \leq \beta_{\text{SIS}}$. Therefore, we conclude that $\mathcal{S}$ successfully solves the $\text{BASIS}_{n,m,q,\beta_{\text{SIS}},\tau_1,\ell_1}$ problem with probability at least $\epsilon - \text{negl}(\lambda)$. This completes the proof. $\square$

C.3 Proof of Theorem 4

Proof. We first focus on the anonymity of the signing key attribute $[\mathbf{z}_0 \parallel \mathbf{z}_1]$.

Anonymity of signing attributes. Our proof follows the framework for establishing perfect privacy in ABS [38]. Specifically, due to lattice trapdoors (Lemma 3), the distribution of any signature $\mathbf{u}$ generated for a message μ and policy C is

statistically close to a fixed Gaussian distribution $\mathcal{D}_{\mathbb{Z}, \tau_2}^{m+(N+1)m'}$. Crucially, $\mathbf{u}$ depends solely on the master public key mpk , the message μ , and the policy C , and thus does not leak information about $[\mathbf{z}_0 || \mathbf{z}_1]$. Therefore, in the context of the challenge ciphertext ct^* (Definition 13), even if decryption is successful, the distributions of the resulting signature $\mathbf{u}$ (whether using signing key ssk_1 or ssk_2) remain statistically indistinguishable. This ensures the challenge bit is hidden.

Next, we address the anonymity of the “ciphertext attributes”, while the anonymity of the signing attribute is not of concern in the following hybrid.

Anonymity of ciphertext attribute. We focus on Case 1 of Definition 13, where the adversary $\mathcal{A}$ does not possess a valid decryption key for the challenge ciphertext. In this case, the experiment for Anon-Sel is consistent with that for IND-Sel-CCA, except that two attribute-policy pairs $\{(\mathbf{x}^{(0)}, G^{(0)}), (\mathbf{x}^{(1)}, G^{(1)})\}$ are selected as challenges in the initial phase. Therefore, our proof invokes the specific hybrid arguments established in Theorem 2.

The hybrid sequence is defined as follows:

Hyb₀ In this hybrid, the challenge ciphertext ct^* is generated honestly, where the challenge bit is $b \in \{0, 1\}$.

Hyb_{0.1} $\approx \dots \approx$ Hyb_{0.7} We apply the same sequence of hybrid swaps as defined in the proof of Theorem 2 (from **Hyb₁** to **Hyb₇**), but we apply them to the ciphertext for $(\mathbf{x}^{(b)}, G^{(b)})$. Let **Hyb_{0.7}** denote the final hybrid of this sequence (corresponding to the state where the ciphertext components are simulated or obfuscated). The indistinguishability between **Hyb₀** and **Hyb_{0.7}** follows directly from Theorem 2.

In **Hyb_{0.7}**, the component related to the attribute/policy (i.e., the obfuscated program) is replaced by the output of the LO simulator Sim , since $\mathcal{A}$ satisfies the condition of Case 1.

Hyb_{1.7} This is the same as **Hyb_{0.7}**, except that the underlying simulator Sim is invoked with respect to $(\mathbf{x}^{(1-b)}, G^{(1-b)})$ instead of $(\mathbf{x}^{(b)}, G^{(b)})$.

Recall that the simulator Sim takes as input the circuit size rather than the attribute values. Since the size parameters for $(\mathbf{x}^{(b)}, G^{(b)})$ and $(\mathbf{x}^{(1-b)}, G^{(1-b)})$ are identical, the internal operation of Sim remains invariant. Under the condition of Case 1, it follows that from $\mathcal{A}$ ’s view, we have **Hyb_{0.7}** $\equiv$ **Hyb_{1.7}**.

Hyb_{1.6} $\approx \dots \approx$ Hyb₁ We now apply a *reverse* sequence, following the strategy of the swap sequence used in **Hyb_{0.1} $\approx \dots \approx$ Hyb_{0.7}**. We transform the simulated ciphertext of $(\mathbf{x}^{(1-b)}, G^{(1-b)})$ back to the real experiment.

Hyb₁ This is the real experiment where the challenge bit is $1 - b$.

In summary, the advantage $\text{Adv}_{H, \mathcal{A}}^{\text{Anon}}(1^\lambda)$ of $\mathcal{A}$ is negligible. This completes the proof. $\square$

C.4 Proof of Theorem 5

Proof. Our Special A²BSC construction with separable keys differs from the one in Sect. 4.2 in that it splits $\text{dsk}_{F, \mathbf{y}}$ into dsk_F and $\text{dsk}_{\mathbf{y}}$, thereby separating policy F

from attributes $\mathbf{y}$. Consequently, we use the same proof strategy as in Theorem 2, except for the following modifications to the master public key and key queries:

$\vdots$
Hyb₁ This is the same as **Hyb₀**, except that $\mathcal{C}$ samples the public parameters with an embedded trapdoor according to a hidden partitioning bit. Specifically, $\mathcal{C}$ samples a random bit $\beta \leftarrow \{0, 1\}$ (representing the branch guess), matrices $\mathbf{U}_0 \leftarrow \{0, 1\}^{m \times \kappa}$, and $\mathbf{U}', \mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3 \leftarrow \{0, 1\}^{m \times m}$. It then sets

$$\mathbf{P} := \mathbf{A}_r \mathbf{U}_0, \quad \mathbf{P}_0 := \begin{cases} \mathbf{A}_r \mathbf{U}', & \text{if } \beta = 0 \\ \mathbf{G} - \mathbf{A}_r \mathbf{U}', & \text{if } \beta = 1 \end{cases}$$

$$\mathbf{A}_1 := \mathbf{A}_r \mathbf{U}_1 - \mathbf{C}_{\mathbf{x}^*}, \quad \mathbf{A}_2 := \mathbf{A}_r \mathbf{U}_2 - \mathbf{C}_{G^*}, \quad \mathbf{A}_3 := \mathbf{A}_r \mathbf{U}_3 - \mathbf{C}_{\text{ovk}^*}.$$

(Note that implicitly $\mathbf{P}_1 = \mathbf{G} - \mathbf{P}_0$, so the trapdoor $\mathbf{U}'$ is embedded in $\mathbf{P}_\beta$). By the Leftover Hash Lemma (Lemma 1), the distribution of these matrices is statistically close to uniform, thus **Hyb₀** $\approx_s$ **Hyb₁** regardless of the value of β .

$\vdots$
Hyb₃ This is the same as **Hyb₂**, except that $\mathcal{C}$ answers the key extraction queries differently depending on the random choice of the partitioning bit $\beta \in \{0, 1\}$ (sampled in **Hyb₁**).

Recall that in **Hyb₁**, if $\beta = 0$, the trapdoor $\mathbf{U}'$ is embedded in $\mathbf{P}_0$; if $\beta = 1$, it is embedded in $\mathbf{P}_1$. $\mathcal{C}$ answers dExtract_i queries as follows:

- a) **Case:** $F(\mathbf{x}^*) = 1 \wedge G^*(\mathbf{y}) = 1$. (Simulatable regardless of β)
 $\text{dExtract}_0(F)$. compute $\mathbf{H}_{F, \mathbf{x}^*} := \text{EvalFX}(\mathbf{B}_1, F, \mathbf{x}^*)$ and sample $\mathbf{D}_0 \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F], (\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)}) \mathbf{H}_{F, \mathbf{x}^*}, \mathbf{P}_0, \tau_1)$ instead of $\mathbf{D}_0 \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_F], \mathbf{T}_{\mathbf{A}_r}, \mathbf{P}_0, \tau_1)$. This is due to the fact that

$$[\mathbf{A}_r \parallel \underbrace{(\mathbf{A}_1 + \mathbf{C}_{\mathbf{x}^*})}_{\mathbf{A}_r \mathbf{U}_1}] \cdot \begin{pmatrix} -\mathbf{Z}_{\mathbf{x}^*} \\ -\mathbf{P}_\ell^{(1)} \end{pmatrix} \cdot \mathbf{H}_{F, \mathbf{x}^*} = \mathbf{A}_r (-\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)}) \cdot \mathbf{H}_{F, \mathbf{x}^*} = \mathbf{B}_F - F(\mathbf{x}^*) \mathbf{G}.$$

Thus, we have $[\mathbf{A}_r \parallel \mathbf{B}_F] \cdot (\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)}) \mathbf{H}_{F, \mathbf{x}^*} = F(\mathbf{x}^*) \mathbf{G}$ as $F(\mathbf{x}^*) = 1$, where $\|(\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)}) \mathbf{H}_{F, \mathbf{x}^*}\| = \tau_0 \cdot m^{O(d)} s \cdot \text{poly}(m, \lambda)$.

$\text{dExtract}_1(\mathbf{y})$. compute $\mathbf{Z}_{G^*, \mathbf{y}} := \text{Open}^c(\text{pp}_{\text{com}}, G^*, \mathbf{y})$ and sample $\mathbf{D}_1 \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_y], (\mathbf{Z}_{G^*, \mathbf{y}} + \mathbf{U}_2 \mathbf{P}_{y,d}^{(2)}) \mathbf{G} - \mathbf{P}_0, \tau_1)$ instead of $\mathbf{D}_1 \leftarrow \text{SamplePre}([\mathbf{A}_r \parallel \mathbf{B}_y], \mathbf{T}_{\mathbf{A}_r}, \mathbf{G} - \mathbf{P}_0, \tau_1)$. This is due to the fact that

$$[\mathbf{A}_r \parallel \underbrace{(\mathbf{A}_2 + \mathbf{C}_{G^*})}_{\mathbf{A}_r \mathbf{U}_2}] \cdot \begin{pmatrix} -\mathbf{Z}_{G^*, \mathbf{y}} \\ -\mathbf{P}_{y,d}^{(2)} \end{pmatrix} = \mathbf{A}_r (-\mathbf{Z}_{G^*, \mathbf{y}} + \mathbf{U}_2 \mathbf{P}_{y,d}^{(2)}) \mathbf{G} = \mathbf{B}_y - G^*(\mathbf{y}) \mathbf{G}.$$

Thus, we have $[\mathbf{A}_r \parallel \mathbf{B}_y] \cdot (\mathbf{Z}_{G^*, \mathbf{y}} + \mathbf{U}_2 \mathbf{P}_{y,d}^{(2)}) \mathbf{G} = G^*(\mathbf{y}) \mathbf{G}$ as $G^*(\mathbf{y}) = 1$, where $\|\mathbf{Z}_{G^*, \mathbf{y}} + \mathbf{U}_2 \mathbf{P}_{y,d}^{(2)}\| = \ell(\tau_0 \cdot m)^{O(d)} \cdot \text{poly}(m, \lambda)$.

- b) **Case:** $F(\mathbf{x}^*) = 0 \wedge G^*(\mathbf{y}) = 1$. (Requires trapdoor for $\mathbf{P}_0$)
 If $\beta = 1$: $\mathcal{C}$ outputs $\perp$ and **aborts** (Simulation Failure).
 If $\beta = 0$: $\mathcal{C}$ can simulate because it knows the trapdoor $\mathbf{U}'$ for $\mathbf{P}_0 = \mathbf{A}_r \mathbf{U}'$.
 $\underline{\text{dExtract}_0(F)}$. Let $\mathbf{R}_{\text{eval}} := (\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}_1 \mathbf{P}_\ell^{(1)}) \mathbf{H}_{F, \mathbf{x}^*}$. Since $F(\mathbf{x}^*) = 0$, we have $[\mathbf{A}_r \parallel \mathbf{B}_F] \cdot \begin{pmatrix} \mathbf{R}_{\text{eval}} \\ \mathbf{I}_m \end{pmatrix} = \mathbf{0}$. Let $\mathbf{T}_P := \begin{pmatrix} \mathbf{U}' \\ \mathbf{0} \end{pmatrix}$ be the particular solution for $\mathbf{P}_0$ satisfying $[\mathbf{A}_r \parallel \mathbf{B}_F] \cdot \mathbf{T}_P = \mathbf{P}_0$. Sample

$$\mathbf{V}_P \leftarrow \text{SampleD} \left(\begin{pmatrix} \mathbf{R}_{\text{eval}} \\ \mathbf{I}_m \end{pmatrix}, \tau_1, \mathbf{T}_P \right)$$

using the **SampleD** algorithm of [30]. Return $\mathbf{D}_0 := \mathbf{T}_P - \mathbf{V}_P$.
 $\underline{\text{dExtract}_1(\mathbf{y})}$. This is the same as Case **a**) (since $G^*(\mathbf{y}) = 1$).

- c) **Case:** $F(\mathbf{x}^*) = 1 \wedge G^*(\mathbf{y}) = 0$. (Requires trapdoor for $\mathbf{P}_1$)
 If $\beta = 0$: $\mathcal{C}$ outputs $\perp$ and **aborts** (Simulation Failure).
 If $\beta = 1$: $\mathcal{C}$ can simulate because it knows the trapdoor $\mathbf{U}'$ for $\mathbf{P}_1 = \mathbf{A}_r \mathbf{U}'$. The simulation is symmetric to Case **b**), using $\mathbf{U}'$ to generate $\text{dExtract}_1(\mathbf{y})$ and running $\text{dExtract}_0(F)$ as in Case **a**).

Analysis of Simulation Failure: Let **Fail** be the event that $\mathcal{C}$ aborts. The simulation fails if the adversary queries a key that requires a trapdoor for $\mathbf{P}_{1-\beta}$, while $\mathcal{C}$ only knows the trapdoor for $\mathbf{P}_\beta$.

We categorize the valid adversary $\mathcal{A}$'s queries into two types. Note that a valid adversary cannot make queries satisfying both Case **b**) and Case **c**) in the same experiment, as this would contradict the challenge phase restrictions.

Type₀: $\mathcal{A}$ makes at least one query of Case **b**) ($F(\mathbf{x}^*) = 0$), but none of Case **c**).
Type₁: $\mathcal{A}$ makes at least one query of Case **c**) ($G^*(\mathbf{y}) = 0$), but none of Case **b**).

Let p be the probability that $\mathcal{A}$ chooses a strategy of **Type₀**. Since the view of $\mathcal{A}$ is statistically independent of β , we have

$$\begin{aligned} \Pr[\neg \text{Fail}] &= \Pr[\beta = 0 \wedge \mathcal{A} \in \text{Type}_0] + \Pr[\beta = 1 \wedge \mathcal{A} \in \text{Type}_1] \\ &= \Pr[\beta = 0] \cdot p + \Pr[\beta = 1] \cdot (1 - p) \\ &= \frac{1}{2} \cdot p + \frac{1}{2} \cdot (1 - p) = \frac{1}{2}. \end{aligned}$$

Therefore, the simulation succeeds with probability exactly $1/2$, introducing a constant loss of $1/2$ in the reduction advantage. This completes the proof. $\square$

C.5 Proof of Theorem 8

Proof. We prove the theorem by defining the following hybrid sequence. The differences between adjacent hybrids are highlighted in **Green**.

Hyb₀ This is a real game, where the challenge bit is $b \in \{0, 1\}$.

- Hyb₁** This is the same as **Hyb₀**, except that the challenger computes $\text{ct}_1^* \leftarrow \text{A}^2\text{BSC}.\text{Signcrypt}(\text{mpk}_0, \text{ssk}_{\mathbf{x}_b \parallel \widehat{\text{st}}}, C[\sigma^*], (\mathbf{x}_b, G_b), \mu'_{1-b})$ where μ'_{1-b} consists of $(\mu_{1-b}, \text{ct}_{\text{FE}}^*)$.
- Hyb₂** This is the same as **Hyb₁**, except that the challenger computes $\text{ct}_1^* \leftarrow \text{A}^2\text{BSC}.\text{Signcrypt}(\text{mpk}_0, \text{ssk}_{\widehat{\mathbf{x}} \parallel \widehat{\text{st}}}, C[\sigma^*], (\mathbf{x}_{1-b}, G_{1-b}), \mu'_{1-b})$.
- Hyb₃** This is the same as **Hyb₂**, except that the challenger computes $\text{ct}_{\text{FE}}^* \leftarrow \text{FE}.\text{Enc}(\text{mpk}_1, (1, \widehat{\text{st}}))$, where $\widehat{\text{st}} := \{\pi_{F_i}\}_{F_i \in \mathcal{F}}$. Here, $\pi_{F_i} \leftarrow \text{FC}.\text{Eval}(\text{crs}, \text{st}, F_i)$.
- Hyb₄** This is the same as **Hyb₃**, except the challenger samples $(\text{crs}, \sigma^*, \text{st}) \leftarrow \mathcal{S}_0(1^\lambda)$ instead of $(\sigma^*, \text{st}) \leftarrow \text{FC}.\text{Commit}(\text{crs}, \mathbf{x}_b)$.
- Hyb₅** This is the same as **Hyb₄**, except that the challenger samples $(\sigma^*, \text{st}) \leftarrow \text{FC}.\text{Commit}(\text{crs}, \mathbf{x}_{1-b})$.
- Hyb₆** This is the same as **Hyb₅**, except that the challenger samples $\text{ct}_{\text{FE}}^* \leftarrow \text{FE}.\text{Enc}(\text{mpk}_1, (0, \text{st}))$.
- Hyb₇** This is the same as **Hyb₆**, except that the challenger computes $\text{ct}_1^* \leftarrow \text{A}^2\text{BSC}.\text{Signcrypt}(\text{mpk}_0, \text{ssk}_{\mathbf{x}_{1-b} \parallel \widehat{\text{st}}}, C[\sigma^*], (\mathbf{x}_{1-b}, G_{1-b}), \mu'_{1-b})$.

We observe that **Hyb₇** corresponds to the real game, where the challenge bit is $1 - b$. Therefore, it suffices to show that **Hyb₁** and **Hyb₇** are computationally indistinguishable. To this end, we establish through a sequence of Lemmas 7-13 that the view of any PPT adversary in adjacent hybrids is indistinguishable. $\square$

Lemma 7. *If Special A²BSC satisfies IND-Sel-CCA, then for any PPT adversary $\mathcal{A}$, we have $\text{Hyb}_0 \approx_c \text{Hyb}_1$.*

Proof. We construct a reduction algorithm $\mathcal{B}$. Suppose there exists an adversary $\mathcal{A}$ that can distinguish **Hyb₀** from **Hyb₁** with non-negligible probability. Then, $\mathcal{B}$ can break the IND-Sel-CCA security of Special A²BSC. $\mathcal{B}$ is defined as follows:

1. Receive a challenge pairs $(\mathbf{x}_0, G_0), (\mathbf{x}_1, G_1)$ specified by $\mathcal{A}$.
2. Select challenge bit $b \leftarrow \{0, 1\}$ and send $(\mathbf{x}_b, G_b)$ to the Special A²BSC challenger.
3. After receiving mpk_0 from the challenger, run $(\text{mpk}_1, \text{msk}_1) \leftarrow \text{FE}.\text{Setup}(1^\lambda)$, $\text{crs} \leftarrow \text{FC}.\text{Setup}(1^\lambda)$.
4. Send $\text{mpk} := (\text{crs}, \text{mpk}_0, \text{mpk}_1)$ to $\mathcal{A}$.
5. Answer the oracle query from $\mathcal{A}$ as follows:
 - For SKGen, when the input is $\mathbf{x} \in \{0, 1\}^\ell$, it is forwarded to the challenger. $\mathcal{B}$ receives $\text{ssk}_{\mathbf{x}}$ from the challenger and returns $\text{ek}_{\mathbf{x}} := \text{ssk}_{\mathbf{x}}$.
 - For RKGen, when the input is $(F \in \mathcal{F}, \mathbf{y} \in \{0, 1\}^\ell)$, it is forwarded to the challenger. Receive $\text{dsk}_{F, \mathbf{y}}$ from the challenger, and compute $\text{sk}_{\text{Eval}_F} \leftarrow \text{FE}.\text{KeyGen}(\text{mpk}_1, \text{msk}_1, F_{\text{Eval}})$. Return $\text{dk}_{F, \mathbf{y}} := (\text{dsk}_{F, \mathbf{y}}, \text{sk}_{\text{Eval}_F})$.
 - For Dec, when the input is $(\text{ct}, (F, \mathbf{y}))$, it is forwarded to the challenger. Return the output from the challenger.

6. Upon receiving the challenge (μ_0, μ_1) chosen by $\mathcal{A}$, set $t_0 := (\mu_b, \text{ct}_{\text{FE}}^*)$, $t_1 := (\mu_{1-b}, \text{ct}_{\text{FE}}^*)$ and send $((t_0, t_1), C[\sigma^*])$ to the challenger. Here, the hardwired constants σ^* and ciphertext ct_{FE}^* are computed in the same manner as in **Hyb**₀.
7. Upon receiving ct_1^* from the challenger, send $\text{ct}^* := (\text{ct}_1^*, \sigma^*)$ to $\mathcal{A}$.
8. Continue responding to subsequent queries from $\mathcal{A}$ as in Step 5.
9. Return the output of $\mathcal{A}$.

Let α be the challenge bit sampled by the Special A²BSC challenger. $\mathcal{B}$ perfectly simulates the view of $\mathcal{A}$. Moreover, $\mathcal{A}$ is a valid adversary satisfying the following conditions: (mismatch) Eq. (2); (match) Eq. (3). We first consider the matching condition, where **Hyb**₀ and **Hyb**₁ are trivially indistinguishable due to $\mu_b = \mu_{1-b}$. We then turn to the mismatch condition:

- if $\mathcal{A}$ guesses that $\mathcal{B}$ simulates **Hyb**₀, then the challenge bit $\alpha = 0$ from the Special A²BSC challenger (i.e., Signcrypt is performed on t_0);
- if $\mathcal{A}$ guesses that $\mathcal{B}$ simulates **Hyb**₁, then the challenge bit $\alpha = 1$ from the Special A²BSC challenger (i.e., Signcrypt is performed on t_1).

Thus, if $\mathcal{A}$ can distinguish **Hyb**₀ and **Hyb**₁ with non-negligible probability, $\mathcal{B}$ can also break the IND-Sel-CCA security with non-negligible probability. $\square$

Lemma 8. *If Special A²BSC satisfies t -sided Anon-Sel, then for any PPT adversary $\mathcal{A}$, we have **Hyb**₁ $\approx_c$ **Hyb**₂.*

Proof. We construct a reduction algorithm $\mathcal{B}$. Suppose there exists an adversary $\mathcal{A}$ that can distinguish **Hyb**₁ from **Hyb**₂ with non-negligible probability. Then, $\mathcal{B}$ can break the t -sided Anon-Sel security of Special A²BSC. $\mathcal{B}$ is defined as follows:

1. Upon receiving the challenge pairs $(\mathbf{x}_0, G_0), (\mathbf{x}_1, G_1)$ specified by $\mathcal{A}$, forward them to the Special A²BSC challenger.
2. After receiving mpk_0 from the challenger, run $(\text{mpk}_1, \text{msk}_1) \leftarrow \text{FE.Setup}(1^\lambda)$ and $\text{crs} \leftarrow \text{FC.Setup}(1^\lambda)$.
3. Send $\text{mpk} := (\text{crs}, \text{mpk}_0, \text{mpk}_1)$ to $\mathcal{A}$.
4. Answer the oracle query from $\mathcal{A}$ as follows:
 - For SKGen, when the input is $\mathbf{x} \in \{0, 1\}^\ell$, it is forwarded to the challenger. $\mathcal{B}$ receives $\text{ssk}_{\mathbf{x}}$ from the challenger and returns $\text{ek}_{\mathbf{x}} := \text{ssk}_{\mathbf{x}}$.
 - For RKGen, when the input is $(F \in \mathcal{F}, \mathbf{y} \in \{0, 1\}^\ell)$, it is forwarded to the challenger. Receive $\text{dsk}_{F, \mathbf{y}}$ from the challenger, and compute $\text{sk}_{\text{Eval}_F} \leftarrow \text{FE.KeyGen}(\text{mpk}_1, \text{msk}_1, F_{\text{Eval}})$. Return $\text{dk}_{F, \mathbf{y}} := (\text{dsk}_{F, \mathbf{y}}, \text{sk}_{\text{Eval}_F})$.
 - For Dec, when the input is $(\text{ct}, (F, \mathbf{y}))$, it is forwarded to the challenger. Return the output from the challenger.
5. Upon receiving the challenge (μ_0, μ_1) chosen by $\mathcal{A}$, set

$$\mathbf{z}^{(0)} := [\mathbf{x}_0 \| \widetilde{\text{st}}] \text{ and } \mathbf{z}^{(1)} := [\hat{\mathbf{x}} \| \widetilde{\text{st}}].$$

Send $(\mu'_{1-b}, (\mathbf{z}^{(0)}, \mathbf{z}^{(1)}), C[\sigma^*])$ to the challenger. Here, σ^* is computed in the same manner as in **Hyb**₁.

6. Upon receiving ct_1^* from the challenger, send $\text{ct}^* := (\text{ct}_1^*, \sigma^*)$ to $\mathcal{A}$.
7. Continue responding to subsequent queries from $\mathcal{A}$ as in Step 4.
8. Return the output of $\mathcal{A}$.

Let α be the challenge bit sampled by the Special A²BSC challenger. $\mathcal{B}$ perfectly simulates $\mathcal{A}$'s view. We first observe that

$$C[\sigma^*](\mathbf{z}^{(0)}) = C[\sigma^*](\mathbf{z}^{(1)}) = 1.$$

By the anonymity of signing key attributes in t -sided Anon-Sel of Special A²BSC, this remains indistinguishable from $\mathcal{A}$'s perspective. Moreover, $\mathcal{A}$ is a valid adversary satisfying the following conditions: (mismatch) Eq. (2); (match) Eq. (3). We first consider the mismatching condition:

- if $\mathcal{A}$ guesses that $\mathcal{B}$ simulates **Hyb**₁, then the challenge bit $\alpha = 0$ from the Special A²BSC challenger;
- if $\mathcal{A}$ guesses that $\mathcal{B}$ simulates **Hyb**₂, then the challenge bit $\alpha = 1$ from the Special A²BSC challenger.

We then consider the matching condition (i.e., $\mathcal{A}$ can successfully decrypt ct^*), and the reduction proceeds identically to the previous case. Thus, if $\mathcal{A}$ can distinguish **Hyb**₁ and **Hyb**₂ with non-negligible probability, $\mathcal{B}$ can also break the t -sided Anon-Sel security with non-negligible probability. $\square$

Lemma 9. *If FE is secure, then for any PPT adversary $\mathcal{A}$, we have **Hyb**₂ $\approx_c$ **Hyb**₃.*

Proof. We construct a reduction algorithm $\mathcal{B}$. Suppose there exists an adversary $\mathcal{A}$ that can distinguish **Hyb**₂ from **Hyb**₃ with non-negligible probability. Then, $\mathcal{B}$ can break the FE security. $\mathcal{B}$ is defined as follows:

1. At the beginning, receive mpk_1 from the FE challenger.
2. Upon receiving $(\mathbf{x}_0, G_0), (\mathbf{x}_1, G_1)$ specified by $\mathcal{A}$, compute $(\text{mpk}_0, \text{msk}_0) \leftarrow \text{A}^2\text{BSC.Setup}(1^\lambda), \text{crs} \leftarrow \text{FC.Setup}(1^\lambda)$.
3. Send $\text{mpk} := (\text{crs}, \text{mpk}_0, \text{mpk}_1)$ to $\mathcal{A}$.
4. Answer the oracle query from $\mathcal{A}$ as follows:
 - For SKGen, upon input $\mathbf{x} \in \{0, 1\}^\ell$, return $\text{ek}_{\mathbf{x}} := \text{ssk}_{\mathbf{x}}$, where

$$\text{ssk}_{\mathbf{x}} \leftarrow \text{A}^2\text{BSC.sExtract}(\text{mpk}_0, \text{msk}_0, \mathbf{x}).$$

- For RKGen, upon input $(F \in \mathcal{F}, \mathbf{y} \in \{0, 1\}^\ell)$, compute

$$\text{dsk}_{F, \mathbf{y}} \leftarrow \text{A}^2\text{BSC.dExtract}(\text{mpk}_0, \text{msk}_0, F, \mathbf{y}).$$

Then, construct the circuit F_{Eval} (as defined in Eq. (10)) and submit it to the FE challenger. Upon receiving $\text{sk}_{\text{Eval}_F}$, return $\text{dk}_{F, \mathbf{y}} := (\text{dsk}_{F, \mathbf{y}}, \text{sk}_{\text{Eval}_F})$.

- For Dec, upon input $(\text{ct}, (F, \mathbf{y}))$, $\text{dk}_{F, \mathbf{y}}$ is obtained in a manner analogous to that of $\mathcal{O}_{\text{RKGen}}$. Return the output of $\text{Dec}(\text{mpk}, \text{dk}_{F, \mathbf{y}}, \text{ct})$.

5. Upon receiving the challenge (μ_0, μ_1) chosen by $\mathcal{A}$, compute $(\sigma^*, \text{st}) \leftarrow \text{FC.Commit}(\text{crs}, \mathbf{x}_b)$ and $\pi_{F_i} \leftarrow \text{FC.Eval}(\text{crs}, \text{st}, F_i)$ for $F_i \in \mathcal{F}$. Set two challenge messages $t_0 := (0, \text{st})$ and $t_1 := (1, \hat{\text{st}})$, where $\hat{\text{st}} := \{\pi_{F_i}\}_{F_i \in \mathcal{F}}$. Here, we assume that t_0 and t_1 are bit strings of equal length (padded if necessary). Send (t_0, t_1) to the challenger.
6. Upon receiving ct_{FE}^* from the challenger, send $\text{ct}^* := (\text{ct}_1^*, \sigma^*)$ to $\mathcal{A}$. Here, ct_1^*, σ^* are computed in the same manner as in **Hyb**₂.
7. Continue responding to subsequent queries from $\mathcal{A}$ as in Step 4.
8. Return the output of $\mathcal{A}$ as the guess for the FE challenge bit.

Let α be the challenge bit sampled by the FE challenger.

- If $\alpha = 0$, ct_{FE}^* is an encryption of t_0 . In this case, $\mathcal{B}$ perfectly simulates **Hyb**₂.
- If $\alpha = 1$, ct_{FE}^* is an encryption of t_1 . In this case, $\mathcal{B}$ perfectly simulates **Hyb**₃.

Crucially, the reduction is valid (admissible) because for any policy F queried by $\mathcal{A}$ (embedded in F_{Eval}), the circuit outputs are identical for both challenge messages:

$$F_{\text{Eval}}[\text{crs}, F](t_0) = F_{\text{Eval}}[\text{crs}, F](t_1).$$

Therefore, if $\mathcal{A}$ can distinguish **Hyb**₂ and **Hyb**₃ with non-negligible probability, then $\mathcal{B}$ can break the FE security with non-negligible probability as well. $\square$

Lemma 10. *If FC provides statistical private openings, then for any PPT adversary $\mathcal{A}$, we have **Hyb**₃ $\approx_s$ **Hyb**₄.*

Proof. We construct a reduction algorithm $\mathcal{B}$. Suppose there exists an adversary $\mathcal{A}$ that can distinguish **Hyb**₃ from **Hyb**₄ with non-negligible probability. Then, $\mathcal{B}$ can break the private openings of FC. $\mathcal{B}$ is defined as follows:

1. At the beginning, receive crs from the FC challenger.
2. Upon receiving $(\mathbf{x}_0, G_0), (\mathbf{x}_1, G_1)$ specified by $\mathcal{A}$, compute $(\text{mpk}_0, \text{msk}_0) \leftarrow \text{A}^2\text{BSC.Setup}(1^\lambda)$ and $(\text{mpk}_1, \text{msk}_1) \leftarrow \text{FE.Setup}(1^\lambda)$. Send $\text{mpk} := (\text{crs}, \text{mpk}_0, \text{mpk}_1)$ to $\mathcal{A}$.
3. Answer the oracle queries from $\mathcal{A}$ as follows:
 - For SKGen, upon input $\mathbf{x} \in \{0, 1\}^\ell$, return $\text{ek}_{\mathbf{x}} := \text{ssk}_{\mathbf{x}}$ where

$$\text{ssk}_{\mathbf{x}} \leftarrow \text{A}^2\text{BSC.sExtract}(\text{mpk}_0, \text{msk}_0, \mathbf{x}).$$

- For RKGen, upon input $(F \in \mathcal{F}, \mathbf{y} \in \{0, 1\}^\ell)$, compute

$$\text{dsk}_{F, \mathbf{y}} \leftarrow \text{A}^2\text{BSC.dExtract}(\text{mpk}_0, \text{msk}_0, F, \mathbf{y}),$$

$$\text{sk}_{\text{Eval}_F} \leftarrow \text{FE.KeyGen}(\text{mpk}_1, \text{msk}_1, F_{\text{Eval}}).$$

Return $\text{dk}_{F, \mathbf{y}} := (\text{dsk}_{F, \mathbf{y}}, \text{sk}_{\text{Eval}_F})$.

- For Dec, upon input $(\text{ct}, (F, \mathbf{y}))$, $\text{dk}_{F, \mathbf{y}}$ is obtained in a manner analogous to that of $\mathcal{O}_{\text{RKGen}}$. Return the output of $\text{Dec}(\text{mpk}, \text{dk}_{F, \mathbf{y}}, \text{ct})$.
4. Upon receiving the challenge (μ_0, μ_1) chosen by $\mathcal{A}$, sample $b \leftarrow \{0, 1\}$. Send $\mathbf{x}_b$ to the challenger.

5. Upon receiving σ^* from the challenger, send all $F_i \in \mathcal{F}$ to the challenger.
6. Upon receiving π_{F_i} for $F_i \in \mathcal{F}$ from the challenger, compute ct_{FE}^* as in **Hyb**₃.
Then, send $\text{ct}^* := (\text{ct}_1^*, \sigma^*)$ to $\mathcal{A}$. Here, ct_1^* is computed as in **Hyb**₃.
7. Continue responding to subsequent queries from $\mathcal{A}$ as in Step 3.
8. Return the output of $\mathcal{A}$.

Let α be the challenge bit sampled by the FC challenger. $\mathcal{B}$ perfectly simulates $\mathcal{A}$'s view:

- If $\mathcal{A}$ guesses that $\mathcal{B}$ simulates **Hyb**₃, then the challenge bit $\alpha = 0$ from the FC challenger (i.e., σ^* is the commitment to $\mathbf{x}_b$ via the honest execution of FC.Commit).
- If $\mathcal{A}$ guesses that $\mathcal{B}$ simulates **Hyb**₄, then the challenge bit $\alpha = 1$ from the FC challenger (i.e., σ^* is generated by simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$).

Thus, if $\mathcal{A}$ can distinguish **Hyb**₃ and **Hyb**₄ with non-negligible probability, then $\mathcal{B}$ can break the private openings of FC with non-negligible probability. $\square$

Lemma 11. *If FC provides statistical private openings, then for any PPT adversary $\mathcal{A}$, we have **Hyb**₄ $\approx_s$ **Hyb**₅.*

Proof. This follows by adapting the strategy of Lemma 10 and reducing to the private openings of FC, as this is the reverse direction from **Hyb**₃ to **Hyb**₄. $\square$

Lemma 12. *If FE is secure, then for any PPT adversary $\mathcal{A}$, we have **Hyb**₅ $\approx_c$ **Hyb**₆.*

Proof. This follows by adapting the strategy of Lemma 9 and reducing to the FE security, as this is the reverse direction from **Hyb**₂ to **Hyb**₃. $\square$

Lemma 13. *If Special A²BSC satisfies t -sided Anon-Sel, then for any PPT adversary $\mathcal{A}$, we have **Hyb**₆ $\approx_c$ **Hyb**₇.*

Proof. This follows by adapting the strategy of Lemma 8 and reducing to the t -sided Anon-Sel security of Special A²BSC, since

$$C[\sigma^*](\hat{\mathbf{x}} \parallel \tilde{\mathbf{st}}) = C[\sigma^*](\mathbf{x}_{1-b} \parallel \tilde{\mathbf{st}}) = 1.$$

$\square$

C.6 Proof of Theorem 9

Proof. We construct a PPT algorithm $\mathcal{B}$ as follows. Suppose there exists an adversary $\mathcal{A}$ that breaks the authenticity of AME with non-negligible probability. Then, $\mathcal{B}$ can break the no-signing EUF-CoSel security of Special A²BSC with non-negligible advantage. $\mathcal{B}$ is defined as follows:

1. At the beginning of the game, $\mathcal{B}$ selects a set of signing key attributes $\{\mathbf{x}^{(i)}\}_{i \in [N]}$ and $\mathcal{U} \subseteq [N]$, and sends them to the A²BSC challenger.

2. Receive mpk_0 from the challenger, and compute $(\text{mpk}_1, \text{msk}_1) \leftarrow \text{FE.Setup}(1^\lambda)$, $\text{crs} \leftarrow \text{FC.Setup}(1^\lambda)$. Send $\text{mpk} := (\text{crs}, \text{mpk}_0, \text{mpk}_1)$ to $\mathcal{A}$.
3. Answer the oracle query from $\mathcal{A}$ as follows:
 - For SKGen , when the input is $\mathbf{x} \in \{\mathbf{x}^{(i)}\}_{i \in \mathcal{U}}$, it is forwarded to the challenger. $\mathcal{B}$ receives $\text{ssk}_{\mathbf{x}}$ from the challenger and returns $\text{ek}_{\mathbf{x}} := \text{ssk}_{\mathbf{x}}$.
 - For RKGen , when the input is $(F \in \mathcal{F}, \mathbf{y} \in \{0, 1\}^\ell)$, it is forwarded to the challenger. Receive $\text{dsk}_{F, \mathbf{y}}$ from the challenger, and compute $\text{sk}_{\text{Eval}_F} \leftarrow \text{FE.KeyGen}(\text{mpk}_1, \text{msk}_1, F_{\text{Eval}})$. Return $\text{dk}_{F, \mathbf{y}} := (\text{dsk}_{F, \mathbf{y}}, \text{sk}_{\text{Eval}_F})$.
4. Upon receiving the forgery $(\text{ct}^*, F^*, \mathbf{y}^*)$ from $\mathcal{A}$, forward $(F^*, \mathbf{y}^*)$ directly to the challenger. After receiving the corresponding output $\text{dsk}_{F^*, \mathbf{y}^*}$, parse ct^* as $(\text{ct}_1^*, \sigma^*)$. If $\mu^* \leftarrow \text{Unsigncrypt}(\text{mpk}_0, \text{ct}_1^*, \text{dsk}_{F^*, \mathbf{y}^*})$, return $(C[\sigma^*], (F^*, \mathbf{y}^*), \mu^*, \text{ct}_1^*)$ as the forgery to the A^2BSC challenger; otherwise, abort.

We observe that the simulation of oracle queries is perfect. Specifically, given $(\text{mpk}_0, \text{mpk}_1, \text{crs})$, for $\mathbf{x} \in \mathcal{U}$ and $(F \in \mathcal{F}, \mathbf{y} \in \{0, 1\}^\ell)$, all oracle queries made by $\mathcal{A}$ can be perfectly simulated by $\mathcal{B}$. Moreover, due to the binding property of FC , the circuit $C[\sigma]$ is strictly bound to the specific attribute $\mathbf{x}$. Thus, if $\mathcal{A}$ has not queried $\text{ek}_{\mathbf{x}}$ such that $F(\mathbf{x}) = 1$, any valid AME ciphertext output by $\mathcal{A}$ allows the reduction $\mathcal{B}$ to extract a valid forgery against the underlying A^2BSC scheme. $\square$