Laconic Pre-Constrained Encryption

Shweta Agrawal[*] Simran Kumari[†] Ryo Nishimaki[‡]

May 29, 2025

Abstract

The recent work of Ananth et al. (ITCS 2022) initiated the study of pre-constrained encryption (PCE) which achieves meaningful security even against the system authority, *without assuming any trusted setup*. They provided constructions for special cases such as pre-constrained Attribute Based Encryption (PC-ABE) for point functions and pre-constrained Identity Based Encryption (PC-IBE) for general functions from the Learning with Errors (LWE) assumption. For the most general notion of PCE for circuits, they provided a construction from indistinguishability obfuscation (iO) and moreover, proved a lower bound showing that the reliance on iO was inherent. In all their constructions, the size of the public key scales linearly with the size of the constraint input to the setup algorithm.

In this work we initiate the study of *laconic* pre-constrained encryption, where the public key is sublinear in the size as well as number of constraints input to the setup algorithm. We make the following contributions:

1. We construct laconic pre-constrained ABE for point functions and laconic pre-constrained IBE for general functions from LWE which achieves succinct public keys, thus improving upon the work of Ananth et al.
2. For general constraints, we sidestep the lower bound by Ananth et al. by defining a weaker *static* notion of pre-constrained encryption (sPCE), which nevertheless suffices for all known applications. We show that laconic sPCE is impossible to achieve in the strongest malicious model of security against authority and provide the first construction of semi-malicious laconic sPCE for general constraints from LWE in the random oracle model.
3. For general constraints, to achieve malicious security without iO, we provide constructions of *non-laconic* sPCE from a variety of assumptions including DDH, LWE, QR and DCR. Our LWE based construction satisfies unconditional security against malicious authorities.
4. As an application of our sPCE, we provide the first construction of pre-constrained group signatures supporting general constraints, achieving unconditional anonymity and unlinkability against malicious authorities from the LWE assumption. The only other construction by Bartusek et al. supports the restricted set/database membership constraint, and achieves computational security from the DDH assumption.

Along the way, we define and construct the notion of pre-constrained Input Obfuscation which may be of independent interest.

Keywords: pre-constrained encryption, security against authority, privacy versus accountability

[*] IIT Madras, India, shweta@cse.iitm.ac.in

[†] IIT Madras, India, sim78608@gmail.com (Part of this work was done while visiting NTT Social Informatics Laboratories as an internship.)

[‡] NTT Social Informatics Laboratories, Tokyo, Japan, NTT Research Center for Theoretical Quantum Information, Atsugi, Japan, ryo.nishimaki@ntt.com

Contents

1 Introduction

An important balance that the field of cryptography seeks to enable is that between privacy and accountability. Consider the classic example of encryption – the key generation procedure for public key encryption (PKE) is typically run by service providers, who are often not trusted by end users but can nevertheless access encrypted user data. Anger against mass surveillance and "hidden" trapdoors led to the creation of end-to-end encryption services (E2EE), which guarantees that even the service provider itself cannot access the information that it is storing or transmitting on behalf of the user. However, the right to privacy of users must be contrasted with the helplessness of law enforcement agencies, such as the police, in being unable to access conversations that are threatening to society, for example those that incite violence or distribute illegal information. Thus, a fundamental question, which has received much attention in recent years, is – *is there a way to guarantee privacy for honest users while enforcing accountability for propagating illegal information?*

1.1 Prior Work

There has been a large body of work addressing this question in recent years, for different primitives and achieving different notions of security. In this work, we continue this study for the fundamental primitive of encryption. We discuss known solutions for PKE below and relegate discussion of other primitives to Section 1.3.

Pre-Constrained Encryption. The work of Ananth et al. [AJJM22] proposed the notion of *pre-constrained encryption* (PCE) as a new model for advanced encryption schemes where *even the setup authority* does not have full decryption power, but instead can only decrypt some "authorized" values on private data. Importantly, this model does not assume any trusted setup or CRS generation. In more detail, Ananth et al. [AJJM22] define pre-constrained encryption with respect to a constraint family $\mathcal{C}$ and a function family $\mathcal{F}$ where the constraint family $\mathcal{C}$ models the kinds of decryption capabilities permitted to the authority. Now, setup is run with respect to some constraint $C \in \mathcal{C}$ and this produces a public key together with a master secret key which is "constrained" to C, the key generation procedure takes this master key and a function $f \in \mathcal{F}$ and outputs a function key SK_f if and only if $C(f) = 1$. Encryption supports computing a ciphertext CT_x for any input x and decryption of CT_x with SK_f enables recovery of $f(x)$. In particular, note that even the authority holding the master secret can only derive function keys and hence perform computations for authorized functions f, as captured by $C(f) = 1$, and nothing else. This is the fundamental difference from the related primitive of functional encryption (FE) where the authority possesses a master secret key which can be used to decrypt *any* ciphertext. Indeed, the existence of such a master key has been the cause of much concern in FE schemes, and several works [BF03, Cha07, LW11, Goy07, GLSW08, BGJS16, GHMR18, GHM+19, GV20] have attempted to find solutions to mitigate this so-called "key escrow" problem.

Ananth et al. [AJJM22] (AJJM) provide several constructions for PCE. Analogously to the literature on FE, they consider special cases of PCE such as identity-based PCE and attribute-based PCE, denoted as IB-PCE and AB-PCE, respectively. These notions are similar to the celebrated notions of identity-based encryption (IBE) [BF03] and attribute-based encryption (ABE) [GPSW06], except that the functions must now additionally be authorized via a constraint specified during setup. In more detail, an AB-PCE scheme consists of four algorithms – Setup takes as input a constraint C and outputs a public key and master key, KeyGen takes as input a predicate f and outputs SK_f if $C(f) = 1$, Enc takes as input an attribute vector x and a message μ and outputs (x, CT_x) and Dec takes SK_f and (x, CT_x) and outputs μ if $f(x) = 1$. In an IB-PCE, the function f is restricted to be a vector y and we define $f(x) = 1$ iff $x = y$.

AJJM provide the first constructions of PCE – they build AB-PCE for point constraints (i.e. $C_{x^*}(f) = 1$ iff $f(x^*) = 0$) and IB-PCE for general constraints C from the Learning With Errors (LWE) assumption. Both these constructions cleverly use the "punctured" proof technique of the ABE scheme by Boneh et al. [BGG+14] to puncture the master key in the constructions.

To construct AB-PCE for general circuit constraints, they rely on the strong primitive of witness encryption (WE) together with NIZK proofs with perfect soundness. Additionally, they construct PCE for general constraints using indistinguishability obfuscation (iO) and NIZKs with perfect soundness. Moreover, they show that the usage of strong primitives like iO and WE is inherent since AB-PCE for general circuit constraints implies WE for NP while PCE for general circuit constraints implies iO for P/poly.

Self-Detecting Encryption. Another closely related primitive is self-detecting encryption defined by Alamati et al. [ABD$^+$21]. A self-detecting encryption scheme is similar to a regular public-key encryption with the key difference that it is possible to detect whether the underlying message of a given ciphertext belongs to a database of certain illegal messages. Moreover, such a check can be performed just by knowing the database values, as opposed to the system's secret key – this enables the feature that illegal contents in encrypted messages can be flagged even without knowing the secret key, without compromising the privacy of honest messages.

Formally, SDE, similar to PKE allows to generate a key pair $(\mathsf{pk}, \mathsf{sk})$. There is a hash algorithm which computes a hash value h_{DB} and state information st from a database DB. A user can then generate a ciphertext ct_m of m by using h_{DB} and pk. The secret key holder can decrypt ct_m. In addition, it has a detection algorithm that can recover m from ct_m and st (without sk) if $m \in \mathsf{DB}$. Observe that DB has no relation to $(\mathsf{pk}, \mathsf{sk})$. The security against the authority of self-detecting encryption guarantees $(\mathsf{pk}, (h_{\mathsf{DB}}, \mathsf{st}), \mathsf{Enc}(\mathsf{pk}, h_{\mathsf{DB}}, m_0)) \overset{c}{\approx} (\mathsf{pk}, (h_{\mathsf{DB}}, \mathsf{st}), \mathsf{Enc}(\mathsf{pk}, h_{\mathsf{DB}}, m_1))$ if $m_0, m_1 \notin \mathsf{DB}$.

The notion of self detecting encryption is philosophically similar to that of PCE with a database constraint – the state information st can be seen as a constrained key which only allows the authority to learn m if it belongs to an illegal set. However, the constructions of SDE use a CRS generated honestly by a Prm algorithm, hence only achieve security against a semi-honest authority[1]. Also, the database constraint, while clearly useful, is nevertheless restrictive and it is useful to study a broader class of constraints.

Set Pre-Constrained Encryption and Group Signatures. Bartusek et al. [BGJP23] defined the notion of set pre-constrained (SPC) encryption where setup is provided a database D and outputs a public and secret key. The public key can be used to encrypt a message (x, m) while the secret key can be used to decrypt a ciphertext to recover m so long as $x \in D$. Similar to SDE, this notion is similar to PCE for the database constraint. However, the precise definitions of security considered are different in the two works. In terms of constructions, the authors provided a concretely efficient scheme from Decision Diffie Hellman (DDH). Further, they defined the notion of set pre-constrained (SPC) group signatures, which enable tracing of users in messaging systems who sign predefined illegal content while providing security against malicious group managers. They also provided concretely efficient protocols for Set Pre-Constrained (SPC) group signatures from DDH, and an implementation to demonstrate practical efficiency.

1.2 Our Approach

In this work, we extend the study of pre-constrained encryption (PCE) as defined by Ananth et al. [AJJM22] (AJJM). We provide new definitions as well as new constructions, summarized next.

1.2.1 Definitions.

We provide the following new definitions in the context of PCE.

Weaker Notion for General Constraints. As discussed above, AJJM shows that in the context of general constraints, the reliance of PCE on strong notions like witness encryption and obfuscation is inherent. This is discouraging – real world applications may require support for arbitrary constraints and using such strong primitives creates barriers to deployment. As an example, consider their own motivating example of spam filtering, where the secret key should open ciphertexts containing spam – here the functionality "is-spam?" is formalized by some constraint C, which could be arbitrarily complex. If the stated definition for general constraints necessitates usage of strong assumptions, is it possible to weaken the definition in a meaningful way?

Taking a step back, we observe that the applications envisaged by AJJM can be realized using a weaker variant of PCE which removes the dynamic key delegation functionality required by their definition and where the constraint acts on the *data* rather than on the keys. In more detail, we define a weaker notion of PCE, which we call *static* PCE (denoted by sPCE) as a PKE scheme where a constraint C is embedded in the secret key created during setup, the encryptor computes a ciphertext for any message x and decryption succeeds to recover x if and only if $C(x) = 1$. More generally, we may embed a set of constraints $C_1, \ldots, C_Q$ in the public key and allow decryption to recover $\{C_i(x)\}_{i \in [Q]}$. As in the AJJM PCE, *even the authority* cannot learn unauthorized functions on the data – the difference is that these functions

[1] The authors do consider maliciously constructed ciphertexts, but not maliciously generated CRS.

are now fixed and provided to setup, without allowing dynamic choice during key generation. Similarly to AJJM, we ask that the public key does not reveal information about C – *constraint-hiding* and that the secret key holder cannot learn any leakage on x if $C(x) = 0$ – *security against authority*. Also, as in [AJJM22], we envisage the deployment of pre-constrained encryption in conjunction with an end-to-end encryption scheme so that user data is encrypted twice, once under each scheme. The former is used for accountability to the authority, while the latter is used for regular communication. Standard cryptographic tools are used to ensure that the data encrypted under both schemes is the same – please see [AJJM22] for a discussion.

Applications. Our notion of sPCE provides a simpler way to realize several applications that were studied by [AJJM22]. For instance, they motivate their notion of identity-based PCE via the example of a "no-fly list" where the public key contains a list (say S) of suspected individuals who should not be allowed to fly. The encryption and function keys are with respect to identities (possibly user public keys). The master key which encodes S can be used to derive the secret key for any identity in S via a key derivation procedure, which in turn can be used to decrypt any ciphertext associated with an identity in S. In sPCE too, the public key can encode a no-fly list as a constraint C and the ciphertext can encode the identity x. If $C(x) = 1$, the ciphertext can be decrypted. Thus, the difference is that we do not explicitly provide a key derivation procedure, but the constrained master secret key suffices for decryption as desired above. We remark that there may be other applications which require the full power of IB-PCE, and we also study this notion below. The primary reason for considering the weaker static notion is for supporting general constraints.

On account of acting directly on data, our notion of sPCE also enables new applications:

1. *Checking Data Sanitization:* Consider a constraint C which encodes some program that checks the content for illegal or undesirable attributes such as violence or racial biases. Now, the setup provides a secret key that encodes C, the encryptor computes a ciphertext for some input x and the authority can recover x if and only if $C(x) = 1$.

2. *Crime Investigation:* During a crime investigation, it is desirable for the authority to have a key encoding some constraint that checks for names of suspects (or such other material) in encrypted chat conversations, and allows them to only decrypt the matching chat messages. Here, the user's message x is encrypted in ciphertext CT_x and recovered by the key if and only if $C(x) = 1$.

Relation with reusable 2 *round* 2 *party secure computation.* The above notion of sPCE is arguably natural – indeed, it is closely connected with reusable 2 round 2 party secure computation, which has been studied extensively in the literature [BL20, AJJM20, BGMM20, BJKL21, AJJM21, BGSZ22, IKSS23]. In more detail, PCE can be seen as a special case of reusable 2PC by collapsing the setup and decrypt algorithms of PCE into the same (first) party with input C and by considering encrypt as the second party with input x. However, we believe that it is meaningful to study PCE as a separate notion for the following reasons:

1. The fundamental security property in PCE is against authority without relying on trusted setup – this renders 2PC protocols with trusted setup (such as CRS) or satisfying only semi-malicious security, ill-suited for our setting.

2. Since reusable 2PC security definitions are simulation based, 4 rounds are optimal for malicious security in the plain model [KO04]. However, PCE generalizes PKE so we cannot admit protocols which incur more than 2 rounds. To the best of our knowledge, 2 round maliciously secure reusable MPC in the plain model relies on super-polynomial-time simulation and strong assumptions such as iO [FJK23]. In contrast, our receiver-side security definition is game-based and admits constructions from standard assumptions.

Laconic Pre-Constrained Encryption. The main notion we study in this work is that of *laconic* (or *succinct*) sPCE, where the public key is sublinear in the size as well as the number of constraints embedded in it. We study laconic PCE both in the static setting for general constraints, as well as in AJJM's dynamic setting for special cases like AB-PCE and IB-PCE.

Pre-Constrained Input Obfuscation. We introduce a natural dual of sPCE which we term *pre-constrained input obfuscation*. In this primitive, the setup algorithm hides a set of inputs in the public key, the obfuscate algorithm produces an obfuscated program and the evaluation algorithm allows to compute the output of the program on the hidden inputs. To see the motivation for this notion, say that a user/prover claims they have an algorithm for a difficult problem. To check this without leaking the algorithm, the verifier chooses several large random inputs and programs them into the public key. The prover obfuscates her code using this key (note that the inputs remain hidden) and sends this to the verifier, who can test whether the correct output is produced in reasonable time.

1.2.2 Constructions.

We provide new constructions from simple, standard assumptions.

Warmup: Non Laconic, General Constraints. As a warm-up, we provide a simple construction of sPCE for *general* constraints, which relies on two-message statistically sender-private oblivious transfer (SSP-OT) as well as garbled circuits and achieves security against a *malicious* authority. Note that SSP-OT can be based on diverse assumptions such as DDH [AIR01, NP01], QR and DCR [HK12], LWE [BD18, DGI+19, ADD+23], LPN and Nisan-Wigderson style derandomization [BF22]. This construction bears similarities to constructions of reusable 2PC that have appeared in the literature [GSW23], though the details are quite different. In more detail, we obtain the following theorem.

Theorem 1.1. Assuming DDH or (QR and DCR) or LWE or (LPN and Nisan-Wigderson style derandomization), there exists a sPCE scheme, for general circuits, satisfying security against a malicious authority (Definition 3.5).

Next we provide a construction which supports general constraints and relies on malicious circuit-private fully homomorphic encryption (FHE)[OPP14], which can be instantiated from the LWE assumption. This construction satisfies *unconditional* security against a *malicious* authority, and can be conjectured post-quantum secure. In contrast, the constructions of AB-PCE and PCE for general constraints, even while relying on strong primitives like WE or iO, do not achieve unconditional security even against a semi-malicious authority.

Theorem 1.2. Assuming LWE, there exists a sPCE scheme, for general circuits, satisfying unconditional security against a malicious authority (Definition 3.6).

Laconic, General Constraints. In the laconic setting, we first show that sPCE is *impossible* to achieve against a malicious authority. Relaxing the security to semi-malicious, we provide the first construction of laconic sPCE from LWE in the random oracle model.

Theorem 1.3. There exists a laconic sPCE scheme for general constraints that satisfies security against a semi-malicious authority (as defined in Definition 3.4) in the random oracle model, under the LWE assumption, achieving a succinct master public key with $|\mathsf{mpk}| = \mathrm{poly}(\lambda)$.

Laconic AB-PCE and IB-PCE. For AB-PCE and IB-PCE, we use the definitions of AJJM supporting dynamic key generation and achieve the following.

Theorem 1.4. There exists a laconic attribute-based pre-constrained encryption (AB-PCE) scheme for point-constraints that satisfies semi-malicious security (as defined in Definition 2.2), under the LWE assumption, achieving a succinct master public key of size $|\mathsf{mpk}| = \mathrm{poly}(\lambda, d)$, where d denotes the maximum depth of the function class supported by the scheme.

In contrast, AJJM constructed a AB-PCE scheme for point-constraints satisfying semi-malicious security assuming LWE and achieving $|\mathsf{mpk}| = \mathrm{poly}(\lambda, d, \ell)$, where d denotes the maximum depth of the function class supported by the scheme and ℓ denotes the attribute length.

Theorem 1.5. There exists a laconic identity-based pre-constrained encryption (IB-PCE) scheme for general constraints that satisfies semi-malicious security (as defined in Definition 2.2), under the LWE assumption, achieving a succinct master public key of size $|\mathsf{mpk}| = \mathrm{poly}(\lambda)$.

The prior work by [AJJM22] constructed a IB-PCE scheme for general constraints satisfying semi-malicious security assuming LWE and achieving $|\mathsf{mpk}| = \mathrm{poly}(\lambda, \ell)$, where ℓ denotes the constraint size.

Pre-Constrained Group Signatures. Using our sPCE, we make the following advances in pre-constrained group signatures (PCGS). We provide the first construction supporting *general* constraints, achieving *unconditional* anonymity and unlinkability against *malicious* authorities. The only other construction by Bartusek et al. [BGJP23] supports the restricted set/database membership constraint, and achieves computational security. Our construction relies on the LWE assumption while theirs relies on DDH – thus, our construction has the advantage of being plausibly post-quantum secure. Additionally, our construction achieves a stronger security notion, namely *unlinkability* as compared to theirs. On the other hand, their construction enjoys concrete efficiency and comes with an implementation, whereas ours does not.

In Appendix D.2 we show that the PCGS compiler by [BGJP23] can be adapted to use our sPCE scheme and we have the following implications.

1. Using a sPCE scheme as in Theorem 1.1, we achieve a PCGS scheme for general constraints achieving anonymity and unlinkability against a malicious authority for general constraints. For constraints in NC^1, we can achieve unconditional anonymity and unlinkability against a malicious authority.

2. Using a sPCE scheme as in Theorem 1.2, we achieve a PCGS scheme for general constraints achieving unconditional anonymity and unlinkability against a malicious authority for general constraints.

3. Using a laconic sPCE scheme as in Theorem 1.3, we achieve a succinct PCGS scheme for general constraints achieving security guarantees against a semi-malicious PPT authority.

Pre-Constrained Input Obfuscation. We provide a construction of pre-constrained input obfuscation (PCIO) scheme from a sPCE scheme. This lets us instantiate PCIO from the same assumptions as those used to instantiate sPCE.

1.3 Technical Overview

We proceed to outline the technical ideas used in our constructions.

Static Pre-Constrained Encryption. We define a static pre-constrained encryption (sPCE) scheme consisting of three algorithms: $\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}$. Here, Enc and Dec are encryption and decryption algorithms of standard FE. Setup takes as input a security parameter and functions $(f_1, \ldots, f_Q)$, and outputs a public key and functional decryption keys $(\mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q})$. As discussed above, this notion simplifies the notion of PCE defined by Ananth et al. [AJJM22] and can also be seen as a natural generalization of the notion of set pre-constrained encryption by Bartusek et al. [BGJP23]. We remark that our security notions are game-based, similar to Ananth et al. [AJJM22] in contrast to the ideal-functionality-based security notions of Bartusek et al. [BGJP23]. We make this choice to separate the authentication of constraints from the schemes.

There are three security requirements for sPCE. One is the standard indistinguishability against adversaries who do not have functional decryption keys. Another is function-hiding, which ensures that public keys do not reveal information about the functions embedded during the setup phase. The third is security against authority – this can be semi-honest, semi-malicious or malicious, which are increasingly stronger. We focus on malicious authority in this section, which allows the authority to behave arbitrarily during the setup phase. The requirement imposed by this notion of security is very strong – a ciphertext of x does not provide any information beyond $(f_1(x), \ldots, f_Q(x))$ even if the malicious authority generates a possibly malformed public key $\widetilde{\mathsf{pk}}$ where $(f_1, \ldots, f_Q) \leftarrow \mathrm{Ext}(1^\lambda, \widetilde{\mathsf{pk}})$ and Ext is a possibly inefficient extractor. When security holds against a computationally unbounded authority, we say that it satisfies unconditional security. It is easy to show that security against semi-honest/semi-malicious/malicious authority and constraint-hiding imply the standard indistinguishability – we do not discuss this in the remainder of this overview.

Warmup: Construction based on OT. As discussed above, Ananth et al. [AJJM22] provide constructions by using the punctured proof technique of Boneh et al. [BGG+14]. This technique is very well suited for constructing pre-constrained encryption since it naturally lends itself to constraining the master key, providing constructions for the restricted primitives of IB-PCE for general constraints and AB-PCE for point constraints. However, this technique is highly specific to LWE-based ABE constructions and appears very hard to generalize (even to punctured proofs in pairing-based ABE constructions, for instance).

From broader assumptions, it appears very challenging to guarantee confidentiality against a malicious authority. This task seems even more difficult when considering security against an unbounded authority. To tackle this difficulty, we take advantage of our simpler definition, which does not require key delegation. In this setting, we then leverage two-message secure two-party computation between an authority with input C (constraint) and a sender with input x (plaintext). The authority obtains $x \cdot C(x)$ (and the sender obtains nothing). If $C(x) = 0$, the authority cannot obtain information about x.

To implement this idea, we make use of two-message statistically sender-private oblivious transfer (SSP-OT) [HK12, BD20]. This primitive satisfies the following two requirements: (1) computational indistinguishability between $\mathsf{ot}_1(0)$ and $\mathsf{ot}_1(1)$ where $\mathsf{ot}_1(\beta)$ is the receiver's message with choice bit β, and (2) statistical indistinguishability between the sender message generated from $(\mu_0, \mu_1, \mathsf{ot}_1)$ and one generated from $(\mu_{\beta'}, \mu_{\beta'}, \mathsf{ot}_1)$ where $\beta' \leftarrow \mathrm{Ext}(\mathsf{ot}_1)$, Ext is a possibly inefficient extractor, and ot_1 is the receiver's message. We can instantiate this primitive with many standard assumptions [HK12, BD20] such as the DDH, QR, and LWE assumptions.

Our sPCE scheme can be constructed as follows. For simplicity we consider that setup is constrained on one function f. The setup algorithm generates an SSP-OT receiver's message $\mathsf{ot}_{1,i}$ with choice bit $\beta_i := f[i]$ for all $i \in [\ell]$ where $|f| = \ell$ and f is the constraint function. The public key is $\{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$. The encryption algorithm generates a garbled circuit $\widetilde{P}$ and its labels $\{\mathsf{lb}_{i,b}\}_{i\in[\ell],b\in\{0,1\}}$ of a circuit $P[x]$ that takes as input a circuit f and outputs $f(x)$. It also generates an SSP-OT sender's message $\mathsf{ot}_{2,i}$ of $(\mathsf{lb}_{i,0}, \mathsf{lb}_{i,1})$ for all $i \in [\ell]$. A ciphertext consists of $(\widetilde{P}, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]})$. The authority can recover $\{\mathsf{lb}_{i,f[i]}\}_{i\in[\ell]}$ from $\{\mathsf{ot}_{2,i}\}_{i\in[\ell]}$ and $P[x](f) = f(x)$ from $\widetilde{P}$ and the labels. The receiver security of SSP-OT guarantees constraint-hiding. The statistical sender security (against malicious receiver) and the garbled circuit security guarantee security against malicious authority since the (inefficient) extractor can extract C from $\{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$ and we can simulate the SSP-OT sender's message and the garbled circuit using only $\{\mathsf{lb}_{i,C[i]}\}_{i\in[\ell]}$ and $f(x)$.

To extend this construction to support Q functions, the setup algorithm uses more SSP-OT instances. That is, it generates $Q \times \ell$ SSP-OT receiver's messages where Q is the number of functions and ℓ is the size of functions. The rest of the construction follows a similar outline as above. The public key size is linear in Q.

Although we use statistical sender privacy for security against malicious authority, the constructions do not achieve unconditional security since we use garbled circuits in the ciphertext. If we restrict the constraint/function class to NC^1, these constructions achieve unconditional security since an information-theoretic version of Yao's garbled circuit exists for NC^1 circuits [IK02]. Please see Section 3.2 for details.

Unconditionally secure sPCE based on FHE. To obtain unconditional security in sPCE, we need to use other tools that support all functions since information-theoretically secure garbled circuits for all circuits do not exist so far. Our next idea is using circuit private fully homomorphic encryption (FHE) to implement the two-message secure two-party computation. The setup algorithm generates a key pair $(\mathsf{fhe.pk}, \mathsf{fhe.sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ and a ciphertext $\mathsf{fhe.ct}_f$ of constraint f and outputs $(\mathsf{fhe.pk}, \mathsf{fhe.ct}_f)$ as a public key. The encryption algorithm applies the evaluation algorithm of FHE to $\mathsf{fhe.ct}$ and a circuit $P[x]$ above. The evaluated ciphertext should be statistically indistinguishable from $\mathsf{FHE.Enc}(\mathsf{fhe.pk}, f(x))$ due to the circuit privacy of FHE. The authority can recover $f(x)$ and nothing beyond that. Constraint-hiding follows from the indistinguishability of FHE.

The big issue in this idea is that a malicious authority may generate a malformed public key (and ciphertext), and the circuit privacy of FHE is not guaranteed. Hence, we use *maliciously circuit private* FHE by Ostrovsky, Paskin-Cherniavsky, and Paskin-Cherniavsky [OPP14], which guarantees statistical circuit privacy even when the adversary generates a pair of malformed public key and ciphertext. This security notion perfectly fits our setting. Let $(\widetilde{\mathsf{fhe.pk}}, \widetilde{\mathsf{fhe.ct}})$ be a pair of *adversarially generated* public key and ciphertext. A simulator of maliciously circuit private FHE is given $(\widetilde{\mathsf{fhe.pk}}, \widetilde{\mathsf{fhe.ct}}, G(m))$ and can output a ciphertext which is *statistically* indistinguishable from $\mathsf{Eval}(\widetilde{\mathsf{fhe.pk}}, G, \widetilde{\mathsf{fhe.ct}})$ (i.e., a circuit G is applied to $\widetilde{\mathsf{fhe.ct}}$). Here, m is a plaintext extracted from $\widetilde{\mathsf{fhe.ct}}$. Hence, the unbounded authority obtains $f(x)$ and nothing beyond if we set $m := f$ and $G := P[x]$. It is easy to extend this construction to sPCE for Q functions. The resultant public key is linear in Q. Please see Section 3.3 for details.

Implications and Lower bounds for Laconic sPCE**.** It is not hard to observe that achieving sPCE for general constraints with succinct ciphertexts is as hard as achieving indistinguishability obfuscation. Succinct ciphertexts mean the ciphertext size is sublinear in Q or the maximum constraint size. If sPCE has succinct ciphertexts, we can transform such a sPCE into a single-key succinct standard FE, which implies indistinguishability obfuscation [BV18] by using known transformations [KNT21, BV18, LPST16a, LPST16b]. Although sPCE does not have a delegation mechanism, its absence does not prevent the use of these transformations. We then consider laconic sPCE, namely we require that the public key size is sublinear in Q as well as the (maximum) length s of any function. First, we show that it is impossible to achieve maliciously secure laconic sPCE via an incompressibility style argument inspired by the impossibility of simulation-based secure FE [AGVW13]. Here, we sketch this argument for the case $|\mathsf{pk}| = O(Q^{1-\gamma})$ – the argument for the case $|\mathsf{pk}| = O(s^{1-\epsilon})$ is similar. Let u_i be a uniformly random string. Suppose that we generate $(\mathsf{pk}, \mathsf{sk}_1, \ldots, \mathsf{sk}_Q)$ from Q functions $(g[u_1], \ldots, g[u_Q])$ where $g[u_i]$ is a constant function that outputs u_i for any input and $|\mathsf{pk}| = O(Q^{1-\gamma})$ or $O(s^{1-\epsilon})$ for some $0 < \gamma, \epsilon < 1$. If a sPCE scheme satisfies security against malicious authority, there exists a possibly inefficient extractor Ext that extracts $(g[u_1], \ldots, g[u_Q])$ from pk. That is, Ext can recover $(u_1, \ldots, u_Q)$ only from pk. This extraction is information theoretically impossible since $|\mathsf{pk}| = O(Q^{1-\gamma})$, but $\sum_{i=1}^{Q} |u_i| = O(Q)$. See Section 3.4 for the details. Hence, the best security we can achieve for laconic public key constructions is semi-malicious security against authority.

Laconic Semi-Malicious sPCE**.** We now turn to the question of constructing laconic sPCE in the semi-malicious setting. To begin, we consider the simpler security requirement of semi-*honest* security against the authority – recall that in this notion, the setup algorithm is run honestly, but the adversary is allowed to see the random coins used for the execution.

Our starting point is the idea that to achieve laconic public key in a pre-constrained static FE scheme, we can leverage any FE scheme where the running times of setup and key generation algorithms are independent of the number of collusions supported. Hopefully, encryption and decryption can work as in the underlying FE scheme, yielding the desired functionality, and we can then try to adapt the scheme to obtain semi-malicious security.

A construction of bounded key ciphertext policy functional encryption by Agrawal et al. [AMVY21] enjoys the above feature and serves as a useful starting point. At a high level, their construction works as follows. They make use of a reusable dynamic MPC (RDMPC) protocol [AV19], an identity-based encryption (IBE) scheme and a garbled circuits scheme. An RDMPC consists of a single client and N servers where the client offloads an apriori bounded number of computations Q to the N servers in two phases: (i) an offline phase, in which the secret circuit C held by the client is encoded into N shares (via a circuit encoding algorithm), and one share is provided to each of the N servers, (ii) An online phase, in which the client runs an input encoding algorithm on each of its Q inputs and provides this to all the servers. Each server now performs some local computation on its circuit encoding and the given input encodings, after which, any subset S of servers of some minimum size (say n) can combine their partial outputs to obtain the final output of the computation.

To leverage an RDMPC to build FE, [AMVY21] do the following: the setup algorithm runs the setup of an IBE scheme and outputs a public and master secret key. The key generator, given an input f[2], computes its input encoding using the RDMPC scheme. It then samples a random set of servers Δ and provides an IBE secret key corresponding to this set of servers and the given input encoding. The encryptor computes garbled circuits for the RDMPC local computation circuit for each share of the circuit encoding, and encrypts the labels of these garbled circuits using IBE encryption. To decrypt, the user first performs IBE decryption to obtain the labels corresponding to the input encodings and chosen servers, then executes the garbled circuit to perform the RDMPC local computation and then performs the RDMPC final evaluation to recover the desired output.

For our setting of sPCE, we first observe that it is beneficial to use the simpler primitive of hash encryption [DGHM18] in place of IBE for our purposes. A hash encryption scheme is similar to a witness encryption scheme and is specified as follows: there is a hash algorithm that hashes an input x to some short value h, an encryption algorithm, which given the hash, encrypts a message μ against h, a position $i \in [|x|]$ and a bit b, and a decrypt algorithm which, given the preimage x to h, recovers μ if and only if $x_i = b$. While hash encryption is known to imply IBE [DGHM18], it is far better for us as a building block since we desire security against a semi-malicious authority. In more detail, the

[2] Although the construction of [AMVY21] is ciphertext-policy, it will be more useful for us to swap the role of the circuit and the input

construction of HE from LWE by Döttling et al. has a random matrix as its public key and does *not* have any master secret! Thus, it is immediate that this HE is secure against semi-honest authority in the standard model and against semi-malicious authority in the ROM (simply by using the RO to generate the public matrix). This is in stark contrast to an IBE, which also has an MSK and is much harder to secure against a semi-malicious authority.

The careful reader may object that using HE to build IBE does not really help with security against authority since the resultant IBE must nevertheless have some master key. Here, we use a second trick – instead of using IBE, we leverage the static setting of our FE to directly use HE to construct FE. In more detail, we collapse the setup and keygen of [AMVY21] into setup for our sPCE where the functions are pre-specified. We use the RDMPC input encoding to encode the Q functions $f_1, \ldots, f_Q$, then hash the concatenation of these encodings using the hash algorithm of the HE scheme. Since the hash is compressing, the public key is laconic. This hash h is now provided to the encryptor, who computes the shares of the circuit encoding, garbles the local computation circuit and uses HE encryption to encode the labels of the garbled circuits. Decryption proceeds by recovering the garbled circuit labels using HE decryption, and executing the garbled circuits to get the RDMPC partial outputs, which are then combined using the RDMPC combine procedure. The next challenge that is encountered is that the construction obtained via the above route does not satisfy function hiding. We then provide a generic way to lift a construction without function hiding to one with function hiding. Please see Section 3.5 for details.

In the above construction, the only randomness used by setup is in sampling the CRS or public key of the hash encryption scheme. Hence, we immediately get security against authority in the semi-honest setting. Moreover, by shifting to the ROM, we also obtain security against a semi-malicious authority.

Laconic Semi-Malicious IB-PCE and AB-PCE. Next, we provide constructions of laconic semi-malicious PCE with dynamic key generation as defined by AJJM, for the special cases of AB-PCE and IB-PCE . Towards this, we leverage lattice-based techniques and avoid general-purpose tools – as in AJJM we focus specifically on AB-PCE for point constraints and IB-PCE for general constraints. In line with the general template proposed by Ananth et al. [AJJM22], our constructions are based on security reductions for selectively secure advanced cryptographic systems. A security reduction embeds the instance of a hard problem into public parameters, simulates key queries, and breaks the problem using an adversary distinguishing a challenge ciphertext simulated by the reduction. The reduction holds a *constrained (a.k.a. punctured)* master secret key, allowing it to simulate all decryption keys except those that satisfy certain predicates. These constrained master secret keys correspond naturally to pre-constrained decryption keys required by PCE. See Ananth et al. [AJJM22] for a detailed overview. While our constructions fit this template, we go further by explicitly instantiating and analyzing the underlying constructions in a non-black-box manner. This enables us to obtain laconic public key, strictly improving the corresponding constructions by AJJM. Below, we outline our key technical insights.

Our laconic AB-PCE for point constraints builds on on Wee's ABE scheme [Wee25].[3] The public parameter and master secret key consist of matrices $(\mathbf{B}, \mathbf{W}, \mathbf{T}, \mathbf{B}_1)$ and $\mathbf{T}_\mathbf{B}$, respectively, where $\mathbf{W}$ and $\mathbf{B}_1$ are uniformly random matrices, $\mathbf{B}$ is sampled along with a trapdoor $\mathbf{T}_\mathbf{B}$, and $\mathbf{T}$ is a matrix such that $[\mathbf{I}_{2m^2} \otimes \mathbf{B} \;\; \mathbf{W}] \cdot \mathbf{T} = (\mathbf{I}_{2m^2} \otimes \mathbf{G})$, where $\mathbf{G}$ is the gadget matrix. The ciphertext w.r.t. an attribute $\mathbf{x}$ is $(\mathbf{sB} + \mathbf{e}_0, \mathbf{s}(\mathbf{B}_1 + \mathbf{C}_\mathbf{x}) + \mathbf{e}_1)$, where $\mathbf{C}_\mathbf{x}$ is a commitment to an attribute $\mathbf{x}$[4] – we describe how to mask the plaintext later. A decryption key is $\mathbf{D}$ such that $[\mathbf{B}\ \mathbf{A}_f]\mathbf{D} = \mathbf{0}$, where $\mathbf{A}_f$ is computed from $\mathbf{A}$ and a policy f via the well-known key-homomorphic property [BGG+14], $\mathbf{A} = -\mathbf{B}_1\mathbf{V}_\ell$, and $\mathbf{V}_\ell$ is a public verification matrix for commitment $\mathbf{C}_\mathbf{x}$. Following the template of Ananth et al., we can use the reduction algorithm by Wee [Wee25] to implement a constrained master secret key. We set $\mathbf{B}_1 = \mathbf{BU} - \mathbf{C}_{\mathbf{x}^*}$, where $\mathbf{x}^*$ is the constraint point, and can sample $\mathbf{D}$ above without $\mathbf{T}_\mathbf{B}$ if $f(\mathbf{x}^*) \neq 1$ since $[\mathbf{B} \;\; \mathbf{A}_f] \begin{pmatrix} (\mathbf{Z}_{\mathbf{x}^*} + \mathbf{UV}_\ell) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}^*} \\ \mathbf{I}_m \end{pmatrix} = f(\mathbf{x}^*)\mathbf{G}$ holds by the property of the commitment and $\mathbf{H}_{\mathbf{A},f,\mathbf{x}^*}$, where $\mathbf{Z}_{\mathbf{x}^*}$ is a decommitment for $\mathbf{C}_{\mathbf{x}^*}$ and $\mathbf{H}_{\mathbf{A},f,\mathbf{x}^*}$ is computed from $\mathbf{A}$, f, and $\mathbf{x}^*$ via the well-known key-homomorphic property [BGG+14] (see Section 4 for the detail). To prevent adversaries from using the trapdoor $\mathbf{T}_\mathbf{B}$ and achieve semi-malicious security, following [AJJM22], we set $\mathbf{B}$ to a structured matrix $\begin{bmatrix} \bar{\mathbf{B}} \\ \mathbf{S}_\mathbf{B} \cdot \bar{\mathbf{B}} + \mathbf{E}_\mathbf{B} \end{bmatrix}$ which is guaranteed to have no trapdoor. Hence the encoding of form $\mathbf{sB} + \mathsf{noise}$ is lossy for $\mathbf{s}$, ensuring that $\mathbf{s}$ retains sufficient min-entropy, and we can hide the plaintext using a randomness extractor seeded by $\mathbf{s}$.

[3] The scheme described below is slightly different from the original scheme [Wee25], but sufficient for our purpose.

[4] Although we refer to some values of the commitment below ($\mathbf{C}_\mathbf{x}$, $\mathbf{Z}_\mathbf{x}$, and $\mathbf{V}_\ell$), the detail of the commitment is not critical here.

However, Wee's scheme introduces additional structure to achieve succinctness, which requires special care in our adaptation. Specifically, Wee's scheme uses the additional matrix $\mathbf{W}$ to compress public parameters and ciphertexts alongside base matrices $\mathbf{B}$ and $\mathbf{B}_1$. His scheme needs to sample the pre-image matrix $\mathbf{T}$ using a trapdoor of $\mathbf{B}$. This poses a challenge in our setting setting: We cannot safely use $\mathbf{T}_\mathbf{B}$ to sample a public parameter and punctured master secret key, as this would violate semi-malicious security. Our key insight is that we can instead use a trapdoor of $\mathbf{W}$ to sample the required pre-image matrix. Importantly, revealing a trapdoor of $\mathbf{W}$ does not compromise security, as $\mathbf{W}$ serves solely for parameter compression, not for security. Proving the constraint-hiding property follows from the standard LWE assumption and the leftover hash lemma. We switch $\mathbf{B}$ above into a uniformly random $\mathbf{B}$ by the LWE assumption. Then, we can use $\mathbf{T}_\mathbf{B}$ to sample $\mathbf{D}$, and $\mathbf{BU}$ is uniformly random by the leftover hash lemma. Hence, $\mathbf{B}_1$ hide information about $\mathbf{x}^*$. As a result, we can adapt Wee's reduction into a constrained master secret key generation algorithm, achieving AB-PCE for point constraints with laconic public keys and ciphertexts. Similar to [AJJM22], we note that with appropriate modifications to our AB-PCE scheme, we can achieve a laconic IB-PCE scheme for general constraints satisfying security against a semi-malicious authority. We refer the readers to Section 4 and Appendix C for details.

Pre-Constrained Group Signatures. Bartusek et al. [BGJP23] define the notion of set pre-constrained group signatures which can be implemented in an end-to-end secure messaging application. This primitive allows to encode a set of predefined illegal content into the public key of a group signature scheme. The main idea is that if a user signs a message that belongs to the predefined illegal set, then the user can be de-anonymized by the group manager. On the contrary, signers of messages outside this set remain anonymous even to the group manager.

We generalize the notion of pre-constrained group signatures (PCGS) to support general constraints beyond set membership. Our definitions for PCGS are game-based instead of ideal-functionality-based definitions of [BGJP23]. We improve the PCGS constructed by [BGJP23] by (i) supporting general constraints, (ii) achieving unconditional anonymity and unlinkability against authority, (iii) obtaining plausible post-quantum security. The construction by Bartusek et al. [BGJP23] designs an SPC group signature scheme from an SPC encryption scheme plus standard cryptographic tools, namely, a one-way function, a digital signature scheme, and a zero-knowledge non-interactive argument of knowledge. Our construction follows the same broad outline except that we use our general PCE and dual-mode NIZK instead of SPC encryption and standard NIZK argument of knowledge, respectively. Additional details need care to handle – for instance, we must use a dual-mode NIZK to achieve unconditional security against malicious authority. In the end, we obtain a PCGS for general constraints against unbounded authorities from the LWE assumption. We refer the reader to Appendix D for the details.

Pre-Constrained Input Obfuscation. We define the notion of pre-constrained input obfuscation and provide a construction. A PCIO has three algorithms, namely $(\mathsf{Setup}, \mathcal{O}, \mathsf{Eval})$. Setup takes as input a security parameter and an input-set $\mathcal{X} := (x_1, \ldots, x_Q)$, and outputs a public key and evaluation key ek. $\mathcal{O}$ takes pk and a circuit C and outputs an obfuscated circuit $\widetilde{C}$. Eval takes ek, $\widetilde{C}$, and x', and outputs y. Correctness posits that if $x' \in \mathcal{X}$, $y = C(x')$.

We observe that PCIO is the dual of sPCE. We can obtain PCIO from sPCE if we set $f_i := U[x_i]$ where $U[x_i]$ is a universal circuit that takes a circuit C and outputs $C(x_i)$, $\mathsf{ek} := (\mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q})$, and encrypt C in sPCE. PCIO should have input-set-hiding and virtual black-box security against malicious authority. The former and latter correspond to function-hiding and security against malicious authority, respectively. On the other side, we can also obtain sPCE from PCIO via a universal circuit. Please see Appendix E for details.

Other Related Work. Next we discuss additional notions related to pre-constrained cryptography.

Privacy Preserving Blueprints. The recent work of Kohlweiss et al. [KLN23] also addresses the question of pre-constraining in the context of anonymous credentials. Their primary motivating example is in anonymous e-cash – here, there is a bank that issues e-coins, users who withdraw and spend these coins, and vendors who verify and accept e-coins as payment. Suppose we want an authority to be able to "watch" suspected users for financial fraud. We would like to have a mechanism which will enable an auditor to trace the transactions of these suspected users (on a "watchlist") without revealing the contents of the watchlist or violating the privacy of honest users.

Kohlweiss et al. suggest to enhance the anonymous transaction between the user and verifier with a "privacy preserving blueprint" which allows the user, engaging in a transaction with the verifier, to compute an "escrow" Z which can convince the auditor that s/he is not on the (secret) watchlist without revealing anything else. The watchlist is encoded in a parameter PK_A which is published by the auditor in advance. The verifier verifies that the escrow Z is consistent with the credentials embedded in the e-coins being provided by the user, and rejects the transaction otherwise. At a high level, privacy preserving blueprints can be seen as embedding accountability into a proof system, where there is a user/prover, a vendor/verifier and an auditor who must learn some constrained function about the user's anonymous credentials. On the other hand, PCFE and SDE embed accountability into encryption.

Kohlweiss et al. also consider security against authority, the auditor in this case, and take care to ensure that even a malicious auditor cannot create a blueprint that corresponds to an unauthorized input – for instance, an honest user who is not in the real watchlist. However, their definition relies on trusted parameters generated by an honest setup algorithm – in particular, if the auditor generates the parameters, then security against a malicious auditor cannot be guaranteed[5]. Thus, the reliance on a trusted party is crucial in their notion.

Conditional disclosure of secrets. We mention the related primitive of conditional disclosure of secrets (CDS). While CDS bears some similarities in syntax to PCE, it is a fundamentally different primitive. In particular, PCE generalizes basic PKE by setting constraint C as the constant function that always outputs 1. It is not known how to achieve PKE only from CDS.

In more detail, in conditional disclosure of secrets [GIKM00], Alice and Bob have access to a *joint source of randomness* r and secret s, and compute $F_1(x, s, r)$ and $F_2(y, s, r)$ from inputs x and y, respectively. Here, x and y are public. They can send some secret s to Carol if $f(x, y) = 1$ using their joint randomness. If $f(x, y) = 0$, Carol cannot obtain any information about s (and r). In PCE, the two parties, i.e., parties running Setup and Enc, do *not* have common randomness. Setup has input a constraint C (unlike CDS, this must be hidden) and outputs pk and sk, Enc has input x and message m and uses pk to generate ct, and Dec uses sk to recover m if $U(C, x) = 1$ (where U is the universal circuit). Here, the output pk of the first party can be used an unbounded number of times for encryption of different (x_i, m_i) whereas CDS is a one-time primitive. CDS can be seen as a weak symmetric key, one time attribute based encryption scheme and does not satisfy any of the scenarios targeted by our work (or by the work of Ananth et al. [AJJM22]).

Group signature with Message-Dependent Opening (GS-MDO). We also compare our pre-constrained group signatures to the notion of group signature with message-dependent opening (GS-MDO). As discussed in [BGJP23], in a GS-MDO scheme [OSEH13], the trust is divided between two entities : the group manager and the admitter. Both entities need to pool their private information in order to trace a user from a signature. We note that in this notion, if the group manager and the admitter collaborate in a malicious way, they can open any signature to reveal the user's identity. In our pre-constrained group signature scheme, even a malicious group manager cannot open any signature if the underlying message does not satisfy the constraint which was committed to during the setup phase.

Exceptional Access for Law Enforcement. A line of work [Sav18, GKVL21, GSW23] has provided approaches to allowing law enforcement agencies exceptional and controlled access to private user data. As noted by [AJJM22], the work of Green et al. [GKVL21] can be seen as an IB-PCE scheme. We also mention the work of [GP17, FPS+18] which seeks to enforce accountability on secret laws. These works study questions in the domain of privacy versus accountability but use very different assumptions and techniques than ours.

2 Preliminaries

In this section we define the notation and preliminaries used in our work.

[5] Indeed, there is an attack against the scheme if the auditor generates the trusted parameters. Concretely, they use Pedersen commitment (in Definition 1) to instantiate the commitment scheme and cpar includes group elements for Pedersen commitment. If the auditor knows the discrete log of the group elements, it can easily break the soundness of the blueprint scheme since the adversary can generate a fake commitment that can be opened to an arbitrary value by using the discrete log.

Notation. We use bold letters to denote vectors and the notation $[a, b]$ to denote the set of integers $\{k \in \mathbb{N} \mid a \leq k \leq b\}$. We use $[n]$ to denote the set $[1, n]$. Concatenation is denoted by the symbol $\|$. We say a function $f(n)$ is *negligible* if it is $O(n^{-c})$ for all $c > 0$, and we use $\mathsf{negl}(n)$ to denote a negligible function of n. We say $f(n)$ is *polynomial* if it is $O(n^c)$ for some constant $c > 0$, and we use $\mathrm{poly}(n)$ to denote a polynomial function of n. We use the abbreviation PPT for probabilistic polynomial-time. We say an event occurs with *overwhelming probability* if its probability is $1 - \mathsf{negl}(n)$. Let $\mathbb{Z}$ be the set of all integers. For any integer q, denote $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$. For any discrete distributions P and Q, we let $\mathsf{SD}(P, Q)$ denote the statistical distance between P and Q, i.e., $\mathsf{SD}(P, Q) = \sum_i |\Pr[P = i] - \Pr[Q = i]|/2$. We use $P \approx_c Q$ (resp., $P \approx_s Q$) denotes that they are computationally indistinguishable for any PPT algorithm (resp., statistically indistinguishable). For any random variables X and Y, we let $\mathsf{H}_\infty(X) = -\log_2(\min_i \Pr[P = i])$ denote the min-entropy of X, and let $\widetilde{\mathsf{H}}_\infty(X|Y) = -\log\left(\mathbb{E}_y\left[\max_x \Pr[X = x \mid Y = y]\right]\right)$ denote the average conditional min-entropy.

Extractors. An algorithm $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^\ell$ is a seeded strong average-case (k, ε)-extractor, if for any random variables X over $\{0,1\}^n$ and Z with $\widetilde{\mathsf{H}}_\infty(X|Z) \geq k$, then $\mathsf{SD}((\mathrm{Ext}(X, \mathbf{r}), \mathbf{r}, Z), (\mathbf{u}, \mathbf{r}, Z)) < \varepsilon$, where $\mathbf{u} \leftarrow \{0,1\}^\ell$ and $\mathbf{r} \leftarrow \{0,1\}^r$ are sampled uniformly at random.

2.1 Attribute-Based PCE and Identity-Based PCE

Here we provide the definition of an attribute-based pre-constrained encryption scheme ($\mathsf{AB\text{-}PCE}$) and identity-based pre-constrained encryption scheme ($\mathsf{IB\text{-}PCE}$) adapting the syntax from [AJJM22].

Attribute-Based PCE An $\mathsf{AB\text{-}PCE}$ scheme for constrained family $\mathcal{C}$, an attribute universe $\mathcal{X}$, and a function family $\mathcal{F}$ consists of four algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ defined as follows

$\mathsf{Setup}(1^\lambda, C) \to (\mathsf{mpk}, \mathsf{msk}[C])$. The setup algorithm takes a security parameter 1^λ and a constraint $C \in \mathcal{C}$, and outputs a master public key mpk and punctured master secret key $\mathsf{msk}[C]$[6].

$\mathsf{KeyGen}(\mathsf{msk}[C], f) \to \mathsf{sk}_f$. The key generation algorithm takes as input mpk, $\mathsf{msk}[C]$ and a function f. It produces a secret key sk_f if $C(f) = 1$ else it outputs $\perp$.

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, m) \to \mathsf{ct}$. The encryption algorithm takes as input the master public key mpk, an attribute $\mathbf{x}$ and a message m, and outputs a ciphertext ct.

$\mathsf{Dec}(\mathsf{sk}_f, \mathbf{x}, \mathsf{ct}) \to m'$. The decryption algorithm takes as input a secret key sk_f, an attribute $\mathbf{x}$ and a ciphertext ct, and outputs a value m'.

Definition 2.1 (Correctness). An $\mathsf{AB\text{-}PCE}$ scheme is said to be correct if for all $C \in \mathcal{C}$ and every $f \in \mathcal{F}$ such that $C(f) = 1$, $(\mathsf{mpk}, \mathsf{msk}[C]) \leftarrow \mathsf{Setup}(1^\lambda, C)$ and $f(\mathbf{x}) = 0$, the following holds

$$\Pr[\mathsf{Dec}(\mathsf{KeyGen}(\mathsf{msk}[C], f), \mathbf{x}, \mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, m)) = m] \geq 1 - \mathsf{negl}(\lambda).$$

Definition 2.2 (Security Against Semi-Malicious Authority). An $\mathsf{AB\text{-}PCE}$ scheme is said to satisfy indistinguishability against a semi-malicious authority if for any admissible PPT adversary $\mathcal{A}$, the following holds

$$\Pr\left[\mathcal{A}(\mathsf{ct}_\beta) = \beta : \begin{array}{l} C, r \in \{0,1\}^\lambda \leftarrow \mathcal{A}(1^\lambda); \\ (\mathsf{mpk}, \mathsf{msk}[C]) \leftarrow \mathsf{Setup}(1^\lambda, C; r); \\ (m_0, m_1) \leftarrow \mathcal{A}(\mathsf{mpk}, \mathsf{msk}[C]); \\ \beta \leftarrow \{0,1\}; \mathsf{ct}_\beta \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, m_\beta) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Here $\mathcal{A}$ is admissible if for all $f \in \mathcal{F}$ satisfying $C(f) = 1$, it holds that $f(\mathbf{x}) = 1$.

[6] we assume mpk is implicitly contained in msk.

Definition 2.3 (Constraint Hiding). An $\mathsf{AB\text{-}PCE}$ scheme is said to satisfy constraint hiding if for any admissible PPT adversary $\mathcal{A}$,

$$\Pr\left[\begin{array}{c} b \leftarrow \{0,1\},\ (C_0, C_1) \leftarrow \mathcal{A}(1^\lambda), \\ (\mathsf{mpk}, \mathsf{msk}[C_b]) \leftarrow \mathsf{Setup}(1^\lambda, C_b) \\ \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk}[C_b],\cdot)}(\mathsf{mpk}) = b \end{array}\right] < \frac{1}{2} + \mathsf{negl}(\lambda),$$

where $\mathcal{A}$ is said to be admissible if $|C_0| = |C_1|$, $C_0, C_1 \in \mathcal{C}$, and every function key query f it issues to the key generation oracle satisfies $C_0(f) = C_1(f)$.

Identity-Based PCE An $\mathsf{IB\text{-}PCE}$ scheme is defined with respect to a constraint family $\mathcal{C}$ and an identity universe $\mathcal{I}$. Here the messages are encrypted under identities and the constraint C determines which identities are authorized for decryption, i.e., only if $C(\mathsf{id}) = 1$ can the authority generate a decryption key for identity id. The syntax and the security definitions of a $\mathsf{IB\text{-}PCE}$ scheme are the same as that of an $\mathsf{AB\text{-}PCE}$ scheme.

2.2 Reusable, Dynamic MPC Protocol

Here we provide the definition of a reusable, dynamic multi-party computation (RDMPC) protocol, adapting the syntax from [AMVY21], for circuit class $\mathcal{C}_{\mathsf{inp}}$ consisting of circuits with input length $\mathsf{inp} = \mathsf{inp}(\lambda)$. The protocol is further associated with polynomial functions $N = N(\lambda, Q)$, $n = n(\lambda, Q)$, and $t = t(\lambda, Q)$.

$\mathsf{CktEnc}(1^\lambda, 1^Q, 1^{\mathsf{inp}}, C) \to (\widehat{C}_1, \ldots, \widehat{C}_N)$**.** The circuit encoding algorithm takes as input the security parameter λ, the number of sessions Q, input length of circuit inp, and a circuit $C \in \mathcal{C}_{\mathsf{inp}}$. It then outputs an encoding $(\widehat{C}_1, \ldots, \widehat{C}_N)$ of the circuit C.

$\mathsf{InpEnc}(1^\lambda, 1^Q, 1^{\mathsf{inp}}, \mathbf{x}) \to \hat{\mathbf{x}}$**.** The input encoding algorithm takes as input the security parameter λ, the number of sessions Q, input length of circuit inp, and an input $\mathbf{x} \in \{0,1\}^{\mathsf{inp}}$. It then outputs an encoding $\hat{\mathbf{x}}$ of the input $\mathbf{x}$.

$\mathsf{Local}(\widehat{C}_u, \hat{\mathbf{x}}) \to \widehat{y}_u$**.** The local computation algorithm takes as input the u-th encoding $\widehat{C}_u$ of C and an encoding $\hat{\mathbf{x}}$ of $\mathbf{x}$ and outputs $\widehat{y}_u$. We assume that this algorithm is deterministic.

$\mathsf{Decode}(\{\widehat{y}_u\}_{u \in S}, S) \to z$**.** The decoding algorithm takes as input a set of encodings $\{\widehat{y}_u\}_{u \in S}$ and a set $S \subseteq [N]$ and outputs z.

Definition 2.4 (Correctness). An RDMPC protocol with parameter (N, n, t) is correct if for all $\mathsf{inp} \in \mathbb{N}$, $\mathbf{x} \in \{0,1\}^{\mathsf{inp}}$, $C \in \mathcal{C}_{\mathsf{inp}}$, and set $S \subset [N]$ of size n, we have

$$\Pr\left[\begin{array}{c} (\widehat{C}_1, \ldots, \widehat{C}_N) \leftarrow \mathsf{CktEnc}(1^\lambda, 1^Q, 1^{\mathsf{inp}}, C), \\ \hat{\mathbf{x}} \leftarrow \mathsf{InpEnc}(1^\lambda, 1^Q, 1^{\mathsf{inp}}, \mathbf{x}), \\ \mathsf{Decode}\left(\left\{\mathsf{Local}(\widehat{C}_u, \hat{\mathbf{x}})\right\}_{u \in S}, S\right) = C(\mathbf{x}) \end{array}\right] = 1$$

where probability is taken over the random coins of CktEnc, InpEnc and Decode.

Definition 2.5 (Security). For a RDMPC protocol for the circuit family $\mathcal{C}_{\mathsf{inp}}$ with parameter (N, n, t), a stateful PPT adversary $\mathcal{A}$, a simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$, and a coin $\beta \in \{0,1\}$ consider the following experiment $\mathsf{Expt}^{\mathsf{RDMPC}}_{\beta,\mathcal{A}}(1^\lambda)$:

1. **Setup phase:** On input 1^λ, $\mathcal{A}$ submits the query bound 1^Q and input length 1^{inp} of the circuits to the challenger. Note that this defines the total number of parties $N = N(\lambda, Q)$, number of parties $n = n(\lambda, Q)$ participating in any session, threshold $t = t(\lambda, Q)$. The adversary $\mathcal{A}$ also chooses $S_{\mathsf{crr}} \subset [N]$ of size at most t and sets $\Delta^{(1)}, \ldots, \Delta^{(Q)} \subseteq [N]$ such that $|\Delta^{(i)}| = n$ and submits it to the challenger.

2. **Query phase:** During the game, $\mathcal{A}$ is allowed to make a total of Q input encoding queries. First, it makes $R_1 \leq Q$ adaptive input encoding queries. Namely, when $\mathcal{A}$ makes the i-th input encoding query $\mathbf{x}^{(i)}$ with $i \leq Q$, the challenger runs $\hat{\mathbf{x}}^{(i)} \leftarrow \mathsf{InpEnc}(1^\lambda, 1^Q, 1^{\mathsf{inp}}, \mathbf{x}^{(i)})$ and returns $\hat{\mathbf{x}}^{(i)}$ to $\mathcal{A}$.

3. **Challenge phase:** During the game, $\mathcal{A}$ is allowed to make single circuit encoding query. When $\mathcal{A}$ submits a circuit $C \in \mathcal{C}_{\mathsf{inp}}$, the challenger proceeds as follows.

   - **Real World.** If $\beta = 0$, the challenger runs $(\widehat{C}_1, \ldots, \widehat{C}_N) \leftarrow \mathsf{CktEnc}(1^\lambda, 1^Q, 1^{\mathsf{inp}}, C)$ and returns $\left(\left\{\widehat{C}_j\right\}_{j \in S_{\mathsf{crr}}}, \left\{\mathsf{Local}(\widehat{C}_j, \hat{\mathbf{x}}^{(i)})\right\}_{i \in [R_1], j \in \Delta^{(i)}}\right)$ to $\mathcal{A}$.
   - **Ideal World.** If $\beta = 1$, we define $\mathcal{V}$ as $\mathcal{V} = \{C(\mathbf{x}^{(i)}), \mathbf{x}^{(i)}\}_{i \in [R_1]}$. Then, the simulator is run as $\left(\mathsf{st}, \left\{\widehat{C}_j\right\}_{j \in S_{\mathsf{crr}}}, \left\{\widehat{y}_j^{(i)}\right\}_{i \in [R_1], j \in \Delta^{(i)}}\right) \leftarrow \mathsf{Sim}_0(1^{|C|}, S_{\mathsf{crr}}, \mathcal{V})$ and the output is returned to $\mathcal{A}$. Here, st is the internal state of the simulator.

4. **Query phase:** $\mathcal{A}$ then makes $R_2 \leq Q - R_1$ input encoding queries. When $\mathcal{A}$ submits an input $\mathbf{x}^{(i)} \in \{0,1\}^{\mathsf{inp}}$, the challenger proceeds as follows.

   - **Real World.** If $\beta = 0$, the challenger runs $\hat{\mathbf{x}}^{(i)} \leftarrow \mathsf{InpEnc}(1^\lambda, 1^Q, 1^{\mathsf{inp}}, \mathbf{x}^{(i)})$ and returns $\left(\hat{\mathbf{x}}^{(i)}, \left\{\widehat{C}_j(\hat{\mathbf{x}}^{(i)})\right\}_{j \in \Delta^{(i)}}\right)$ to $\mathcal{A}$.
   - **Ideal World.** If $\beta = 1$, we run the simulator as $\left(\hat{\mathbf{x}}^{(i)}, \left\{\widehat{y}_j^{(i)}\right\}_{j \in \Delta^{(i)}}\right) \leftarrow \mathsf{Sim}_1(\mathsf{st}, \Delta^{(i)}, C(\mathbf{x}^{(i)}), \mathbf{x}^{(i)})$ and returns the output to $\mathcal{A}$.

5. **Output phase:** $\mathcal{A}$ outputs a guess bit β' as the output of the experiment.

We say that an RDMPC protocol is secure if for every adversary $\mathcal{A}$, there exists a PPT simulator Sim such that

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{RDMPC}}(\lambda) = \left|\Pr\left[\mathsf{Expt}_{0,\mathcal{A}}^{\mathsf{RDMPC}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}_{1,\mathcal{A}}^{\mathsf{RDMPC}}(1^\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

Theorem 2.6 (Adapted from [AV19]). Assuming the existence of one-way functions, there exists an RDMPC protocol with parameter $N = \Theta(Q^2\lambda)$, $t = \Theta(Q\lambda)$, and $n = \Theta(t)$ for $\mathcal{C}_{\mathsf{inp}}$ with any $\mathsf{inp} = \mathrm{poly}(\lambda)$.

2.3 Hash Encryption

Here we provide the definition of a hash encryption (HE) scheme from [DGHM18].
A HE scheme consists of four algorithms $(\mathsf{Gen}, \mathsf{Hash}, \mathsf{Enc}, \mathsf{Dec})$ with the following syntax.

$\mathsf{Gen}(1^\lambda, m) \to \mathsf{key}$. The generation algorithm takes as input the security parameter, input parameter m and outputs a key key,

$\mathsf{Hash}(\mathsf{key}, \mathbf{x}) \to h$. The hashing algorithm takes as input a key key, an input $\mathbf{x} \in \{0,1\}^m$ and outputs a hash value h of λ bits.

$\mathsf{Enc}(\mathsf{key}, (h, i, c), \mu) \to \mathsf{ct}$. The encryption algorithm takes as input a key key, a hash value h, an index $i \in [m]$, a bit $c \in \{0,1\}$, and a message $\mu \in \{0,1\}^*$ and outputs a ciphertext ct.

$\mathsf{Dec}(\mathsf{key}, \mathbf{x}, \mathsf{ct}) \to \mu'$. The decryption algorithm takes as input a key key, an input $\mathbf{x} \in \{0,1\}^m$, and a ciphertext ct and outputs $\mu' \in \{0,1\}^* \cup \{\perp\}$.

Definition 2.7 (Correctness). A HE scheme is said to be correct if for any input $\mathbf{x} \in \{0,1\}^m$ and index $i \in [m]$, the following holds

$$\Pr[\mathsf{Dec}(\mathsf{key}, \mathbf{x}, \mathsf{Enc}(\mathsf{key}, (\mathsf{Hash}(\mathsf{key}, \mathbf{x}), i, x_i), \mu)) = \mu] \geq 1 - \mathsf{negl}(\lambda)$$

where x_i denotes the i-th bit of $\mathbf{x}$ and the randomness is taken over $\mathsf{key} \leftarrow \mathsf{Gen}(1^\lambda, m)$.

Definition 2.8 (Semi-Honest Security). For a HE scheme and an adversary $\mathcal{A}$, we consider the following experiment $\mathsf{Expt}_{\beta,\mathcal{A}}^{\mathsf{HE}}(1^\lambda)$.

1. $\mathcal{A}$ outputs an input $\mathbf{x} \in \{0,1\}^m$.
2. The challenger generates $\mathsf{key} \leftarrow \mathsf{Gen}(1^\lambda, m)$ using some randomness R and sends key to $\mathcal{A}$.
3. $\mathcal{A}$ outputs an index $i \in [m]$, $c \in \{0,1\}$, such that $x_i \neq c$ and two messages μ_0, μ_1.
4. The challenger samples $\beta \leftarrow \{0,1\}$ and returns ct_β to $\mathcal{A}$ where $\mathsf{ct}_\beta \leftarrow \mathsf{Enc}(\mathsf{key}, \mathsf{Hash}(\mathsf{key}, \mathbf{x}), i, c), \mu_\beta)$.
5. $\mathcal{A}$ outputs a guess bit β' as the output of the experiment.

We define the advantage $\mathsf{Adv}^{\mathsf{HE}}_{\mathcal{A}}(\lambda)$ of $\mathcal{A}$ in the above game as

$$\mathsf{Adv}^{\mathsf{HE}}_{\mathcal{A}}(\lambda) := \left| \Pr\left[\mathsf{Expt}^{\mathsf{HE}}_{0,\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{HE}}_{1,\mathcal{A}}(1^\lambda) = 1\right] \right|.$$

We say that a HE scheme is selectively secure if for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{HE}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

Definition 2.9 (Succinctness). We say that a HE scheme is *succinct* if $|\mathsf{key}| = \text{poly}(\lambda)$, where $\mathsf{key} \leftarrow \mathsf{Gen}(1^\lambda, m)$, i.e., the size of key is independent of the size of input $\mathbf{x} \in \{0,1\}^m$.

Definition 2.10 (Semi-Malicious security). We say that a HE scheme satisfies *semi-malicious* security if it is secure in the above sense even if the random coins R used by the challenger to generate key in Step 2 are provided by the adversary in Step 1.

Theorem 2.11 ([DGHM18]). There exists a HE scheme satisfying semi-honest security assuming learning with errors (LWE).

Remark 2.12 (Multi-challenge Security). We note that a security game where an adversary can adaptively send polynomially many challenge queries $\{i, \mu_0^{(i)}, \mu_1^{(i)}\}$, for a single challenge input string $\mathbf{x}$, is implied by the single-challenge query security game as defined above via a simple hybrid argument. In the argument we can consider polynomially many single-challenge security sub-hybrids, one for each query.

2.4 Garbling Scheme

Here we provide the definition of a garbling scheme for circuit class $\mathcal{C} = \{C : \{0,1\}^{\ell_{\mathsf{in}}} \to \{0,1\}^{\ell_{\mathsf{out}}}\}$. A garbling scheme for circuit class $\mathcal{C}$ consists of a pair of algorithms $(\mathsf{Garble}, \mathsf{Eval})$ with the following syntax.

$\mathsf{Garble}(1^\lambda, C) \to (\tilde{C}, \mathsf{lb})$. The garbling algorithm takes as input the security parameter λ and a circuit $C \in \mathcal{C}$, and outputs a garbled circuit $\tilde{C}$ and a set of labels $\mathsf{lb} = \{\mathsf{lb}_{i,b}\}_{i\in[\ell_{\mathsf{in}}], b\in\{0,1\}}$.

$\mathsf{Eval}(\tilde{C}, \mathsf{lb}_x) \to \mathbf{y}$. The evaluation algorithm takes as input the garbled circuit $\tilde{C}$ and labels corresponding to an input $\mathbf{x} \in \{0,1\}^{\ell_{\mathsf{in}}}$, $\mathsf{lb}_{\mathbf{x}} = \{\mathsf{lb}_{i,x_i}\}_{i\in[\ell_{\mathsf{in}}]}$ where x_i denotes the i-th bit of $\mathbf{x}$, and it outputs $\mathbf{y} \in \{0,1\}^{\ell_{\mathsf{out}}}$.

A garbling scheme satisfies the following properties.

Definition 2.13 (Correctness). A garbling scheme is said to be correct if for any circuit $C \in \mathcal{C}$ and any input $\mathbf{x} \in \{0,1\}^{\ell_{\mathsf{in}}}$, the following holds

$$\Pr\left[\, \mathbf{y} = C(\mathbf{x}) : (\tilde{C}, \mathsf{lb}) \leftarrow \mathsf{Garble}(1^\lambda, C); \mathbf{y} \leftarrow \mathsf{Eval}(\tilde{C}, \mathsf{lb}_x)\right] = 1.$$

Definition 2.14 (Security). A garbling scheme is secure if there exists a PPT simulator SIM such that for any circuit $C \in \mathcal{C}$ and any input $\mathbf{x} \in \{0,1\}^{\ell_{\mathsf{in}}}$, the following holds

$$\{(\tilde{C}, \mathsf{lb}_x) \mid (\tilde{C}, \mathsf{lb}) \leftarrow \mathsf{Garble}(1^\lambda, C)\} \approx_c \{(\bar{C}, \bar{\mathsf{lb}}) \mid (\bar{C}, \bar{\mathsf{lb}}) \leftarrow \mathsf{SIM}(1^\lambda, C(\mathbf{x}))\}$$

where $\mathsf{lb} = \{\mathsf{lb}_{i,b}\}_{i\in[\ell_{\mathsf{in}}], b\in\{0,1\}}$ and $\mathsf{lb}_x = \{\mathsf{lb}_{i,x_i}\}_{i\in[\ell_{\mathsf{in}}]}$.

Remark 2.15 (Multi-challenge Security). Similar to Remark 2.12, we note that the above security definition implies multi-challenge security of a garbled circuit scheme, where the adversary can adaptively make polynomial number of garbling queries to the challenger,i.e., adversary can query $(C_1, \mathbf{x})$ and receive $(\tilde{C}_1, \mathsf{lb}_x)$, then query $(C_2, \mathbf{y})$, and so forth. All queries are answered by honestly using Garble algorithm in the real world whereas they are all simulated in the ideal world.

2.5 Lattice Preliminaries

Here, we recall some facts on lattices that are needed for the exposition of our construction. Throughout this section, n, m, and q are integers such that $n = \mathrm{poly}(\lambda)$ and $m \geq n\lceil \log q \rceil$. Let

$$\mathbf{g} = (1, 2, ...2^{\lfloor \log q \rfloor})^{\mathsf{T}}\ ,\ \ \mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^{\mathsf{T}}$$

be the gadget vector and the gadget matrix. For $\mathbf{p} \in \mathbb{Z}_q^n$, we write $\mathbf{G}^{-1}(\mathbf{p})$ for the m-bit vector $(\mathsf{bits}(\mathbf{p}[1]), \ldots, \mathsf{bits}(\mathbf{p}[n]))^{\mathsf{T}}$, where $\mathsf{bits}(\mathbf{p}[i])$ are m/n bits for each $i \in [n]$. The notation extends column-wise to matrices and it holds that $\mathbf{G}\mathbf{G}^{-1}(\mathbf{P}) = \mathbf{P}$.

Trapdoors. Let us consider a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$. For all $\mathbf{V} \in \mathbb{Z}_q^{n\times m'}$, we let $\mathbf{A}^{-1}(\mathbf{V})$ be an output distribution of $\mathsf{SampZ}(\gamma)^{m\times m'}$ conditioned on $\mathbf{A} \cdot \mathbf{A}^{-1}(\mathbf{V}, \gamma) = \mathbf{V}$. A γ-trapdoor for $\mathbf{A}$ is a trapdoor that enables one to sample from the distribution $\mathbf{A}^{-1}(\mathbf{V}, \gamma)$ in time $\mathrm{poly}(n, m, m', \log q)$ for any $\mathbf{V}$. We slightly overload notation and denote a γ-trapdoor for $\mathbf{A}$ by $\mathbf{A}_\gamma^{-1}$. The following properties had been established in a long sequence of works [GPV08, CHKP10, ABB10a, ABB10b, MP12, BLP+13].

Lemma 2.16 (Properties of Trapdoors). Lattice trapdoors exhibit the following properties.

1. Given $\mathbf{A}_\tau^{-1}$, one can obtain $\mathbf{A}_{\tau'}^{-1}$ for any $\tau' \geq \tau$.
2. Given $\mathbf{A}_\tau^{-1}$, one can obtain $[\mathbf{A}\|\mathbf{B}]_\tau^{-1}$ and $[\mathbf{B}\|\mathbf{A}]_\tau^{-1}$ for any $\mathbf{B}$.
3. There exists an efficient procedure $\mathsf{TrapGen}(1^n, 1^m, q)$ that outputs $(\mathbf{A}, \mathbf{A}_{\tau_0}^{-1})$ where $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ for some $m = O(n \log q)$ and is 2^{-n}-close to uniform, where $\tau_0 = \omega(\sqrt{n \log q \log m})$.

Useful Lemmata.

Lemma 2.17 (tail and truncation of $\mathcal{D}\mathbb{Z}, \gamma$). There exists $B_0 \in \Theta(\sqrt{\lambda})$ such that

$$\Pr[x \leftarrow \mathcal{D}\mathbb{Z}, \gamma : |x| > \gamma B_0(\lambda)] \leq 2^{-\lambda} \quad \text{for all} \quad \gamma \geq 1 \text{ and } \lambda \in \mathbb{N}.$$

Lemma 2.18 (Smudging Lemma [WWW22]). Let λ be a security parameter. Take any $a \in \mathbb{Z}$ where $|a| \leq B$. Suppose $\gamma \geq B\lambda^{\omega(1)}$. Then the statistical distance between the distributions $\{z : z \leftarrow \mathcal{D}_{\mathbb{Z},\gamma}\}$ and $\{z + a : z \leftarrow \mathcal{D}_{\mathbb{Z},\gamma}\}$ is $\mathsf{negl}(\lambda)$.

Lemma 2.19 (Leftover Hash Lemma). Fix some $n, m, q \in \mathbb{N}$. The leftover hash lemma states that if $m \geq 2n \log q$, then for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{x} \leftarrow \{0, 1\}^m$ and $\mathbf{y} \leftarrow \mathbb{Z}_q^n$ the statistical distance between $(\mathbf{A}, \mathbf{A} \cdot \mathbf{x})$ and $(\mathbf{A}, \mathbf{y})$ is negligible. More concretely, it is bounded by $q^n\sqrt{2^{1-m}}$.

Hardness Assumptions

Assumption 2.20 (The LWE Assumption). Let $n = n(\lambda)$, $m = m(\lambda)$, and $q = q(\lambda) > 2$ be integers and $\chi = \chi(\lambda)$ be a distribution over $\mathbb{Z}_q$. We say that the LWE(n, m, q, χ) hardness assumption holds if for any PPT adversary $\mathcal{A}$ we have

$$|\Pr[\mathcal{A}(\mathbf{A}, \mathbf{s}^{\mathsf{T}}\mathbf{A} + \mathbf{e}^{\mathsf{T}}) \rightarrow 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{v}^{\mathsf{T}}) \rightarrow 1]| \leq \mathsf{negl}(\lambda)$$

where the probability is taken over the choice of the random coins by the adversary $\mathcal{A}$ and $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$, and $\mathbf{v} \leftarrow \mathbb{Z}_q^m$.

2.6 Homomorphic Computation on Matrices

In this section we define homomorphic evaluation matrices from [BGG+14, GSW13], adapting the syntax from [Wee25].

Lemma 2.21 (EvalF, EvalFX **[BGG+14, GSW13]).** Fix lattice parameters n, q and $m \geq 2n \log q$. Let $\mathcal{F}_{\ell,d,s}$ denote the family of functions $f : \{0,1\}^\ell \to \{0,1\}$ computable by circuits of depth d and size s. There exist a pair of efficient algorithms $(\mathsf{EvalF}, \mathsf{EvalFX})$ where:

– $\mathsf{EvalF}(\mathbf{A}, f) \to \mathbf{A}_f$: On input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times \ell m}$ and a function $f \in \mathcal{F}_{\ell,d,s}$, outputs a matrix $\mathbf{A}_f \in \mathbb{Z}_q^{n\times m}$.

– $\mathsf{EvalFX}(\mathbf{A}, f, \mathbf{x}) \to \mathbf{H}_{\mathbf{A},f,\mathbf{x}}$: On input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times \ell m}$, a function $f \in \mathcal{F}_{\ell,d,s}$, and an input $\mathbf{x} \in \{0,1\}^\ell$, outputs a matrix $\mathbf{H}_{\mathbf{A},f,\mathbf{x}} \in \mathbb{Z}^{\ell m\times m}$.

For all $\mathbf{A} \in \mathbb{Z}_q^{n\times \ell m}$, $f \in \mathcal{F}_{\ell,d,s}$, and $\mathbf{x} \in \{0,1\}^\ell$, the matrices $\mathbf{A}_f \leftarrow \mathsf{EvalF}(\mathbf{A}, f)$ and $\mathbf{H}_{\mathbf{A},f,\mathbf{x}} \leftarrow \mathsf{EvalFX}(\mathbf{A}, f, \mathbf{x})$ satisfy:

$$(\mathbf{x} \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} = \mathbf{A}_f - f(\mathbf{x})\mathbf{G} \tag{1}$$

$$\left\|\mathbf{H}_{\mathbf{A},f,\mathbf{x}}\right\| = m^{O(d)} \cdot s.$$

2.7 Commitment to vectors

Here we define succinct commitment scheme for vectors, adapting the syntax from [Wee25].

Lemma 2.22 (Vector commitment). There exist efficient algorithms $(\mathsf{Com}^{\mathsf{vc}}, \mathsf{Ver}^{\mathsf{vc}}, \mathsf{Open}^{\mathsf{vc}})$ such that

- $\mathsf{Com}^{\mathsf{vc}}(\mathsf{pp}, \mathbf{x} \in \mathbb{Z}_q^\ell)$. On input $\mathbf{x}$ outputs $\mathbf{C} \in \mathbb{Z}_q^{n\times m}$,
- $\mathsf{Ver}^{\mathsf{vc}}(\mathsf{pp}, 1^\ell)$. On input 1^ℓ, outputs $\mathbf{V}_\ell \in \mathbb{Z}_q^{m\times \ell m}$,
- $\mathsf{Open}^{\mathsf{vc}}(\mathsf{pp}, \mathbf{x})$. On input $\mathbf{x}$ outputs $\mathbf{Z} \in \mathbb{Z}_q^{m\times \ell m}$.

For all pp, $\ell \in \mathbb{N}$, and $\mathbf{x} \in \mathbb{Z}_q^\ell$, the matrices $\mathbf{C} \leftarrow \mathsf{Com}^{\mathsf{vc}}(\mathsf{pp}, \mathbf{x})$, $\mathbf{V}_\ell \leftarrow \mathsf{Ver}^{\mathsf{vc}}(\mathsf{pp}, 1^\ell)$, $\mathbf{Z} \leftarrow \mathsf{Open}^{\mathsf{vc}}(\mathsf{pp}, \mathbf{x})$ satisfy:

$$\mathbf{C} \cdot \mathbf{V}_\ell = \mathbf{x} \otimes \mathbf{G} - \mathbf{B} \cdot \mathbf{Z}, \tag{2}$$

$$\|\mathbf{V}_\ell\| \leq O(\|\mathbf{T}\| \cdot m^4 \log q), \quad \|\mathbf{Z}\| \leq O(\|\mathbf{T}\| \cdot \log \ell \cdot m^7 \log q).$$

2.8 Statistical Sender-Private Two-Message Oblivious Transfer

Here we provide the definition of a two-message statistically sender-private oblivious transfer (SSP-OT) scheme, adapted from [BD18].
A two-message oblivious transfer scheme, for an input space $\mathcal{I}$, consists of three algorithms $(\mathsf{OTR}, \mathsf{OTS}, \mathsf{OTD})$ with the following syntax.

$\mathsf{OTR}(1^\lambda, \beta) \to (\mathsf{ot}_1, \mathsf{st})$. This algorithm takes as input the security parameter λ and a choice bit $\beta \in \{0,1\}$ and outputs a message ot_1 and a secret state st.

$\mathsf{OTS}(1^\lambda, (\mu_0, \mu_1), \mathsf{ot}_1) \to \mathsf{ot}_2$. This algorithm takes as input the security parameter λ, two inputs $\mu_0, \mu_1 \in \mathcal{I}$, and a message ot_1 and outputs a message ot_2.

$\mathsf{OTD}(1^\lambda, \beta, \mathsf{st}, \mathsf{ot}_2) \to \mu$. This algorithm takes as input the security parameter λ, the bit $\beta \in \{0,1\}$, the secret state st, and the message ot_2 and outputs $\mu \in \mathcal{I}$.

Next, we define the properties satisfied by a statistical sender-private two-message oblivious transfer scheme.

Definition 2.23 (Correctness). A SSP-OT scheme is said to be correct if for any $\lambda \in \mathbb{N}$, and inputs $\mu_0, \mu_1 \in \mathcal{I}$ the following holds.

$$\Pr\left[\begin{array}{ll} \mu = \mu_\beta : & \begin{array}{l} \beta \leftarrow \{0,1\}; (\mathsf{ot}_1, \mathsf{st}) \leftarrow \mathsf{OTR}(1^\lambda, \beta); \\ \mathsf{ot}_2 \leftarrow \mathsf{OTS}(1^\lambda, (\mu_0, \mu_1), \mathsf{ot}_1); \\ \mu = \mathsf{OTD}(1^\lambda, \beta, \mathsf{st}, \mathsf{ot}_2) \end{array} \end{array}\right] = 1.$$

Definition 2.24 (Receiver Privacy). A SSP-OT scheme is said to satisfy receiver privacy if the following two distributions are computationally indistinguishable

$$\{\mathsf{ot}_1 \mid (\mathsf{ot}_1, \mathsf{st}) \leftarrow \mathsf{OTR}(1^\lambda, 0)\} \approx_c \{\mathsf{ot}_1 \mid (\mathsf{ot}_1, \mathsf{st}) \leftarrow \mathsf{OTR}(1^\lambda, 1)\}.$$

Definition 2.25 (Statistical Sender Privacy). A SSP-OT scheme is said to satisfy statistical sender-privacy if there exists an *admissible* unbounded extractor algorithm $\mathsf{OT.Ext}$ such that for any sequence of messages ot_1 output by an unbounded receiver and for any inputs $\mu_0, \mu_1 \in \mathcal{I}$, the following two distributions are statistically indistinguishable

$$\{\mathsf{ot}_2 \mid \mathsf{ot}_2 \leftarrow \mathsf{OTS}(1^\lambda, (\mu_0, \mu_1), \mathsf{ot}_1)\} \approx_s \{\mathsf{ot}_2 \mid \mathsf{ot}_2 \leftarrow \mathsf{OTS}(1^\lambda, (\mu_{\beta'}, \mu_{\beta'}), \mathsf{ot}_1)\}$$

where $\beta' = \mathsf{OT.Ext}(\mathsf{ot}_1)$. We say that $\mathsf{OT.Ext}$ is admissible if for any $\beta \in \{0,1\}$ and randomness r, we have $\beta = \beta'$ where (i) $\mathsf{ot}_1 \leftarrow \mathsf{OTR}(1^\lambda, \beta; r)$ and (ii) $\beta' = \mathsf{OT.Ext}(\mathsf{ot}_1)$.

SSP-OT can be based on a wide variety of assumptions – number-theoretic assumptions such as DDH [AIR01, NP01], QR and DCR [HK12], LWE [BD18, DGI+19, ADD+23], LPN and Nisan-Wigderson style derandomization [BF22].

2.9 Maliciously Circuit-Private FHE

A fully homomorphic encryption (FHE) scheme, for circuit class $\mathcal{C}_n$ of all efficiently computable circuits of input length $n = n(\lambda)$, consists of four algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ with the following syntax.

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{sk})$. The key generation algorithm takes as input the security parameter and outputs a public key pk and a secret key sk.

$\mathsf{Enc}(\mathsf{pk}, m) \rightarrow \mathsf{ct}$. The encryption algorithm takes as input the public key pk and a message $m \in \{0,1\}$, and outputs a ciphertext ct.

$\mathsf{Eval}(\mathsf{pk}, C, c_1, \ldots, c_n) \rightarrow \hat{\mathsf{ct}}$. The evaluation algorithm takes as input the public key pk, a circuit $C \in \mathcal{C}_n$ with input size n and ciphertexts $c_1, \ldots, c_n$, where $c_i \leftarrow \mathsf{Enc}(\mathsf{pk}, m_i)$ for $i \in [n]$, and outputs a ciphertext $\hat{\mathsf{ct}}$.

$\mathsf{Dec}(\mathsf{sk}, c) \rightarrow m$. The decryption algorithm takes as input the secret key sk and a ciphertext c and outputs a message m.

Definition 2.26 (Correctness). A FHE scheme for circuit class $\mathcal{C}_n$ is correct if, for any key-pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, any circuit $C \in \mathcal{C}_n$, any plaintexts $m \in \{0,1\}, m_1 \in \{0,1\}, \ldots, m_n \in \{0,1\}$, the following two condition holds

$$\Pr[m = \mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m))] = 1$$

and

$$\Pr[C(m_1, \ldots, m_n) = \mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, C, \mathsf{Enc}(\mathsf{pk}, m_1), \ldots, \mathsf{Enc}(\mathsf{pk}, m_n)))] = 1.$$

Definition 2.27 (Semantic Security). A FHE scheme is said to satisfy semantic security if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any two messages $m_0, m_1 \in \{0,1\}$, the following holds

$$\Pr\left[\begin{array}{ll} \beta' = \beta : & \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda); \\ \beta \leftarrow \{0,1\}; \mathsf{ct}_\beta \leftarrow \mathsf{Enc}(\mathsf{pk}, m_\beta); \\ \beta' \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{ct}_\beta) \end{array} \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Definition 2.28 (Malicious Circuit Privacy). A FHE scheme is said to satisfy malicious circuit privacy if there exists unbounded simulator Sim, a and an *admissible* deterministic extractor Ext, such that for all $\lambda \in \mathbb{N}$, all $C \in \mathcal{C}_n$ and any adversary $\mathcal{A}$, the following two distributions are statistically indistinguishable

$$\mathsf{Sim}(\mathsf{pk}^*, c_1^*, \ldots, c_n^*, C(m_1^*, \ldots, m_n^*)) \approx_s \mathsf{Eval}(\mathsf{pk}^*, C, c_1^*, \ldots, c_n^*)$$

where $(\mathsf{pk}^*, c_1^*, \ldots, c_n^*) \leftarrow \mathcal{A}(1^\lambda)$ and $(m_1^*, \ldots, m_n^*) = \mathrm{Ext}(\mathsf{pk}^*, c_1^*, \ldots, c_n^*)$. We say that Ext is admissible if for any $(m_1, \ldots, m_n) \in \{0,1\}^n$, keygen randomness r and encryption randomness $\bar{r}$, we have $(m_1, \ldots, m_n) = (m_1', \ldots, m_n')$ where (i) $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda; r)$, (ii) $c_i \leftarrow \mathsf{Enc}(\mathsf{pk}, m_i; \bar{r})$ for all $i \in [n]$ and (iii) $m_i' = \mathrm{Ext}(\mathsf{pk}, c_1, \ldots, c_n)$.

Definition 2.29 (Compactness). A FHE scheme is compact if there exists a fixed polynomial bound $p(\cdot)$ such that for any circuit $C \in \mathcal{C}_n$ and ciphertexts $\{c_i\}_{i \in [n]}$ where $c_i \leftarrow \mathsf{Enc}(\mathsf{pk}, m_i)$, we have $|\hat{\mathsf{ct}}| \leq p(\lambda)$ where $\hat{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathsf{pk}, C, c_1, \ldots, c_n)$, i.e., the size of the evaluated ciphertext is independent of the size of the evaluated circuit.

Definition 2.30 (Linear efficiency). A FHE scheme satisfies linear efficiency if the ciphertext size is linear in the plaintext size. That is, $|\mathsf{ct}| = \mathrm{poly}(\lambda) \cdot |m|$ where $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$.

Ostrovsky et al. [OPP14] showed that any compact FHE scheme can be converted to one that satisfies malicious circuit privacy using statistically sender private OT and Brakerski and Dottling [BD18] showed how to achieve statistically sender private OT from the Learning With Errors (LWE) assumption. Also, the resulting scheme retains the linear efficiency property of the underlying compact FHE scheme. Since compact FHE with linear efficiency can also be constructed from LWE [GSW13], we obtain the following:

Theorem 2.31 ([OPP14, BD18, GSW13]). There exists a malicious circuit-private compact fully homomorphic encryption scheme with linear efficiency from the polynomially hard learning with errors (LWE) assumption.

3 Static Pre-Constrained Encryption

As discussed in Section 1, we introduce *static* pre-constrained encryption (sPCE). This notion is a relaxation of PCE introduced by Ananth et al. [AJJM22]. The big difference is that our sPCE does not have a delegation mechanism. Although we can generate functional decryption keys, all functions are fixed at the setup phase.

3.1 Definition

A static pre-constrained encryption scheme (sPCE) for a function family $\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y}\}$ with input space $\mathcal{X}$ and output space $\mathcal{Y}$ consists of three algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ defined as follows.

$\mathsf{Setup}(1^\lambda, \{f_1, \ldots, f_Q\}) \to (\mathsf{pk}, \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q})$. The setup algorithm takes as input the security parameter λ, and functions $f_1, \ldots, f_Q \in \mathcal{F}$, and returns the public key pk and secret keys $\mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q}$.

$\mathsf{Enc}(\mathsf{pk}, x) \to \mathsf{ct}$. The encryption algorithm takes as input the public key pk and an input $x \in \mathcal{X}$, and outputs a ciphertext ct.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to y$. The decryption algorithm takes as input a secret key sk and a ciphertext ct, and outputs $y \in \mathcal{Y}$.

Definition 3.1 (Correctness). A sPCE scheme is said to be correct if for any $f \in \mathcal{F}$ and $x \in \mathcal{X}$, the following holds

$$\Pr\Big[\mathsf{Dec}(\mathsf{sk}_{f_i}, \mathsf{Enc}(\mathsf{pk}, x)) = f_i(x)\Big] \geq 1 - \mathsf{negl}(\lambda)$$

where $i \in [Q]$ and $(\mathsf{pk}, \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q}) \leftarrow \mathsf{Setup}(1^\lambda, \{f_1, \ldots, f_Q\})$.

If correctness holds with probability 1, the scheme is said to be *perfectly* correct.

Definition 3.2 (Function-Hiding). A sPCE scheme is said to satisfy function-hiding security if for any PPT adversary $\mathcal{A}$, the following holds

$$\Pr\left[\begin{array}{ll} \mathcal{A}(\mathsf{pk}_\beta) = \beta : & \{(f_1^0, f_1^1), \ldots, (f_Q^0, f_Q^1)\} \leftarrow \mathcal{A}; \\ & \beta \leftarrow \{0,1\}; \\ & (\mathsf{pk}_\beta, \mathsf{sk}^\beta_{f_1}, \ldots, \mathsf{sk}^\beta_{f_Q}) \leftarrow \mathsf{Setup}(1^\lambda, \{f_1^\beta, \ldots, f_Q^\beta\}); \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ is admissible if $|f_i^0| = |f_i^1|$ and $f_i^0, f_i^1 \in \mathcal{F}$ for all $i \in [Q]$.

Definition 3.3 (SIM Security Against Semi-Malicious Authority). For a sPCE scheme, an adversary $\mathcal{A}$ and a PPT simulator SIM we define the experiment for security against semi malicious authority $\mathsf{Expt}^{\mathsf{SMS}}_{\beta,\mathcal{A}}(1^\lambda)$ as follows.

1. $\mathcal{A}$ outputs functions $\{f_1, \ldots, f_Q\}$ and randomness $r \in \{0,1\}^\lambda$.

2. On input $1^\lambda, \{f_1, \ldots, f_Q\}$ and randomness r, the challenger generates $(\mathsf{pk}, \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q}) \leftarrow \mathsf{Setup}(1^\lambda, \{f_1, \ldots, f_Q\}; r)$. It sends the public key pk to $\mathcal{A}$.

3. $\mathcal{A}$ outputs x. the challenger samples $\beta \leftarrow \{0,1\}$. If $\beta = 0$, it computes $\mathsf{ct}_0 \leftarrow \mathsf{Enc}(\mathsf{pk}, x)$ else if $\beta = 1$, it computes $\mathsf{ct}_1 \leftarrow \mathsf{SIM}(\mathsf{pk}, 1^{|x|}, \{f_1(x), \ldots, f_Q(x)\})$. It sends ct_β to $\mathcal{A}$.

4. $\mathcal{A}$ outputs a guess bit β' as the output of the experiment.

We define the advantage $\mathsf{Adv}^{\mathsf{SMS}}_{\mathcal{A}}(\lambda)$ of $\mathcal{A}$ in the above game as

$$\mathsf{Adv}^{\mathsf{SMS}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{Expt}^{\mathsf{SMS}}_{0,\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{SMS}}_{1,\mathcal{A}}(1^\lambda) = 1\right]\right|.$$

We say that a sPCE scheme satisfies security against a semi malicious authority if there exists a PPT simulator SIM such that for every PPT adversary $\mathcal{A}$, we have $\mathsf{Adv}^{\mathsf{SMS}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

Definition 3.3 is slightly strong compared with the simulation-based security of standard FE [GVW12] in the sense that Sim takes only $(f_1(x), \ldots, f_Q(x))$. In the standard simulation-based security [GVW12], Sim can take $(f_1(x), \ldots, f_Q(x))$, $(f_1, \ldots, f_Q)$, and $(\mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q})$. We consider this variant as a relaxed definition.

Definition 3.4 (Relaxed-SIM Security against a Semi-Malicious Authority). We define relaxed-SIM security against a semi-malicious authority exactly as above except that the simulator SIM takes as input $(\mathsf{pk}, 1^{|x|}, \mathcal{V})$ where $\mathcal{V} = \left\{f_i(x), f_i, \mathsf{sk}_{f_i}\right\}_{i \in [Q]}$.

This relaxed definition is also meaningful since Sim does not use information about x beyond $\{f_i(x)\}_{i \in [Q]}$ (and $|x|$).

Definition 3.5 (SIM Security against Malicious Authority). For a sPCE scheme, an adversary $\mathcal{A}$ and a PPT simulator Sim we define the experiment for security against malicious authority $\mathsf{Expt}^{\mathsf{MS}}_{\beta,\mathcal{A}}(1^\lambda)$ as follows.

1. $\mathcal{A}$ outputs a public key pk and an input x.

2. The challenger samples a random bit $\beta \leftarrow \{0,1\}$. If $\beta = 0$, it computes $\mathsf{ct}_0 \leftarrow \mathsf{Enc}(\mathsf{pk}, x)$ else if $\beta = 1$, it computes $\mathsf{ct}_1 \leftarrow \mathsf{Sim}(\mathsf{pk}, 1^{|x|}, f_1(x), \ldots, f_Q(x))$, where $(f_1, \ldots, f_Q) \leftarrow \mathrm{Ext}(1^\lambda, \mathsf{pk})$. It sends ct_β to $\mathcal{A}$. Here Ext is an extractor algorithm.

3. $\mathcal{A}$ outputs a guess bit β' as the output of the experiment.

We define the advantage $\mathsf{Adv}^{\mathsf{MS}}_{\mathcal{A}}(\lambda)$ of $\mathcal{A}$ in the above game as

$$\mathsf{Adv}^{\mathsf{MS}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{Expt}^{\mathsf{MS}}_{0,\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{MS}}_{1,\mathcal{A}}(1^\lambda) = 1\right]\right|.$$

We say that a sPCE scheme satisfies security against a malicious authority if there exists a PPT simulator Sim and an *admissible* (possibly inefficient) extractor Ext such that for every PPT adversary $\mathcal{A}$, we have $\mathsf{Adv}^{\mathsf{MS}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$. We say that Ext is admissible if for every $(f_1, \ldots, f_Q)$ and randomness r, we have that $(f_1, \ldots, f_Q) = (f_1', \ldots, f_Q')$, where (i) $(\mathsf{pk}, \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q}) \leftarrow \mathsf{Setup}(1^\lambda, \{f_1, \ldots, f_Q\}; r)$ and (ii) $(f_1', \ldots, f_Q') \leftarrow \mathrm{Ext}(1^\lambda, \mathsf{pk})$.

Definition 3.6 (Unconditional SIM Security against Malicious Authority). We say that a sPCE scheme satisfies unconditional simulation security against a malicious authority if there exists a PPT simulator Sim such that for *any* (unbounded) adversary $\mathcal{A}$, the advantage of $\mathsf{Adv}^{\mathsf{MS}}_{\mathcal{A}}(\lambda)$ of $\mathcal{A}$ (as defined in Definition 3.5) is negligible in the security parameter.

Definition 3.7 (Security against Outsiders). A sPCE scheme for function family $\mathcal{F}$ is said to satisfy security against outsiders if for any PPT adversary $\mathcal{A}$, any $x_0, x_1 \in \mathcal{X}$, $\{f_1, \ldots, f_Q\} \in \mathcal{F}$, the following holds

$$\Pr\left[\ \mathcal{A}(\mathsf{pk}, \mathsf{ct}_\beta) = \beta : \begin{array}{l} (\mathsf{pk}, \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q}) \leftarrow \mathsf{Setup}(1^\lambda, \{f_1, \ldots, f_Q\}); \\ \beta \leftarrow \{0,1\}; \mathsf{ct}_\beta \leftarrow \mathsf{Enc}(\mathsf{pk}, x_\beta) \end{array}\ \right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Lemma 3.8. If a sPCE scheme satisfies function-hiding and is secure against a (semi-malicious/malicious) authority, then it is also secure against outsiders.

Proof. Recall that in the security against outsiders we prove that $\mathsf{Enc}(\mathsf{pk}, x_0) \approx_c \mathsf{Enc}(\mathsf{pk}, x_1)$ where $\mathsf{pk} \leftarrow \mathsf{Setup}(1^\lambda, f_1, \ldots, f_Q)$ for any functions $(f_1, \ldots, f_Q)$ and inputs $x_0, x_1 \in \{0,1\}^L$. To prove this, we define the first hybrid game as follows. We change functions f_i to f_0 for all i where $f_0(x) = 0$ for any $x \in \{0,1\}^L$. By function-hiding property of the sPCE scheme, the first hybrid game is indistinguishable from the original game where x_0 is encrypted. Next we define the second hybrid game as follows. We encrypt x_1 instead of x_0. By the simulation security against the malicious authority, it holds that $\mathsf{Enc}(\mathsf{pk}, x_0) \approx_c \mathsf{Sim}(\mathsf{pk}, 1^{|x_0|}, \{f_0(x_0), \ldots, f_0(x_0)\})$. Since $|x_0| = |x_1|$ and $f_0(x_0) = f_0(x_1) = 0$, we have $\mathsf{Sim}(\mathsf{pk}, 1^{|x_0|}, \{f_0(x_0), \ldots, f_0(x_0)\}) = \mathsf{Sim}(\mathsf{pk}, 1^{|x_1|}, \{f_0(x_1), \ldots, f_0(x_1)\})$. Again using the simulation security against the authority, we have $\mathsf{Sim}(\mathsf{pk}, 1^{|x_1|}, \{f_0(x_1), \ldots, f_0(x_1)\}) \approx_c \mathsf{Enc}(\mathsf{pk}, x_1)$. The second hybrid game is indistinguishable from the original game where x_1 is encrypted due to the function-hiding property. Thus, we obtain the lemma. □

Definition 3.9 (Laconic sPCE). A laconic sPCE scheme is the same as a sPCE scheme with an additional property that the size of the public key is sublinear in the number of functions Q input to setup as well as the (maximum) length s of any function, that is $|\mathsf{pk}| = O(Q^{1-\gamma} s^{1-\varepsilon})$ for some $0 < \gamma, \varepsilon < 1$.

On non-trivial sPCE. We observe that the notion of sPCE is trivially achievable if both of the following properties hold.

1. The public key size $|\mathsf{pk}|$ is linear in Q (i.e., non-succinct public keys).
2. The public key pk reveals the information about $(f_1, \ldots, f_Q)$ (i.e., non-function-hiding).

We can easily achieve non-function-hiding sPCE that does not have succinct public keys. The public key consists of $(f_1, \ldots, f_Q)$, and the encryptor computes and sends $(f_1(x), \ldots, f_Q(x))$ as the ciphertext. The notion is primarily meaningful when the public key hides the functions, or (ideally) is sublinear in Q.

3.2 sPCE from SSP-OT and Garbled Circuits

We construct a single-key static pre constrained encryption scheme $\mathsf{sPCE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ for function family $\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y}\}$. We consider the boolean representation of the functions in $\mathcal{F}$ using an ℓ bit string.

Building Blocks. We use the following ingredients for our construction.

1. A garbling scheme $\mathsf{GC} = (\mathsf{Garble}, \mathsf{GCEval})$ for universal circuit $U[x]$, with $x \in \mathcal{M}$ hardwired, that takes as input a function from function family $\mathcal{F}$. This can be instantiated from Yao's scheme [Yao82], which can be based on any one way function.
2. A two-message SSP-OT scheme $\mathsf{OT} = (\mathsf{OTR}, \mathsf{OTS}, \mathsf{OTD})$ with input space as the space of labels of the above garbled circuit scheme. This can be instantiated from a wide variety of assumptions as discussed in Section 2.8.

Construction. We describe our construction below.

$\mathsf{Setup}(1^\lambda, f) \to (\mathsf{pk}, \mathsf{sk})$. The setup algorithm does the following.

- Parse f as an ℓ bit string and let $f[i]$ denote the i-th bit of f. For $i \in [\ell]$, compute $(\mathsf{ot}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{OTR}(1^\lambda, f[i])$.
- Output $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$ and $\mathsf{sk}_f = \{f[i], \mathsf{st}_i\}_{i\in[\ell]}$.

$\mathsf{Enc}(\mathsf{pk}, x) \to \mathsf{ct}$. The encryption algorithm does the following.

- Define the circuit $U[x]$, with x hardwired, as follows : On input a function f, $U[x](f) = f(x)$.
- Compute $(\tilde{U}, \{\mathsf{lb}_{i,b}\}) \leftarrow \mathsf{Garble}(1^\lambda, U[x])$ for $i \in [\ell], b \in \{0,1\}$.
- Parse $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$ and compute $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{i,0}, \mathsf{lb}_{i,1}, \mathsf{ot}_{1,i})$ for all $i \in [\ell]$.
- Output $\mathsf{ct} = (\tilde{U}, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]})$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to \{m/\bot\}$. The decryption algorithm does the following.

- Parse $\mathsf{sk} = \{f[i], \mathsf{st}_i\}_{i\in[\ell]}$ and $\mathsf{ct} = (\tilde{U}, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]})$.
- For each $i \in [\ell]$, compute $\mathsf{lb}_{i,j} \leftarrow \mathsf{OTD}(1^\lambda, f[i], \mathsf{st}_i, \mathsf{ot}_{2,i})$ where $j \in \{0,1\}$.
- Compute and output $y \leftarrow \mathsf{GCEval}(\tilde{U}, \{\mathsf{lb}_{i,j}\}_{i\in[\ell]})$.

Correctness. We show that the above construction satisfies correctness via the following theorem.

Theorem 3.10. Suppose the OT scheme and the GC scheme satisfy correctness as defined in Definition 2.23 and Definition 2.13, respectively. Then the above construction satisfies *perfect* correctness as defined in Definition 3.1.

Proof. We note that for any $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, x)$, we have $\mathsf{ct} = (\tilde{U}, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]})$, where $(\tilde{U}, \{\mathsf{lb}_{i,b}\}) \leftarrow \mathsf{Garble}(1^\lambda, U[x])$ and $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{i,0}, \mathsf{lb}_{i,1}, \mathsf{ot}_{1,i})$ for $i \in [\ell]$. By the correctness OT scheme we have for all $i \in [\ell]$, $\mathsf{lb}_{i,f[i]} = \mathsf{OTD}(1^\lambda, f[i], \mathsf{st}_i, \mathsf{ot}_{2,i})$ with probability 1. Also, from the correctness of the GC scheme it follows that $f(x) = U[x](f) \leftarrow \mathsf{GCEval}(\tilde{U}, \{\mathsf{lb}_{i,f[i]}\}_{i\in[\ell]})$ with probability 1.
So we get $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = f(x)$ with probability 1. Hence the above scheme is perfectly correct. □

Function-hiding. This follows directly from the receiver privacy of the underlying OT scheme.

Theorem 3.11. Suppose the OT scheme satisfies receiver privacy (Definition 2.24). Then the above construction of the sPCE scheme satisfies function-hiding (Definition 3.2).

Proof. Recall that to show function-hiding, we want

$$\{\mathsf{pk} \mid (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda, f_0)\} \approx_c \{\mathsf{pk} \mid (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda, f_1)\}$$

for any functions $f_0, f_1 \in \mathcal{F}$. The proof proceeds via the following sequence of hybrid games between the challenger and a PPT adversary $\mathcal{A}$.

Hyb_0. This is the real world with bit $\beta = 0$, i.e., the challenge public key is computed using the function f_0. We write the complete game to set up the notations and easy reference in the later hybrids.

1. $\mathcal{A}$ outputs two functions $f_0, f_1 \in \mathcal{F}$ where $|f_0| = |f_1|$.
2. The challenger computes $(\mathsf{ot}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{OTR}(1^\lambda, f_0[i])$ for all $i \in [\ell]$ where $C[i]$ denotes the i-th bit of C. It returns $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}$ to $\mathcal{A}$.
3. In the end $\mathcal{A}$ outputs a bit β'.

$\mathsf{Hyb}_{k;1\leq k\leq \ell}$. This hybrid is same as the previous hybrid except that the challenger computes $(\mathsf{ot}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{OTR}(1^\lambda, f_1[i])$ for $1 \leq i \leq k$ and $(\mathsf{ot}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{OTR}(1^\lambda, f_0[i])$ for $k+1 \leq i \leq \ell$, where $f_b[i]$ denotes the i-th bit of the function f_b for $b \in \{0,1\}$.

Note that Hyb_ℓ is the real world with bit $\beta = 1$, i.e., the challenge public key is computed using the function f_1.

We note that it is sufficient to argue $\mathsf{Hyb}_{k-1} \approx_c \mathsf{Hyb}_k$, $k \in [\ell]$, to complete the proof. We show that if there exists a PPT adversary $\mathcal{A}$ who can distinguish between the two hybrids with non-negligible advantage ϵ, then there exists a PPT adversary $\mathcal{B}$ against the receiver privacy of the OT scheme with the same advantage ϵ. The reduction is as follows.

1. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs the challenge functions f_0, f_1 such that $f_0, f_1 \in \mathcal{F}$.
2. $\mathcal{B}$ parses f_0 and f_1 as ℓ bit strings and forwards $f_0[k]$ and $f_1[k]$ to the OT challenger, where $f_b[k]$ denotes the k-th bit of the function f_b for $b \in \{0,1\}$. The OT challenger samples a bit $\beta \leftarrow \{0,1\}$ and computes $(\mathsf{ot}_{1,k}, \mathsf{st}_{1,k}) \leftarrow \mathsf{OTR}(1^\lambda, f_\beta[k])$ and returns $\mathsf{ot}_{k,1}$ to $\mathcal{B}$.
3. $\mathcal{B}$ computes $(\mathsf{ot}_{1,i}, \mathsf{st}_{1,i}) \leftarrow \mathsf{OTR}(1^\lambda, f_1[i])$ for $1 \leq i \leq k-1$ and $(\mathsf{ot}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{OTR}(1^\lambda, f_0[i])$ for $k+1 \leq i \leq \ell$, sets $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$ and forwards it to $\mathcal{A}$.
4. In the end $\mathcal{A}$ outputs a bit β'. $\mathcal{B}$ forwards β' to the OT challenger.

We observe that if the OT challenger samples $\beta = 0$, then $\mathcal{B}$ simulated the distribution Hyb_{k-1} , else Hyb_k with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1|\beta = 0) - \Pr(\beta' = 1|\beta = 1)|$ = $|\Pr(\beta' = 1|\mathsf{Hyb}_{k-1}) - \Pr(\beta' = 1|\mathsf{Hyb}_k)|$ = ϵ (by assumption). □

SIM **security against malicious authority.** This follows from the statistical sender security of the underlying OT scheme and the simulation security of GC scheme.

Theorem 3.12. Suppose the OT scheme satisfies statistical sender-privacy (Definition 2.25) and the GC scheme satisfies simulation security (Definition 2.14). Then the above construction of the sPCE scheme satisfies SIM security against a malicious authority (Definition 3.5).

Proof. To prove the theorem, we first construct the simulator Sim for the security against malicious authority of the sPCE scheme. Note that the simulator is given $\mathsf{pk}, 1^{|x|}, f(x)$, where $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$ as input. We now provide the description of the simulator Sim.

On input $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}_{i\in[\ell]}, 1^{|x|}, f(x)$

1. Compute $(\tilde{U}, \{\mathsf{lb}_i\}_{i\in[\ell]}) \leftarrow \mathsf{GC.Sim}(1^\lambda, f(x))$.
2. Compute $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_i, \mathsf{lb}_i, \mathsf{ot}_{1,i})$ for all $i \in [\ell]$.
3. Output $\mathsf{ct} = (\tilde{U}, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]})$.

To prove the security, we consider the following sequence of hybrids.

Hyb_0. This is the real world, i.e., challenge ct is computed by honestly running the Enc algorithm. We write the complete game here to set up the notations and easy reference in later hybrids.

1. $\mathcal{A}$ outputs a public key pk and an input x.
2. The challenger defines the circuit $U[x]$ as in the construction and computes $(\tilde{U}, \{\mathsf{lb}_{i,b}\}_{i\in[\ell],b\in\{0,1\}}) \leftarrow \mathsf{GC.Garble}(1^\lambda, U[x])$. It parses $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$ and computes $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{i,0}, \mathsf{lb}_{i,1}, \mathsf{ot}_{1,i})$ for all $i \in [\ell]$. It sets $\mathsf{ct} = (\tilde{U}, \{\mathsf{ot}_{2,i}\})$ and returns ct to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a guess bit β'.

Hyb_1. This hybrid is same as the previous hybrid except that the challenger computes $\mathsf{ot}_{2,i}$ differently, i.e., $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{i,b}, \mathsf{lb}_{i,b}, \mathsf{ot}_{1,i})$ for all $i \in [\ell]$, where $b \leftarrow \mathsf{OT.Ext}(\mathsf{ot}_{1,i})$.

Hyb_2. This hybrid is same as the previous hybrid except that the challenger computes the garbled circuit and the labels differently using GC.Sim, i.e., $(\tilde{U}, \{\mathsf{lb}_i\}_{i\in[\ell]}) \leftarrow \mathsf{GC.Sim}(1^\lambda, f(x))$ where $f = f[1]\ldots f[\ell]; f[i] \leftarrow \mathsf{OT.Ext}(\mathsf{ot}_{1,i})$ for $i \in [\ell]$. This is the ideal world.

Indistinguishability of hybrids. We now prove that the consecutive hybrids are indistinguishable.

Claim 3.13. Assume that OT satisfies statistical sender privacy, then $\mathsf{Hyb}_0 \approx_s \mathsf{Hyb}_1$.

Proof. To prove the claim we consider sub hybrids $\mathsf{Hyb}_{0.k}$ for $k = 0$ to ℓ, where $\mathsf{Hyb}_{0.k}$ is same as Hyb_0 except that $\mathsf{ot}_{2,i}$ is generated differently for all $i \leq k$, i.e., $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{i,b}, \mathsf{lb}_{i,b}, \mathsf{ot}_{1,i})$, where $b = \mathsf{OT.Ext}(\mathsf{ot}_{1,i})$ for $1 \leq i \leq k$ and $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{i,0}, \mathsf{lb}_{i,1}, \mathsf{ot}_{1,i})$ for $k+1 \leq i \leq \ell$. We note that $\mathsf{Hyb}_0 = \mathsf{Hyb}_{0.0}$ and $\mathsf{Hyb}_{0.\ell} = \mathsf{Hyb}_1$.
To prove the above claim it suffices to show that $\mathsf{Hyb}_{0.k-1} \approx \mathsf{Hyb}_{0.k}$ for $k \in [\ell]$. We show that if there exists an unbounded adversary $\mathcal{A}$ who can distinguish between $\mathsf{Hyb}_{0.k-1}$ and $\mathsf{Hyb}_{0.k}$ with non-negligible advantage ϵ, then there exists an unbounded adversary $\mathcal{B}$ against the statistical sender privacy security of OT scheme with the same advantage ϵ. The reduction is as follows.

1. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs a public key pk and input x.

2. $\mathcal{B}$ parses $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$, defines the circuit $U[x]$ as in the construction and computes $(\tilde{U}, \{\mathsf{lb}_{i,b}\}) \leftarrow \mathsf{GC.Garble}(1^\lambda, U[x])$ for all $i \in [\ell]$ and $b \in \{0,1\}$.

3. $\mathcal{B}$ sends $(\mathsf{lb}_{k,0}, \mathsf{lb}_{k,1}), \mathsf{ot}_{1,k}$ to the OT challenger. The challenger samples a bit $\beta \leftarrow \{0,1\}$ and computes $\mathsf{ot}_{2,k} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{k,0}, \mathsf{lb}_{k,1}, \mathsf{ot}_{1,k})$ if $\beta = 0$, else if $\beta = 1$, it computes $\mathsf{ot}_{2,k} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{k,b}, \mathsf{lb}_{k,b}, \mathsf{ot}_{1,k})$ where $b = \mathsf{OT.Ext}(\mathsf{ot}_{1,k})$. It returns $\mathsf{ot}_{2,k}$ to $\mathcal{B}$.

4. $\mathcal{B}$ computes $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{i,b}, \mathsf{lb}_{i,b}, \mathsf{ot}_{1,i})$, where $b = \mathsf{OT.Ext}(\mathsf{ot}_{1,i})$ for $1 \leq i \leq k-1$ and $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(1^\lambda, \mathsf{lb}_{i,0}, \mathsf{lb}_{i,1}, \mathsf{ot}_{1,i})$ for $k+1 \leq i \leq \ell$. It sets $\mathsf{ct} = (\tilde{U}, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]})$ and returns ct to $\mathcal{A}$.

5. $\mathcal{A}$ outputs a guess bit β'. $\mathcal{B}$ forwards β' to the OT challenger.

We observe that if the OT challenger samples $\beta = 0$, then $\mathcal{B}$ simulated the distribution $\mathsf{Hyb}_{0.k-1}$, else $\mathsf{Hyb}_{0.k}$ with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1|\beta = 0) - \Pr(\beta' = 1|\beta = 1)|$ = $|\Pr(\beta' = 1|\mathsf{Hyb}_{0.k-1}) - \Pr(\beta' = 1|\mathsf{Hyb}_{0.k})|$ = ϵ (by assumption). □

Claim 3.14. Assume that GC satisfies simulation security, then $\mathsf{Hyb}_1 \approx_c \mathsf{Hyb}_2$.

Proof. We show that if there exists a non-uniform PPT $\mathcal{A}$ who can distinguish between Hyb_1 and Hyb_2 with non-negligible advantage ϵ, then there exists a non-uniform PPT adversary $\mathcal{B}$ against the security of GC scheme with the same advantage ϵ. The reduction is as follows.

1. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs a public key pk and input x.

2. $\mathcal{B}$ parses $\mathsf{pk} = \{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$ and computes $f = f[1]\ldots f[\ell]$ where $f[i] = \mathsf{OT.Ext}(\mathsf{ot}_{1,i})$ for $i \in [\ell]$.

3. $\mathcal{B}$ defines the circuit $U[x]$ as in the construction and sends $U[x]$ and f to the GC challenger as the challenge circuit and challenge input. The challenger chooses a bit $\beta \leftarrow \{0,1\}$ and does the following:

   - If $\beta = 0$, it computes $(\tilde{U}, \{\mathsf{lb}_{i,b}\}_{i\in[\ell],b\in\{0,1\}}) \leftarrow \mathsf{GC.Garble}(1^\lambda, U[x])$. It sets $U' = \tilde{U}$ and $\mathsf{lb}_i = \{\mathsf{lb}_{i,f[i]}\}$ for $i \in [\ell]$.
   - If $\beta = 1$, it computes $(\tilde{U}, \{\mathsf{lb}_i\}_{i\in[\ell]}) \leftarrow \mathsf{GC.Sim}(1^\lambda, U[x](f))$. It sets $U' = \tilde{U}$.

   The GC challenger returns $(U', \{\mathsf{lb}_i\}_{i\in[\ell]})$ to $\mathcal{B}$.

4. $\mathcal{B}$ computes $\mathsf{ot}_{2,i} \leftarrow \mathsf{OTS}(\mathsf{lb}_i, \mathsf{lb}_i, \mathsf{ot}_{1,i})$ for all $i \in [\ell]$. It returns $\mathsf{ct} = (U', \{\mathsf{ot}_{2,i}\}_{i\in[\ell]})$.

5. In the end, $\mathcal{A}$ outputs a bit β'. $\mathcal{B}$ sends β' to the GC challenger.

We observe that if the GC challenger samples $\beta = 0$, then $\mathcal{B}$ simulated Hyb_1, else Hyb_2 with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1|\beta = 0) - \Pr(\beta' = 1|\beta = 1)|$ = $|\Pr(\beta' = 1|\mathsf{Hyb}_1) - \Pr(\beta' = 1|\mathsf{Hyb}_2)|$ = ϵ (by assumption). □

□

Security against outsiders. This follows from Lemma 3.8.

Unconditional security for NC^1. We note that by using an information theoretic version of Yao's garbled circuit [IK02], efficient for NC^1, in the above construction we can achieve a sPCE scheme, for class NC^1, with unconditional security against a malicious authority.

Bounded key setting. We observe that it is easy to extend the construction above to bounded multi-key construction by simply preparing more OT instances for f_i. Note that this makes $|\mathsf{pk}|$ linear in the number of functions for which we generate the secret keys.

Instantiating the underlying two-message SSP-OT as discussed in Section 2.8, we get the following.

Theorem 3.15. Assuming DDH / (QR and DCR) / LWE / (LPN and Nisan-Wigderson style derandomization), there exists a sPCE scheme, for general circuits, satisfying security against a malicious authority (Definition 3.5).

Theorem 3.16. Assuming DDH / (QR and DCR) / LWE / (LPN and Nisan-Wigderson style derandomization), there exists a sPCE scheme, for NC^1 , satisfying unconditional security against a malicious authority (Definition 3.6).

3.3 sPCE with Unconditional Security from FHE

In this section we see that by using a slightly stronger assumption we can achieve unconditional security against a malicious authority. We construct a single-key sPCE scheme for function family $\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y}\}$. We consider the boolean representation of the functions in $\mathcal{F}$ using an ℓ bit string.

Building block. We use a FHE scheme $\mathsf{FHE} = \mathsf{FHE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ satisfying malicious circuit privacy. This can be instantiated from LWE (Theorem 2.31).

Construction. We describe our sPCE construction below.

$\mathsf{Setup}(1^\lambda, f) \to (\mathsf{pk}, \mathsf{sk})$. The setup algorithm does the following.

- Generate $(\mathsf{FHE.pk}, \mathsf{FHE.sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$. Parse f as an ℓ bit string and compute $\mathsf{FHE.ct}_f \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE.pk}, f)$.
- Output $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct}_f)$ and $\mathsf{sk} = \mathsf{FHE.sk}$.

$\mathsf{Enc}(\mathsf{pk}, x) \to \mathsf{ct}$. The encryption algorithm does the following.

- Define the circuit $G[x]$, with x hardwired, as follows: On input a function f, $G[x](f) = f(x)$.
- Parse $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct}_f)$ and compute $\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Eval}(\mathsf{FHE.pk}, G[x], \mathsf{FHE.ct}_f)$.
- Output $\mathsf{ct} = \mathsf{FHE.ct}$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to y$. The decryption algorithm parses $\mathsf{sk} = \mathsf{FHE.sk}$, $\mathsf{ct} = \mathsf{FHE.ct}$ and returns $\mathsf{FHE.Dec}(\mathsf{FHE.sk}, \mathsf{FHE.ct})$.

Succinct Ciphertexts. We note that due to the sub-linearity of the underlying FHE scheme, our scheme achieves succinct ciphertexts.

Theorem 3.17. The construction of sPCE from maliciously circuit private FHE is correct and satisfies function-hiding, unconditional security against a malicious authority and security against outsiders.

Proof. The correctness and function-hiding of the construction follows immediately from the correctness and semantic-security of the underlying FHE scheme, respectively.
We prove the security against the authority. First, we construct a simulator Sim for the security against malicious authority of the sPCE scheme. Note that the simulator is given $\mathsf{pk}, 1^{|x|}, f(x)$, where $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct}_f)$ as inputs. The simulator proceeds as follows:

1. Runs the FHE simulator to compute $\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Sim}(\mathsf{FHE.pk}, \mathsf{FHE.ct}_f, f(x))$.
2. Output $\mathsf{ct} = \mathsf{FHE.ct}$.

To prove the security, we consider the following sequence of hybrids.

Hyb_0. This is the real world, i.e., challenge ct is computed by honestly running the Enc algorithm, using the possibly malformed public key pk.

Hyb_1. This hybrid is same as the previous hybrid except that the challenger computes $\mathsf{FHE.ct}$ as

$$\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Sim}(\mathsf{FHE.pk}, \mathsf{FHE.ct}_f, G[x](f))$$

where $f = \mathsf{FHE.Ext}(\mathsf{FHE.pk}, \mathsf{FHE.ct}_f)$. We note that this is the ideal world.

The indistinguishability of the above two hybrids follows from the malicious circuit private security of the underlying FHE scheme. Recall that maliciously circuit-private FHE ensures that even for possibly malformed $\mathsf{FHE.pk}, \mathsf{FHE.ct}$, $\mathsf{FHE.Eval}(\mathsf{FHE.pk}, G[x], \mathsf{FHE.ct}_f)$ is statistically indistinguishable from
$\mathsf{FHE.Sim}(G[x](f)) = \mathsf{FHE.Sim}(f(x))$.
Also, the security against outsiders follows from Lemma 3.8. □

Instantiating the underlying malicious circuit private FHE as in Theorem 2.31, we get the following.

Theorem 3.18. Assuming LWE, there exists a sPCE scheme, for general circuits, satisfying unconditional security against a malicious authority (Definition 3.6).

3.4 Implications and Lower Bounds

We present lower bounds of sPCE in this section. These lower bounds hold even for a weaker IND style security definition provided below.

Definition 3.19 (IND Security against Malicious Authority). A sPCE scheme is said to satisfy indistinguishability against a malicious authority if for any PPT and admissible adversary $\mathcal{A}$, the following holds

$$\Pr\left[\ \mathcal{A}(\mathsf{ct}_\beta) = \beta : \begin{array}{l} \mathsf{pk}, (x_0, x_1) \leftarrow \mathcal{A}; \\ \beta \leftarrow \{0,1\}; \mathsf{ct}_\beta \leftarrow \mathsf{Enc}(\mathsf{pk}, x_\beta) \end{array}\ \right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

$\mathcal{A}$ is admissible if (i) $f_1, \ldots, f_Q \in \mathcal{F}$ where $(\{f_1, \ldots, f_Q\}) \leftarrow \mathrm{Ext}(1^\lambda, \mathsf{pk})$, where Ext is an extractor algorithm and (ii) $f_i(x_0) = f_i(x_1)$ for all $i \in [Q]$.

We observe that for an indistinguishability-based security definition, if $|\mathsf{ct}|$ is sublinear in Q or s (we call it weakly CT-collusion-succinct and weakly CT-succinct, respectively, in this paper), sPCE implies IO by known results (via [LPST16b]). In addition, this holds even for security against *semi-honest authority* since the transformation from single-key weakly succinct PKFE to IO needs the indistinguishability-based security for standard FE. Thus, we can say achieving sPCE with succinct ciphertexts is as hard as achieving IO. We formalise it using the following lemma.

Lemma 3.20. If there exists a sPCE scheme that supports all polynomial size circuits, satisfies indistinguishability against semi-honest authority and the succinct ciphertexts property (i.e., $|\mathsf{ct}|$ is $O(Q^{1-\gamma})$ or $O(s^{1-\epsilon})$ of for some $0 < \gamma, \epsilon < 1$), there exists IO for all polynomial size circuits.

Proof. To prove the lemma, it suffices to show that a sPCE scheme with succinct ciphertext satisfying IND security implies IO. We observe that

1. We can construct single-key weakly CT-succinct PKFE from Q-key weakly CT-collusion-succinct sPCE via the transformation by [KNT21, Section 4] (almost the same as [BV18, Section 4.2]). This transformation works since weakly selective security for standard FE (both target plaintexts and functions are fixed at the beginning of the game) is sufficient for our purpose. Note that the syntax of single-key PKFE by Bitansky and Vaikuntanathan [BV18] is the same as single-key sPCE. Although the transformation by Kitagawa et al. or Bitansky and Vaikuntanathan is for weakly succinct schemes (i.e., the *encryption circuit* is weakly succinct), it works for weakly CT-succinct schemes if the goal is weakly CT-succinct.)

2. We can construct single-key weakly succinct PKFE from single-key weakly CT-succinct and PKFE and the LWE assumption via the transformation by [LPST16a]. This implies IO [BV18]. Or we can use output compressing randomized encoding in the CRS model in [LPST16b]. We can construct weakly sublinear compact randomized encoding scheme for Turing machines in the CRS model from single-key weakly CT-succinct and PKFE (note that [LPST16b] calls weakly CT-succinct as weakly sublinear compact). □

We also show that it is impossible to achieve sPCE with succinct public key which is secure against a malicious authority, using a natural incompressibility style argument inspired from [AGVW13]. Moreover, in our setting, the impossibility holds even for the indistinguishability-based definition of sPCE. We prove it using the following theorem.

Theorem 3.21. There does not exist a sPCE scheme that supports all polynomial size circuits, satisfies indistinguishability against malicious authority, and whose public key size is $O(Q^{1-\gamma})$ or $O(s^{1-\epsilon})$ for some $0 < \gamma, \epsilon < 1$ where Q is the number of the functions used in the setup algorithm and s is the maximum length of the functions.

Proof. First, we focus on the case $|\mathsf{pk}| = O(Q^{1-\gamma})$. We consider the following constant function $C[\rho]$ where a uniformly random string $\rho \leftarrow \{0,1\}^\lambda$ is hardwired. The function $C[\rho]$ takes $\mathbf{x} \in \{0,1\}^{\lambda/2}$ as as input, and outputs ρ. Let $\mathcal{F}_c := \{C[\rho] : \{0,1\}^{\lambda/2} \to \{0,1\}^\lambda\}$. Assume that there exists a sPCE scheme that is indistinguishable against malicious authority and has succinct public keys for all polynomial size circuits. Then, there also exists such a sPCE scheme that supports the function family $\mathcal{F}_c$ above. This implies that there exists an adversary $\mathcal{A}$ that outputs a public key pk and two inputs $(\mathbf{x}_0, \mathbf{x}_1) \in (\{0,1\}^{\lambda/2})^2$. There also exists a (possibly inefficient) extractor Ext that takes pk and outputs $(C[\rho_1], \ldots, C[\rho_Q])$. Note that it holds that $C[\rho_i](\mathbf{x}_0) = C[\rho_i](\mathbf{x}_1)$ for all $i \in [Q]$. The extractor can also compute $\rho_i = C[\rho_i](\mathbf{x}')$ for all $i \in [Q]$ where $\mathbf{x}'$ is an arbitrary input. Here, $|\mathsf{pk}| = O(Q^{1-\gamma})$ holds due the succinct public key property. That is, $\mathcal{A}$ compressed a uniformly random $Q \times \lambda$ bit string into an $O(Q^{1-\gamma})$ bit string since the extractor can recover a uniformly random $Q \times \lambda$ bit string from pk, which is an $O(Q^{1-\gamma})$ bit string. $O(Q^{1-\gamma})$ is asymptotically smaller than $Q \times \lambda$. This compression is information theoretically impossible since the Kolmogorov complexity of a uniformly random $Q \times \lambda$ string is at least $Q \times \lambda$ by the definition.

In the case $|\mathsf{pk}| = O(s^{1-\epsilon})$, a similar argument works. We consider only one function $C[\rho_1]$. Then, $|\mathsf{pk}| = O(\lambda^{1-\epsilon})$ holds since $s \geq |\rho_1| = \lambda$. However, the extractor can recover a uniformly random λ bit string from pk. This compression is also impossible as above.

Thus, the proof is concluded. □

3.5 Laconic sPCE for General Constraints

In this section we provide a two-step construction for laconic sPCE, as follows.

1. First we provide a sPCE construction, with laconic public key, that does not achieve function-hiding (and consequently security against outsiders).

2. Next, we show how to compile the above sPCE into a function-hiding sPCE.

We consider semi-malicious security to achieve laconic public keys due to the impossibility result in Section 3.4.

3.5.1 Construction without Function-Hiding.

Building blocks. We use the following ingredients for our construction.

1. A reusable, dynamic MPC protocol $\mathsf{RDMPC} = (\mathsf{CktEnc}, \mathsf{InpEnc}, \mathsf{Local}, \mathsf{Decode})$ with $N = \Theta(R^2\lambda)$, $t = \Theta(R\lambda)$, and and $n = \Theta(t)$. This can be instantiated from one way functions (Theorem 2.6). We use $\widehat{\ell}$ to denote the size of the output of InpEnc algorithm. We also denote the size of the circuit $\mathsf{Local}(\widehat{C}_j, \cdot)$ by $\widehat{s}(\lambda, |C|)$, where $\widehat{C}_j$ is the output of CktEnc on input a circuit C.

2. A garbled circuit scheme $\mathsf{GC} = (\mathsf{GC.Garble}, \mathsf{GC.Eval})$ for circuit class $\mathcal{C}_{\widehat{\ell}}$, where $\mathcal{C}_{\widehat{\ell}}$ is the set of all polynomial size circuits with input length $\widehat{\ell}$. [7]

3. A hash encryption scheme $\mathsf{HE} = (\mathsf{HE.Gen}, \mathsf{HE.Hash}, \mathsf{HE.Enc}, \mathsf{HE.Dec})$ for hash domain $\{-1, 0, 1\}^{\overline{\ell}}$ satisfying succinctness and semi-malicious security. This can be instantiated using LWE in ROM (Appendix B.1).

<u>Construction.</u> We describe the construction of the laconic sPCE scheme below.

$\mathsf{Setup}(1^\lambda, \{f_1, \ldots, f_Q\})$. The setup algorithm does the following.

1. For each f_i, where $i \in [Q]$, do the following.
   (a) Parse f_i as an ℓ bit string.
   (b) Compute $\hat{f}_i \leftarrow \mathsf{InpEnc}(1^\lambda, 1^\ell, f_i)$ and let $|\hat{f}_i| = \widehat{\ell}$.
2. For each $i \in [Q]$, sample random set $\Delta^{(i)} \subset [N]$ such that $|\Delta^{(i)}| = n$.
   If $\left|\bigcup_{i,i' \in [Q], i \neq i'} \left(\Delta^{(i)} \cap \Delta^{(i')}\right)\right| > t$, abort.
3. For $i \in [Q]$, let $S_i = \{(i, j, k, \hat{f}_i[k])\}_{j \in \Delta^{(i)}, k \in [\widehat{\ell}]}$, where $\hat{f}_i[k]$ is the k-th bit of $\hat{f}_i$. Set $S := \|_{i \in [Q]} S_i$.
4. Let str be a binary string of length with $\overline{\ell} = Q \cdot N \cdot \widehat{\ell}$, indexed as (i, j, k), for $i \in [Q], j \in [N]$ and $k \in [\widehat{\ell}]$, where
$$\mathsf{str}_{i,j,k} = \begin{cases} 1, & \text{if } (i, j, k, 1) \in S \\ 0, & \text{if } (i, j, k, 0) \in S \\ -1, & \text{if } j \notin \cup_{i \in [Q]} \Delta^{(i)} \end{cases}$$
5. Generate $\mathsf{key} \leftarrow \mathsf{HE.Gen}(1^\lambda, \overline{\ell})$ and compute $h_{\mathsf{str}} \leftarrow \mathsf{HE.Hash}(\mathsf{key}, \mathsf{str})$.
6. Output $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$ and for $i \in [Q]$,
$$\mathsf{sk}_{f_i} = \left(i, \Delta^{(i)}, \hat{f}_i\right). \tag{3}$$

$\mathsf{Enc}(\mathsf{pk}, x, 1^Q)$. On input the public key $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$, an input x and the query bound $1 \leq Q \leq 2^\lambda$ in unary form, do the following.

1. Define circuit C_x, which on input a function f, outputs $f(x)$.
2. Compute $(\widehat{C}_{i,1}, \ldots, \widehat{C}_{i,N}) \leftarrow \mathsf{CktEnc}(1^\lambda, 1^\lambda, 1^\ell, C_x)$ for $i \in [Q]$.
3. Define the circuit $\mathsf{L}_{i,j}(\cdot) := \mathsf{Local}(\widehat{C}_{i,j}, \cdot)$ with input length $\widehat{\ell}$.
   For all $i \in [Q]$ and $j \in [N]$, do the following.
   (a) Run the garbling algorithm $\left(\left\{\mathsf{lab}_{i,j,k,b}\right\}_{k \in [\widehat{\ell}], b \in \{0,1\}}\right) \leftarrow \mathsf{GC.Garble}(1^\lambda, \mathsf{L}_{i,j})$.
   (b) For all $k \in [\widehat{\ell}]$ and $b \in \{0, 1\}$, compute [8]
$$\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i, j, k), b), \mathsf{lab}_{i,j,k,b}).$$
4. Output
$$\mathsf{ct} = \left\{\mathsf{HE.ct}_{i,j,k,b}\right\}_{i \in [Q], j \in [N], k \in [\widehat{\ell}], b \in \{0,1\}}. \tag{4}$$

[7] For the ease of notations, we will use the syntax of the garbling scheme by [AMVY21], where the output of the Garble algorithm and the input to the Eval algorithm consists of only labels. This can be shown equivalent to the standard syntax (Section 2.4), by including the garbled circuit into a label.

[8] Note that we do not generate any ciphertext for $b = -1$, as it is not required for the functionality of our scheme. We want to recover the labels $\mathsf{lab}_{i,j,k,b}$ when $\mathsf{str}_{i,j,k} = b$ using the secret key as generated in the setup phase and we set $\mathsf{str}_{i,j,k} = -1$ only when $j \notin \cup_{i \in [Q]} \Delta^{(i)}$, i.e., for those $j \in [N]$ which is not a part of any of the secret keys. So, we do not need $\mathsf{HE.ct}$ corresponding to $b = -1$.

$\mathsf{Dec}(\mathsf{ct}, \mathsf{sk}, 1^Q)$. On input a secret key sk, a ciphertext ct, and the query bound $1 \le Q \le 2^\lambda$ in unary form, do the following.

1. Parse the secret key as Eq. (3) and the ciphertext as Eq. (4).
2. For all $j \in \Delta^{(i)}$, do the following.
   (a) Compute $\mathsf{lab}'_{i,j,k} := \mathsf{HE.Dec}(\mathsf{key}, (h_{\mathsf{str}}, i, j, k, \hat{f}_i[k]), \mathsf{HE.ct}_{i,j,k,\hat{f}_i[k]})$, for all $k \in [\widehat{\ell}]$, where $\hat{f}_i[k]$ is the k-th bit of $\hat{f}_i$.
   (b) Compute $\widehat{y}'_{i,j} := \mathsf{GC.Eval}(\{\mathsf{lab}'_{i,j,k}\}_{k\in[\widehat{\ell}]})$.
3. Compute and output $z_i = \mathsf{Decode}(\{\widehat{y}'_{i,j}\}_{j\in\Delta^{(i)}}, \Delta^{(i)})$.

Laconic Public Key. We note that in the above construction $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$, where $\mathsf{key} \leftarrow \mathsf{HE.Gen}(1^\lambda, \overline{\ell})$ and $h_{\mathsf{str}} \leftarrow \mathsf{HE.Hash}(\mathsf{key}, \mathsf{str})$. For the laconic public keys we instantiate the underlying HE scheme as described in Appendix B.1, where $|\mathsf{key}| = \lambda$ and $|h_{\mathsf{str}}| = \lambda$. So it follows that $|\mathsf{pk}| = 2\lambda$, which is independent of the number of functions(Q) in the setup phase and the size of the function ℓ. Hence the above construction achieves full succinctness.

Theorem 3.22 (Correctness). Suppose the HE scheme, GC and RDMPC scheme satisfies correctness as defined in Definition 2.7, 2.13 and 2.4 respectively. Then the above construction of sPCE is correct.

Proof. By the correctness of HE scheme, we have $\mathsf{lab}_{i,j,k,\hat{f}[k]} = \mathsf{HE.Dec}(\mathsf{key}, (\mathsf{Hash}(\mathsf{key}, \mathsf{str}), i, j, k, \hat{f}[k]), \mathsf{HE.ct}_{i,j,k,\hat{f}[k]})$ with all but negligible probability, since $\mathsf{str}_{i,j,k} = \hat{f}[k]$ by the definition of str. So, $\mathsf{lab}'_{i,j,k} = \mathsf{lab}_{i,j,k,\hat{f}[k]}$ holds for all $\mathsf{lab}'_{i,j,k}$ recovered in Step 2a of the decryption algorithm. Next, by the correctness of GC scheme, we have $\mathsf{L}_{i,j}(\hat{f}_i) = \mathsf{GC.Eval}(\widetilde{L}_{i,j}, \{\mathsf{lab}_{i,j,k,\hat{f}[k]}\}_{k\in[\widehat{\ell}]})$. So, $\widehat{y}'_{i,j} = \mathsf{L}_{i,j}(\hat{f}_i) = \mathsf{Local}(\widehat{C}_{i,j}, \hat{f}_i)$ for all $\widehat{y}'_{i,j}$ recovered in Step 2b of the decryption algorithm.

Finally, since $|\Delta^{(i)}| = n$ we have $C_x(f_i) = \mathsf{Decode}(\{\mathsf{Local}(\widehat{C}_{i,j}, \hat{f}_i)\}_{j\in\Delta^{(i)}}, \Delta^{(i)})$ by the correctness of the RDMPC scheme. So, $z_i = C_x(f_i) = f_i(x)$ for z_i recovered in Step 3 of the decryption algorithm. □

3.5.2 Security

Here, we show that our construction satisfies relaxed-SIM security against a semi-malicious authority.

Theorem 3.23. Assume that HE scheme is secure against a semi malicious setup , GC is a secure garbled circuit scheme, and RDMPC is secure as per Definition 2.8, 2.14, and 2.5 respectively. Then the above construction of laconic sPCE scheme satisfies security against a semi malicious authority Definition 3.4.

Proof. To prove the theorem, we first construct the simulator Sim for the security against malicious authority of the sPCE scheme. Note that the simulator is given $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$, $1^{|x|}$, and $\mathcal{V}$, where $\mathcal{V} = \left\{ f_i(x), f_i, \mathsf{sk}_{f_i} = \left(i, \Delta^{(i)}, \hat{f}_i\right) \right\}_{i\in[Q]}$. We begin with setting up few notations. We define sets S_{crr} and S_{dis} as follows.

$$S_{\mathsf{crr}} := \bigcup_{i,i'\in[Q], i\neq i'} \left(\Delta^{(i)} \cap \Delta^{(i')}\right), \quad S_{\mathsf{dis}} := \left(\bigcup_{i\in[Q]} \Delta^{(i)}\right) \setminus S_{\mathsf{crr}}.$$

We now describe the sPCE ciphertext simulator SIM. On input $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$, $1^{|x|}$, and $\mathcal{V}$, it runs as follows.

1. For each $i \in [Q]$, run

$$\left(\left\{\widehat{C}_{i,j}\right\}_{j\in S_{\mathsf{crr}}}, \{\widehat{y}_{i,j}\}_{i\in[Q], j\in\Delta^{(i)}}\right) \leftarrow \mathsf{RDMPC.Sim}_0\left(1^{|C_x|}, S_{\mathsf{crr}}, \{f_i(x), f_i\}_{i\in[Q]}\right)$$

where C_x is the universal circuit with x hardwired.

2. For all $i \in [Q]$ and $j \in [N]$, do the following.

   - If $j \in S_{\mathsf{crr}}$, do the following.

     (a) Set $\mathsf{L}_{i,j}(\cdot) := \mathsf{Local}(\widehat{C}_{i,j}, \cdot)$ and compute $\left(\widetilde{L}_{i,j}, \left\{\mathsf{lab}_{i,j,k,b}\right\}_{k\in[\widehat{\ell}],b\in\{0,1\}}\right) \leftarrow \mathsf{GC.Garble}(1^\lambda, \mathsf{L}_{i,j})$.

     (b) For all $k \in [\widehat{\ell}]$ and $b \in \{0,1\}$, compute $\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), \mathsf{lab}_{i,j,k,b})$.

   - If $j \in S_{\mathsf{dis}}$, then there exists an unique i such that $j \in \Delta^{(i)}$. Retrieve the corresponding $i \in [Q]$ and do the following.

     (a) Compute $\left(\widetilde{L}_{i,j}, \{\mathsf{lab}_{i,j,k}\}_{k\in[\widehat{\ell}]}\right) \leftarrow \mathsf{GC.Sim}(1^\lambda, \widehat{y}_{i,j})$.

     (b) For all $k \in [\widehat{\ell}]$ and $b \in \{0,1\}$, compute $\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), \mathsf{lab}_{i,j,k})$.

   - If $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$, do the following.

     (a) Set $L_{i,j}$ to be a dummy circuit of input length $\widehat{\ell}$, which always outputs $0^{\widehat{s}}$ and compute $\left(\widetilde{L}_{i,j}, \left\{\mathsf{lab}_{i,j,k,b}\right\}_{k\in[\widehat{\ell}],b\in\{0,1\}}\right) \leftarrow$ $\mathsf{GC.Garble}(1^\lambda, \mathsf{L}_{i,j})$.

     (b) For all $k \in [\widehat{\ell}]$ and $b \in \{0,1\}$, compute $\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), 0^{\mathsf{len}})$ where len is the length of the labels.

3. Output the ciphertext $\mathsf{ct} = \left(\left\{\widetilde{L}_{i,j}\right\}_{i\in[Q],j\in[N]}, \left\{\mathsf{HE.ct}_{i,j,k,b}\right\}_{i\in[Q],j\in[N],k\in[\widehat{\ell}],b\in\{0,1\}}\right)$.

To prove the security, we consider following hybrids.

Hyb_0. This is the real world. We write the complete game here to set up notations and for easy reference in the later hybrids.

1. The adversary outputs the functions $f_1, \ldots, f_Q$ and the randomness $\Delta^{(1)}, \ldots, \Delta^{(Q)}, r$. Abort and output $\perp$ if either of the following is true.

   - $\left|\Delta^{(i)}\right| \neq n$, for $i \in [Q]$.
   - $|S_{\mathsf{crr}}| = \left|\bigcup_{i,i'\in[Q],i\neq i'} \left(\Delta^{(i)} \cap \Delta^{(i')}\right)\right| > t$.

2. The challenger computes $S_i = \{(i,j,k,\hat{f}_i[k])\}_{j\in\Delta^{(i)},k\in[\widehat{\ell}]}$, and defines S and string str as in the construction. It lets the $\mathsf{key} = r$ and computes $h_{\mathsf{str}} \leftarrow \mathsf{Hash}(\mathsf{key}, \mathsf{str})$. It sets $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$ and $\mathsf{sk}_{f_i} = \left(i, \Delta^{(i)}, \hat{f}_i\right)$ and returns pk and $\{\mathsf{sk}_{f_i}\}_{i\in[Q]}$ to $\mathcal{A}$.

3. $\mathcal{A}$ outputs the challenge input x. The challenger defines C_x and computes

$$\mathsf{ct} = \left(\left\{\widetilde{L}_{i,j}\right\}_{i\in[Q],j\in[N]}, \left\{\mathsf{HE.ct}_{i,j,k,b}\right\}_{i\in[Q],j\in[N],k\in[\widehat{\ell}],b\in\{0,1\}}\right)$$

   as in the construction. It returns ct to $\mathcal{A}$.

4. In the end, $\mathcal{A}$ outputs a guess bit β'.

Hyb_1. In this hybrid we change the way ciphertext is generated. In particular, we change the way $\mathsf{HE.ct}_{i,j,k,b}$ is computed in the following cases.

- If $j \in S_{\mathsf{dis}}$, the challenger computes $\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), \mathsf{lab}_{i,j,k,\hat{f}_i[k]})$, where i is the unique index such that $j \in \Delta^{(i)}$.
- If $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$, the challenger computes $\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), 0^{\mathsf{len}})$, where len is the length of the labels.

Hyb_2. In this hybrid, we further change the way ciphertext is generated. In particular, we change the way $\widetilde{L}_{i,j}$ and $\mathsf{lab}_{i,j,k,b}$ is computed in the following cases.

- If $j \in S_{\mathsf{dis}}$, then there exists an unique i such that $j \in \Delta^{(i)}$. Retrieve the corresponding $i \in [Q]$ and compute $(\widetilde{L}_{i,j}, \{\mathsf{lab}_{i,j,k}\}_{k \in [\widehat{\ell}]}) \leftarrow \mathsf{GC.Sim}(1^\lambda, \widehat{y}_{i,j})$ where $\widehat{y}_{i,j} = \mathsf{Local}(\widehat{C}_{i,j}, \hat{f}_i)$. Then it sets $\mathsf{lab}_{i,j,k,\hat{f}_i[k]} := \mathsf{lab}_{i,j,k}$ for $k \in [\widehat{\ell}]$.
- If $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$, the challenger sets $L_{i,j}$ to be a dummy circuit of input length $\widehat{\ell}$, which always outputs $0^{\widehat{s}}$ and computes $\left(\widetilde{L}_{i,j}, \left\{\mathsf{lab}_{i,j,k,b}\right\}_{k \in [\widehat{\ell}], b \in \{0,1\}}\right) \leftarrow \mathsf{GC.Garble}(1^\lambda, \mathsf{L}_{i,j})$.

Hyb_3. In this hybrid, we further change the way ciphertext is generated. In particular, we change the way the circuit encoding $\widehat{C}_{i,j}$ and local output encodings $\widehat{y}_{i,j}$ are generated as follows. For each $i \in [Q]$, run

$$\left(\left\{\widehat{C}_{i,j}\right\}_{j \in S_{\mathsf{crr}}}, \left\{\widehat{y}_{i,j}\right\}_{i \in [Q], j \in \Delta^{(i)}}\right) \leftarrow \mathsf{RDMPC.Sim}_0\left(1^{|C_x|}, S_{\mathsf{crr}}, \{f_i(x), f_i\}_{i \in [Q]}\right)$$

where C_x is the universal circuit with x hardwired. We note that we only have $\widehat{C}_{i,j}$ for all $j \in S_{\mathsf{crr}}$ and this suffices for the ciphertext generation due to the changes made in Hyb_0 and Hyb_1.
We also note that this is the ideal world with the simulator SIM defined above.

Next, we show the indistinguishability of consecutive hybrids for Theorem 3.23.

Claim 3.24. Assume HE is a semi malicious secure hash encryption scheme, then $\mathsf{Hyb}_0 \approx_c \mathsf{Hyb}_1$.

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ who can distinguish between Hyb_0 and Hyb_1 with non-negligible advantage ϵ, then there exists an adversary $\mathcal{B}$ against the semi malicious secure hash encryption scheme with the same advantage ϵ. The reduction is as follows.

1. The HE challenger samples a bit $\beta \leftarrow \{0,1\}$ and initiates the multi challenge HE security game with $\mathcal{B}$.
2. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs the functions $f_1, \ldots, f_Q$ and the randomness $\Delta^{(1)}, \ldots, \Delta^{(Q)}, r$. $\mathcal{B}$ aborts and output $\perp$ if $\left|\Delta^{(i)}\right| \neq n$ or $|S_{\mathsf{crr}}| > t$.
3. $\mathcal{B}$ computes $\hat{f}_i$ for $i \in [Q]$, defines the set S and the string str as in the construction. It sends the string str and the randomness r to the HE challenger and gets back the hash key key.
4. $\mathcal{B}$ computes $h_{\mathsf{str}} = \mathsf{Hash}(\mathsf{key}, \mathsf{str})$. It sets $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$ and $\mathsf{sk}_{f_i} = \left(i, \Delta^{(i)}, \hat{f}_i\right)$ and returns pk and $\{\mathsf{sk}_{f_i}\}_{i \in [Q]}$ to $\mathcal{A}$.
5. $\mathcal{A}$ outputs the challenge input x. $\mathcal{B}$ computes $\left(\widetilde{L}_{i,j}, \left\{\mathsf{lab}_{i,j,k,b}\right\}_{k \in [\widehat{\ell}], b \in \{0,1\}}\right)$ for all $i \in [Q]$ and $j \in [N]$ as in the construction.
6. $\mathcal{B}$ computes $\left\{\mathsf{HE.ct}_{i,j,k,b}\right\}_{i \in [Q], j \in [N], k \in [\widehat{\ell}], b \in \{0,1\}}$ as follows
   - If $j \in S_{\mathsf{dis}}$ and $b \neq \hat{f}_i[k]$, it first retrieves i such that $j \in \Delta^{(i)}$ and then sends the index (i, j, k) and two messages $(\mathsf{lab}_{i,j,k,1-\hat{f}_i[k]}, \mathsf{lab}_{i,j,k,\hat{f}_i[k]})$ to the HE challenger. The challenger computes

$$\mathsf{HE.ct}_{i,j,k,b} \leftarrow \begin{cases} \mathsf{HE.Enc}(\mathsf{key}, (\mathsf{Hash}(\mathsf{key}, \mathsf{str}), (i,j,k), b), \mathsf{lab}_{i,j,k,1-\hat{f}_i[k]}) & \text{if } \beta = 0 \\ \mathsf{HE.Enc}(\mathsf{key}, (\mathsf{Hash}(\mathsf{key}, \mathsf{str}), (i,j,k), b), \mathsf{lab}_{i,j,k,\hat{f}_i[k]}) & \text{if } \beta = 1 \end{cases}$$

   and returns $\mathsf{HE.ct}_{i,j,k,b}$ to $\mathcal{B}$.

- If $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$, for each $b \in \{0,1\}$, it sends the index (i,j,k) and two messages $(\mathsf{lab}_{i,j,k,b}, 0^{\mathsf{len}})$ to the HE challenger, where len is the length of the labels. The challenger computes

$$\mathsf{HE.ct}_{i,j,k,b} \leftarrow \begin{cases} \mathsf{HE.Enc}(\mathsf{key}, (\mathsf{Hash}(\mathsf{key}, \mathsf{str}), (i,j,k), b), \mathsf{lab}_{i,j,k,b}) & \text{if } \beta = 0 \\ \mathsf{HE.Enc}(\mathsf{key}, (\mathsf{Hash}(\mathsf{key}, \mathsf{str}), (i,j,k), b), 0^{\mathsf{len}}) & \text{if } \beta = 1 \end{cases}$$

  and returns $\mathsf{HE.ct}_{i,j,k,b}$ to $\mathcal{B}$.

- Else, it computes $\mathsf{HE.ct}_{i,j,k,b}$ as in the construction.

7. $\mathcal{B}$ sends $\mathsf{ct} = \left(\left\{ \widetilde{L}_{i,j} \right\}_{i \in [Q], j \in [N]}, \left\{ \mathsf{HE.ct}_{i,j,k,b} \right\}_{i \in [Q], j \in [N], k \in [\widehat{\ell}], b \in \{0,1\}} \right)$ to $\mathcal{A}$.

8. $\mathcal{A}$ outputs a guess bit β'.

We observe that if the HE challenger samples $\beta = 0$, then $\mathcal{B}$ simulated the distribution Hyb_0, else Hyb_1 with $\mathcal{A}$. Hence, the advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1 | \beta = 0) - \Pr(\beta' = 1 | \beta = 1)|$ = $|\Pr(\beta' = 1 | \mathsf{Hyb}_0) - \Pr(\beta' = 1 | \mathsf{Hyb}_1)|$ = ϵ (by assumption). □

Claim 3.25. Assume GC is a secure garbled circuit scheme, then $\mathsf{Hyb}_1 \approx_c \mathsf{Hyb}_2$.

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ who can distinguish between Hyb_1 and Hyb_2 with non-negligible advantage ϵ, then there exists an adversary $\mathcal{B}$ against the security of GC scheme with the same advantage ϵ. The reduction is as follows.

1. The GC challenger samples a bit $\beta \leftarrow \{0,1\}$ and initiates the multi challenge GC security game with $\mathcal{B}$.

2. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs the functions $f_1, \ldots, f_Q$ and the randomness $\Delta^{(1)}, \ldots, \Delta^{(Q)}, r$. $\mathcal{B}$ aborts and output $\perp$ if $\left|\Delta^{(i)}\right| \neq n$ or $|S_{\mathsf{crr}}| > t$.

3. $\mathcal{B}$ computes $\hat{f}_i$ for $i \in [Q]$, defines the set S and the string str as in the construction. $\mathcal{B}$ generates $\mathsf{key} \leftarrow \mathsf{HE.Gen}(1^\lambda, r)$ and computes $h_{\mathsf{str}} = \mathsf{Hash}(\mathsf{key}, \mathsf{str})$. It sets $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$ and $\mathsf{sk}_{f_i} = \left(i, \Delta^{(i)}, \hat{f}_i\right)$ and returns pk and $\{\mathsf{sk}_{f_i}\}_{i \in [Q]}$ to $\mathcal{A}$.

4. $\mathcal{A}$ outputs the challenge input x. $\mathcal{B}$ defines the circuit C_x as in the construction and computes $(\widehat{C}_{i,1}, \ldots, \widehat{C}_{i,N}) \leftarrow \mathsf{CktEnc}(1^\lambda, 1^\lambda, 1^\ell, C_x)$ for $i \in [Q]$.

5. For all $i \in [Q]$ and $j \in [N]$, $\mathcal{B}$ does the following.

- If $j \in S_{\mathsf{crr}}$, set $\mathsf{L}_{i,j}(\cdot) := \mathsf{Local}(\widehat{C}_{i,j}, \cdot)$ and compute

$$\left(\widetilde{L}_{i,j}, \left\{ \mathsf{lab}_{i,j,k,b} \right\}_{k \in [\widehat{\ell}], b \in \{0,1\}} \right) \leftarrow \mathsf{GC.Garble}(1^\lambda, \mathsf{L}_{i,j}).$$

- If $j \in S_{\mathsf{dis}}$, it first retrieves i such that $j \in \Delta^{(i)}$ and then sends the circuit $\mathsf{L}_{i,j}(\cdot) := \mathsf{Local}(\widehat{C}_{i,j}, \cdot)$ and input $\hat{f}_i$ to the GC challenger. The challenger computes

$$\left(\widetilde{L}_{i,j}, \left\{ \mathsf{lab}_{i,j,k,b} \right\}_{k \in [\widehat{\ell}], b \in \{0,1\}} \right) \leftarrow \mathsf{GC.Garble}(1^\lambda, \mathsf{L}_{i,j}) \text{ if } \beta = 0$$

  or

$$\left(\widetilde{L}_{i,j}, \left\{ \mathsf{lab}_{i,j,k} \right\}_{k \in [\widehat{\ell}]} \right) \leftarrow \mathsf{GC.Sim}(1^\lambda, \mathsf{Local}(\widehat{C}_{i,j}, \hat{f}_i)) \text{ if } \beta = 1$$

  and returns $\left(\widetilde{L}_{i,j}, \left\{ \mathsf{lab}_{i,j,k,\hat{f}_i[k]} \right\}_{k \in [\widehat{\ell}]} \right)$ to $\mathcal{B}$, where $\mathsf{lab}_{i,j,k,\hat{f}_i[k]} = \mathsf{lab}_{i,j,k}$ for $\beta = 1$.

   - If $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$, set $L_{i,j}$ to be a dummy circuit of input length $\widehat{\ell}$, which always outputs $0^{\widehat{s}}$ and compute

$$\left(\widetilde{L}_{i,j}, \left\{\mathsf{lab}_{i,j,k,b}\right\}_{k\in[\widehat{\ell}],b\in\{0,1\}}\right) \leftarrow \mathsf{GC.Garble}(1^{\lambda}, \mathsf{L}_{i,j}).$$

6. For all $i \in [Q]$ and $j \in [N]$, $\mathcal{B}$ does the following.
   - If $j \in S_{\mathsf{crr}}$, it computes $\{\mathsf{HE.ct}_{i,j,k,b}\}_{k\in[\widehat{\ell}],b\in\{0,1\}}$ as in the construction.
   - If $j \in S_{\mathsf{dis}}$, it computes $\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), \mathsf{lab}_{i,j,k,\hat{f}_i[k]})$ for all $k \in [\widehat{\ell}]$ and $b \in \{0,1\}$.
   - If $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$, then for all $k \in [\widehat{\ell}]$ and $b \in \{0,1\}$, it computes $\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), 0^{\mathsf{len}})$, where len is the length of the labels.
7. $\mathcal{B}$ sends $\mathsf{ct} = \left(\left\{\widetilde{L}_{i,j}\right\}_{i\in[Q],j\in[N]}, \left\{\mathsf{HE.ct}_{i,j,k,b}\right\}_{i\in[Q],j\in[N],k\in[\widehat{\ell}],b\in\{0,1\}}\right)$ to $\mathcal{A}$.
8. $\mathcal{A}$ outputs a guess bit β'.

First, we note that for $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$, the way $\mathcal{B}$ generates $\left(\widetilde{L}_{i,j}, \left\{\mathsf{lab}_{i,j,k,b}\right\}_{k\in[\widehat{\ell}],b\in\{0,1\}}\right)$ does not effect the adversary's view because of the change we introduced in Hyb_1. In particular, we do not use the labels $\mathsf{lab}_{i,j,k,b}$ generated for $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$ anywhere in Hyb_1 or Hyb_2 and hence the adversary does not have sufficient information to compute on the garbled circuit $\widetilde{L}_{i,j}$ for the corresponding set of labels. Next, we observe that if the GC challenger samples $\beta = 0$, then $\mathcal{B}$ simulated the distribution Hyb_1 , else Hyb_2 with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1|\beta = 0) - \Pr(\beta' = 1|\beta = 1)|$ = $|\Pr(\beta' = 1|\mathsf{Hyb}_1) - \Pr(\beta' = 1|\mathsf{Hyb}_2)|$ = ϵ (by assumption). □

Claim 3.26. Assume RDMPC is a secure reusable dynamic MPC scheme, then $\mathsf{Hyb}_2 \approx_c \mathsf{Hyb}_3$.

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ who can distinguish between Hyb_2 and Hyb_3 with non-negligible advantage ϵ, then there exists an adversary $\mathcal{B}$ against the security of RDMPC scheme with the same advantage ϵ. The reduction is as follows.

1. The RDMPC challenger samples a bit $\beta \leftarrow \{0,1\}$ and initiates the multi challenge RDMPC security game with $\mathcal{B}$.
2. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs the functions $f_1, \ldots, f_Q$ and the randomness $\Delta^{(1)}, \ldots, \Delta^{(Q)}, r$. $\mathcal{B}$ aborts and output $\perp$ if $\left|\Delta^{(i)}\right| \neq n$ or $|S_{\mathsf{crr}}| > t$.
3. $\mathcal{B}$ sends the query bound 1^Q, input length 1^{ℓ}, where $|f_i| = \ell$ to the challenger. Note that this defines the total number of parties $N = (\lambda, Q)$, number of parties $n = n(\lambda, Q)$ participating in any session, threshold $t = t(\lambda, Q)$. It also sends $S_{\mathsf{crr}}, \Delta^{(1)}, \ldots, \Delta^{(Q)}$ to the challenger.
4. $\mathcal{B}$ sends the functions $f_1, \ldots, f_Q$ as the input encoding query to the RDMPC challenger. The challenger computes and returns $\hat{f}_i \leftarrow \mathsf{InpEnc}(1^{\lambda}, 1^{\ell}, f_i)$ for each f_i, where $i \in [Q]$, to $\mathcal{B}$.
5. $\mathcal{B}$ generates $\mathsf{key} \leftarrow \mathsf{HE.Gen}(1^{\lambda}, \overline{\ell})$ and computes $h_{\mathsf{str}} \leftarrow \mathsf{HE.Hash}(\mathsf{key}, \mathsf{str})$, where the string str is as defined in the construction. It sends $\mathsf{pk} = (\mathsf{key}, h_{\mathsf{str}})$ and $\mathsf{sk}_{f_i} = \left(i, \Delta^{(i)}, \hat{f}_i\right)$, for $i \in [Q]$, to the adversary $\mathcal{A}$.
6. $\mathcal{A}$ outputs the challenge input x. $\mathcal{B}$ does the following.
   (a) It defines the circuit C_x as in the construction and sends the circuit C_x as the challenge for all the input encoding queries to the RDMPC challenger. For each $i \in [Q]$, the challenger does the following:

- If $\beta = 0$, it computes $(\widehat{C}_{i,1}, \ldots, \widehat{C}_{i,N}) \leftarrow \mathsf{CktEnc}(1^\lambda, 1^\lambda, 1^\ell, C_x)$ for each $i \in [Q]$ and $y_{i,j} := \mathsf{Local}(\widehat{C}_{i,j}, \hat{f}_i)$ for all $i \in [Q]$ and $j \in \Delta^{(i)}$.
- If $\beta = 1$, it computes

$$\left(\left\{ \widehat{C}_{i,j} \right\}_{j \in S_{\mathsf{crr}}}, \{\widehat{y}_{i,j}\}_{i \in [Q], j \in \Delta^{(i)}} \right) \leftarrow \mathsf{RDMPC.Sim}_0 \left(1^{|C_x|}, S_{\mathsf{crr}}, \{f_i(x), f_i\}_{i \in [Q]} \right)$$

- It returns $\left(\left\{ \widehat{C}_{i,j} \right\}_{j \in S_{\mathsf{crr}}}, \{\widehat{y}_{i,j}\}_{i \in [Q], j \in \Delta^{(i)}} \right)$ to $\mathcal{B}$.

(b) For all $i \in [Q]$ and $j \in [N]$,

- if $j \in S_{\mathsf{crr}}$, sets $\mathsf{L}_{i,j}(\cdot) := \mathsf{Local}(\widehat{C}_{i,j}, \cdot)$ and computes $(\widetilde{L}_{i,j}, \{\mathsf{lab}_{i,j,k,b}\}_{k \in [\widehat{\ell}], b \in \{0,1\}})$ and $\{\mathsf{HE.ct}_{i,j,k,b}\}_{k \in \widehat{\ell}, b \in \{0,1\}}$ honestly as in the construction.
- if $j \in S_{\mathsf{dis}}$, then there exists an unique i such that $j \in \Delta^{(i)}$. Retrieve the corresponding $i \in [Q]$ and computes $\left(\widetilde{L}_{i,j}, \left\{ \mathsf{lab}_{i,j,k} \right\}_{k \in [\widehat{\ell}]} \right) \leftarrow \mathsf{GC.Sim}(1^\lambda, \widehat{y}_{i,j})$. Then for all $k \in [\widehat{\ell}]$ and $b \in \{0,1\}$, it computes

$$\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), \mathsf{lab}_{i,j,k}).$$

- if $j \notin S_{\mathsf{crr}} \cup S_{\mathsf{dis}}$, set $L_{i,j}$ to be a dummy circuit of input length $\widehat{\ell}$, which always outputs $0^{\widehat{s}}$ and compute

$$\left(\widetilde{L}_{i,j}, \left\{ \mathsf{lab}_{i,j,k,b} \right\}_{k \in [\widehat{\ell}], b \in \{0,1\}} \right) \leftarrow \mathsf{GC.Garble}(1^\lambda, \mathsf{L}_{i,j}).$$

Then for all $k \in [\widehat{\ell}]$ and $b \in \{0,1\}$, it computes

$$\mathsf{HE.ct}_{i,j,k,b} \leftarrow \mathsf{HE.Enc}(\mathsf{key}, (h_{\mathsf{str}}, (i,j,k), b), 0^{\mathsf{len}})$$

where len is the length of the labels.

(c) $\mathcal{B}$ sends $\mathsf{ct} = \left(\left\{ \widetilde{L}_{i,j} \right\}_{i \in [Q], j \in [N]}, \left\{ \mathsf{HE.ct}_{i,j,k,b} \right\}_{i \in [Q], j \in [N], k \in [\widehat{\ell}], b \in \{0,1\}} \right)$ to $\mathcal{A}$.

7. $\mathcal{A}$ outputs a guess bit β'.

We observe that if the RDMPC challenger samples $\beta = 0$, then $\mathcal{B}$ simulated the distribution Hyb_2 , else Hyb_3 with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1 | \beta = 0) - \Pr(\beta' = 1 | \beta = 1)|$ = $|\Pr(\beta' = 1 | \mathsf{Hyb}_2) - \Pr(\beta' = 1 | \mathsf{Hyb}_3)|$ = ϵ (by assumption). □

□

3.6 Laconic sPCE with Function-Hiding

In this section we provide a compiler to convert a semi-malicious sPCE without function-hiding and security against outsiders into a laconic sPCE with function-hiding and security against outsiders using a maliciously circuit-private FHE with linear efficiency. We also show that the resulting scheme retains the succinctness and security against a semi-malicious authority property of the underlying sPCE scheme.

Building Blocks. We use the following ingredients for our construction.

1. A sPCE scheme $\mathsf{SFE} = (\mathsf{SFE.Setup}, \mathsf{SFE.Enc}, \mathsf{SFE.Dec})$ satisfying (SIM/relaxed-SIM) security against a semi-malicious authority with succinct public keys.

2. A compact maliciously circuit-private FHE scheme with linear efficiency $\mathsf{FHE} = (\mathsf{FHE.KeyGen}, \mathsf{FHE.Enc}, \mathsf{FHE.Dec})$.

Construction. Below we provide the description of our compiler.

$\mathsf{Setup}(1^\lambda, f_1, \ldots, f_Q)$. The setup algorithm does the following.

1. Generate $(\mathsf{FHE.pk}, \mathsf{FHE.sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$ and $(\mathsf{SFE.pk}, \mathsf{SFE.sk}_{f_1}, \ldots, \mathsf{SFE.sk}_{f_Q}) \leftarrow \mathsf{SFE.Setup}(1^\lambda, f_1, \ldots, f_Q)$.
2. Compute $\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE.pk}, \mathsf{SFE.pk})$.
3. Output $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct})$ and $\mathsf{sk}_{f_i} = (\mathsf{FHE.sk}, \mathsf{SFE.sk}_{f_i})$ for $i \in [Q]$.

$\mathsf{Enc}(\mathsf{pk}, x)$. The encryption algorithm does the following.

1. Parse $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct})$.
2. Choose encryption randomness r for $\mathsf{SFE.Enc}$.
3. Let $G[x, r]$ be a circuit that on input key evaluates $\mathsf{SFE.Enc}(\mathsf{key}, x; r)$ (this notation means encryption randomness is r).
4. Compute $\mathsf{FHE.}\widehat{\mathsf{ct}} \leftarrow \mathsf{FHE.Eval}(\mathsf{FHE.pk}, G[x, r], \mathsf{FHE.ct})$.
5. Output $\mathsf{ct} = \mathsf{FHE.}\widehat{\mathsf{ct}}$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$. The decryption algorithm does the following.

1. Parse $\mathsf{sk} = (\mathsf{FHE.sk}, \mathsf{SFE.sk}_{f_i})$ for some $i \in [Q]$ and $\mathsf{ct} = \mathsf{FHE.}\widehat{\mathsf{ct}}$.
2. Compute $y \leftarrow \mathsf{FHE.Dec}(\mathsf{FHE.sk}, \mathsf{FHE.}\widehat{\mathsf{ct}})$.
3. Output $\mathsf{SFE.Dec}(\mathsf{SFE.sk}_{f_i}, y)$.

Succinct public keys. First we note that $|\mathsf{SFE.pk}| = O(Q^{1-\gamma})$ since SFE has succinct public keys. Also, it is easy to see that $|\mathsf{FHE.pk}|$ is independent of Q and $|\mathsf{FHE.ct}| = \mathrm{poly}(\lambda) \cdot |\mathsf{SFE.pk}| = \mathrm{poly}(\lambda) \cdot O(Q^{1-\gamma})$ due to the linear efficiency of FHE. Hence, we have $|\mathsf{pk}| = O(Q^{1-\gamma})$.

Correctness. We show that the above construction is correct via the following theorem.

Theorem 3.27. Assume that the FHE scheme and the SFE scheme satisfies correctness. Then the above construction is correct.

Proof. For any $\mathsf{sk} = (\mathsf{FHE.sk}, \mathsf{SFE.sk}_{f_i})$ and $\mathsf{ct} = \mathsf{FHE.}\widehat{\mathsf{ct}} = \mathsf{FHE.Eval}(\mathsf{FHE.pk}, G[x, r], \mathsf{FHE.ct})$, where $\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE.pk}, \mathsf{SFE.pk})$, we have

$$\mathsf{FHE.Dec}(\mathsf{FHE.sk}, \mathsf{FHE.}\widehat{\mathsf{ct}}) = G[x, r](\mathsf{SFE.pk}) = \mathsf{SFE.Enc}(\mathsf{SFE.pk}, x; r)$$

with probability 1 by the correctness of the underlying FHE scheme. So we have $y = \mathsf{SFE.Enc}(\mathsf{SFE.pk}, x; r)$ in Step 2 of the decryption algorithm. Next, by the correctness of the underlying SFE scheme, we have

$$\mathsf{SFE.Dec}(\mathsf{SFE.sk}_{f_i}, y) = \mathsf{SFE.Dec}(\mathsf{SFE.sk}_{f_i}, \mathsf{SFE.Enc}(\mathsf{SFE.pk}, x)) = f_i(x)$$

with all but negligible probability. Hence, the correctness follows. □

Function-hiding. This follows from the semantic security of underlying FHE scheme. We prove it using the following theorem.

Theorem 3.28. Assume that the FHE scheme satisfies semantic security (Definition 2.27). Then the above construction of the sPCE scheme satisfies function-hiding (Definition 3.2).

Proof. Recall that to show constraint hiding, we want

$$\{\mathsf{pk}|\mathsf{pk} \leftarrow \mathsf{Setup}(1^\lambda, f_1^0, \dots f_Q^0)\} \approx_c \{\mathsf{pk}|\mathsf{pk} \leftarrow \mathsf{Setup}(1^\lambda, f_1^1, \dots f_Q^1)\}$$

where $f_i^b \in \mathcal{F}$ for $i \in [Q]$ and $b \in \{0,1\}$. We show that if there exists a PPT adversary $\mathcal{A}$ who can distinguish between the above two distributions with non-negligible advantage ϵ, then there exists a PPT adversary $\mathcal{B}$ against the semantic security of the FHE scheme with the same advantage ϵ. The reduction is as follows.

1. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs the challenge functions $\{(f_1^0, f_1^1), \dots, (f_Q^0, f_Q^1)\}$.
2. $\mathcal{B}$ computes $(\mathsf{SFE.pk}^b, \mathsf{SFE.sk}^b_{f_1}, \dots, \mathsf{SFE.sk}^b_{f_Q}) \leftarrow \mathsf{SFE.Setup}(1^\lambda, f_1^b, \dots, f_Q^b)$ for $b \in \{0,1\}$.
3. $\mathcal{B}$ sends $(\mathsf{SFE.pk}^0, \mathsf{SFE.pk}^1)$ to the FHE challenger. The challenger generates $(\mathsf{FHE.pk}, \mathsf{FHE.sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$, samples a bit $\beta \leftarrow \{0,1\}$, computes $\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE.pk}, \mathsf{SFE.pk}^\beta)$, and returns $\mathsf{FHE.pk}, \mathsf{FHE.ct}$ to $\mathcal{B}$.
4. $\mathcal{B}$ sets $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct})$ and forwards it to $\mathcal{A}$.
5. In the end $\mathcal{A}$ outputs a bit β'. $\mathcal{B}$ forwards β' to the FHE challenger.

We observe that if the FHE challenger samples $\beta = 0$, then $\mathcal{B}$ simulated the distribution $D_0 = \{\mathsf{pk}|\mathsf{pk} \leftarrow \mathsf{Setup}(1^\lambda, f_1^0, \dots f_Q^0)\}$, else $D_1 = \{\mathsf{pk}|\mathsf{pk} \leftarrow \mathsf{Setup}(1^\lambda, f_1^1, \dots f_Q^1)\}$ with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1|\beta = 0) - \Pr(\beta' = 1|\beta = 1)|$ = $|\Pr(\beta' = 1|D_0) - \Pr(\beta' = 1|D_1)|$ = ϵ (by assumption). □

Security against semi-malicious authority. This follows from the security of the underlying maliciously circuit private FHE and the semi-malicious secure SFE scheme.

Theorem 3.29. Assume that FHE is maliciously circuit-private (Definition 2.28) and SFE is (SIM/relaxed-SIM) secure against a semi malicious authority (Definition 3.4). Then the above construction of sPCE scheme satisfies (SIM/relaxed-SIM) security against a semi malicious authority (Definition 3.4).

Proof. To prove the theorem, we first construct the simulator SIM for the security against the semi-malicious authority of the sPCE scheme. Note that the simulator takes as input $(\mathsf{pk}, 1^{|x|}, \mathcal{V})$ where

$$\mathcal{V}^9 = \left\{ f_i(x), f_i, \mathsf{sk}_{f_i} \right\}_{i \in [Q]}.$$

We now provide the description of the simulator SIM. On input $(\mathsf{pk}, 1^{|x|}, \mathcal{V})$, do the following.

1. Parse $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct})$ and $\mathsf{sk}_{f_i} = (\mathsf{FHE.sk}, \mathsf{SFE.sk}_{f_i})$ for $i \in [Q]$.
2. Let $\mathcal{V}' = \left\{ f_i(x), f_i, \mathsf{SFE.sk}_{f_i} \right\}_{i \in [Q]}$.
3. Compute $\mathsf{SFE.pk} \leftarrow \mathsf{FHE.Dec}(\mathsf{FHE.sk}, \mathsf{FHE.ct})$.
4. Compute $\mathsf{SFE.ct} \leftarrow \mathsf{SFE.Sim}(\mathsf{SFE.pk}, 1^{|x|}, \mathcal{V}')$.
5. Compute $\mathsf{FHE.\widehat{ct}} \leftarrow \mathsf{FHE.Sim}(\mathsf{FHE.pk}, \mathsf{FHE.ct}, \mathsf{SFE.ct})$.
6. Output $\mathsf{ct} = \mathsf{FHE.\widehat{ct}}$.

[9] If the underlying SFE scheme satisfies SIM security against a semi-malicious authority, then we have $\mathcal{V} = \{f_i(x)\}_{i \in [Q]}$

To prove the security, we consider the following sequence of hybrids.

Hyb_0. This is the real world. We write the complete game here to set up notations and for easy reference in the later hybrids.

1. The adversary outputs the functions $f_1, \ldots, f_Q$ and the randomness $\mathsf{FHE}.r_1$, $\mathsf{FHE}.r_2$, and $\mathsf{SFE}.r$ corresponding to the randomness used in the $\mathsf{FHE.KeyGen}$, $\mathsf{FHE.Enc}$ and $\mathsf{SFE.Setup}$ algorithms respectively.
2. The challenger computes $(\mathsf{FHE.pk}, \mathsf{FHE.sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda; \mathsf{FHE}.r_1)$, $(\mathsf{SFE.pk}, \mathsf{SFE.sk}_{f_1}, \ldots, \mathsf{SFE.sk}_{f_Q}) \leftarrow \mathsf{SFE.Setup}(1^\lambda, f_1, \ldots, f_Q; \mathsf{SFE}.r)$, and $\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE.pk}, \mathsf{SFE.pk}; \mathsf{FHE}.r_2)$. It returns $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct})$ and $\mathsf{sk}_{f_i} = (\mathsf{FHE.sk}, \mathsf{SFE.sk}_{f_i})$ for $i \in [Q]$ to the adversary.
3. The adversary outputs the challenge input x. The challenger computes $\mathsf{ct} = \mathsf{FHE}.\widehat{\mathsf{ct}}$ as in the construction and returns ct to the adversary.
4. In the end, the adversary outputs a guess bit β'.

Hyb_1. In this hybrid we change the way ciphertext is generated. In particular, we change the way $\mathsf{FHE}.\widehat{\mathsf{ct}}$ is computed. The challenger computes $\mathsf{FHE}.\widehat{\mathsf{ct}} \leftarrow \mathsf{FHE.Sim}(\mathsf{FHE.pk}, \mathsf{FHE.ct}, \mathsf{SFE.ct})$, where $\mathsf{SFE.ct} \leftarrow \mathsf{SFE.Enc}(\mathsf{pk}, x)$.

Hyb_2. In this hybrid we further change the way ciphertext is generated. In particular, we change the way $\mathsf{SFE.ct}$ is computed. The challenger computes $\mathsf{SFE.ct} \leftarrow \mathsf{SFE.Sim}(\mathsf{SFE.pk}, 1^{|x|}, \mathcal{V}')$, where $\mathcal{V}' = \left\{ f_i(x), f_i, \mathsf{SFE.sk}_{f_i} \right\}_{i \in [Q]}$. We also note that this is the ideal world with the simulator SIM defined above.

Indistinguishability of hybrids. We now show that the consecutive hybrids are indistinguishable.

Claim 3.30. Assume that FHE satisfies malicious circuit privacy, then $\mathsf{Hyb}_0 \approx_s \mathsf{Hyb}_1$.

Proof. We show that if there exists an unbounded adversary $\mathcal{A}$ who can distinguish between Hyb_0 and Hyb_1 with non-negligible advantage ϵ, then there exists an unbounded adversary $\mathcal{B}$ against the malicious circuit privacy security of FHE scheme with the same advantage ϵ. The reduction is as follows.

1. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs the functions $f_1, \ldots, f_Q$ and the randomness $\mathsf{FHE}.r_1$, $\mathsf{FHE}.r_2$, and $\mathsf{SFE}.r$.
2. $\mathcal{B}$ computes $(\mathsf{FHE.pk}, \mathsf{FHE.sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda; \mathsf{FHE}.r_1)$, $(\mathsf{SFE.pk}, \mathsf{SFE.sk}_{f_1}, \ldots, \mathsf{SFE.sk}_{f_Q}) \leftarrow \mathsf{SFE.Setup}(1^\lambda, f_1, \ldots, f_Q; \mathsf{SFE}.r)$, and $\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE.pk}, \mathsf{SFE.pk}; \mathsf{FHE}.r_2)$. It returns $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct})$ and $\mathsf{sk}_{f_i} = (\mathsf{FHE.sk}, \mathsf{SFE.sk}_{f_i})$ for $i \in [Q]$ to $\mathcal{A}$.
3. $\mathcal{A}$ outputs the challenge input x. $\mathcal{B}$ defines the circuit $G[x, r]$ as in the construction and sends $\mathsf{FHE.pk}$, $\mathsf{FHE.ct}$, and $G[x, r]$ to the FHE challenger. The FHE challenger computes $\mathsf{SFE.pk} = \mathsf{FHE.Ext}(\mathsf{FHE.pk}, \mathsf{FHE.ct})$, samples $\beta \leftarrow \{0, 1\}$ and returns $\mathsf{FHE}.\widehat{\mathsf{ct}}_\beta$ to $\mathcal{B}$, where $\mathsf{FHE}.\widehat{\mathsf{ct}}_0 \leftarrow \mathsf{FHE.Eval}(\mathsf{FHE.pk}, G[x, r], \mathsf{FHE.ct})$ and $\mathsf{FHE}.\widehat{\mathsf{ct}}_1 \leftarrow \mathsf{FHE.Sim}(\mathsf{FHE.pk}, \mathsf{FHE.ct}, \mathsf{SFE.Enc}(\mathsf{SFE.pk}, x))$ (since $G[x, r](\mathsf{SFE.pk}) = \mathsf{Enc}(\mathsf{SFE.pk}, x; r)$).
4. $\mathcal{B}$ sets and forwards $\mathsf{ct} = \mathsf{FHE}.\widehat{\mathsf{ct}}_\beta$ to $\mathcal{A}$.
5. In the end, $\mathcal{A}$ outputs a bit β'. $\mathcal{B}$ sends β' to the FHE challenger.

We observe that if the FHE challenger samples $\beta = 0$, then $\mathcal{B}$ simulated Hyb_0, else Hyb_1 with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1 | \beta = 0) - \Pr(\beta' = 1 | \beta = 1)|$ = $|\Pr(\beta' = 1 | \mathsf{Hyb}_0) - \Pr(\beta' = 1 | \mathsf{Hyb}_1)|$ = ϵ (by assumption). □

Claim 3.31. Assume that SFE satisfies relaxed-SIM security against a semi malicious authority. Then $\mathsf{Hyb}_1 \approx_c \mathsf{Hyb}_2$.

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ who can distinguish between Hyb_1 and Hyb_2 with non-negligible advantage ϵ, then there exists a PPT adversary $\mathcal{B}$ against the malicious circuit privacy security of SFE scheme with the same advantage ϵ. The reduction is as follows.

1. $\mathcal{B}$ first runs $\mathcal{A}$. $\mathcal{A}$ outputs the functions $f_1, \ldots, f_Q$ and the randomness $\mathsf{FHE}.r_1$, $\mathsf{FHE}.r_2$, and $\mathsf{SFE}.r$.

2. $\mathcal{B}$ sends the functions $f_1, \dots, f_Q$ and the randomness $\mathsf{SFE}.r$ to the SFE challenger and gets back $(\mathsf{SFE.pk}, \mathsf{SFE.sk}_{f_1}, \dots, \mathsf{SFE.sk}_{f_Q})$.

3. $\mathcal{B}$ generates $(\mathsf{FHE.pk}, \mathsf{FHE.sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda; \mathsf{FHE}.r_1)$, and computes $\mathsf{FHE.ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE.pk}, \mathsf{SFE.pk}; \mathsf{FHE}.r_2)$. It returns $\mathsf{pk} = (\mathsf{FHE.pk}, \mathsf{FHE.ct})$ and $\mathsf{sk}_{f_i} = (\mathsf{FHE.sk}, \mathsf{SFE.sk}_{f_i})$ for $i \in [Q]$ to $\mathcal{A}$.

4. $\mathcal{A}$ outputs the challenge input x. $\mathcal{B}$ forwards x to the SFE challenger as the challenge ciphertext. The SFE challenger samples a bit $\beta \leftarrow \{0,1\}$ and returns $\mathsf{SFE.ct}_\beta$ to $\mathcal{B}$ where $\mathsf{SFE.ct}_0 \leftarrow \mathsf{SFE.Enc}(\mathsf{SFE.pk}, x)$ and $\mathsf{SFE.ct}_1 \leftarrow \mathsf{SFE.Sim}(\mathsf{SFE.pk}, 1^{|x|}, \mathcal{V}')$ where $\mathcal{V}' = \left\{ f_i(x), f_i, \mathsf{SFE.sk}_{f_i} \right\}_{i \in [Q]}$.

5. $\mathcal{B}$ computes $\mathsf{FHE}.\widehat{\mathsf{ct}} \leftarrow \mathsf{FHE.Sim}(\mathsf{FHE.pk}, \mathsf{FHE.ct}, \mathsf{SFE.ct}_\beta)$ and forwards $\mathsf{ct} = \mathsf{FHE}.\widehat{\mathsf{ct}}$ to $\mathcal{A}$.

6. In the end, $\mathcal{A}$ outputs a bit β'. $\mathcal{B}$ sends β' to the SFE challenger.

We observe that if the SFE challenger samples $\beta = 0$, then $\mathcal{B}$ simulated Hyb_1, else Hyb_2 with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1 | \beta = 0) - \Pr(\beta' = 1 | \beta = 1)|$ = $|\Pr(\beta' = 1 | \mathsf{Hyb}_1) - \Pr(\beta' = 1 | \mathsf{Hyb}_2)|$ = ϵ (by assumption). □

□

Security against the outsiders. This follows from Lemma 3.8.

We summarize the result of this section using the following theorem.

Theorem 3.32. There exists a laconic sPCE scheme for general constraints that satisfies security against a semi-malicious authority (Definition 3.4), under the LWE assumption, achieving a succinct master public key with $|\mathsf{mpk}| = \mathrm{poly}(\lambda)$.

4 Laconic Attribute-Based PCE for Point Constraints

In this section we give a construction of an attribute-based pre-constrained encryption scheme for point constraints for function family $\mathcal{F} : \{0,1\}^\ell \to \{0,1\}$ with a succinct master public key. Our construction achieves the definition of AJJM and supports dynamic key generation – please see Section 2.1.

4.1 Construction

We use the following ingredients.

- $\mathrm{Ext}(X, \mathbf{r})$: a seeded strong extractor, where X is a random variable with sufficient min-entropy, and $\mathbf{r} \leftarrow \{0,1\}^\lambda$ is uniformly at random.
- A vector commitment scheme $\mathsf{VC} = (\mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ as defined in Lemma 2.22.
- We let m, n, q denote the LWE parameters such that $n = d^{1/\epsilon} \cdot \mathrm{poly}(\lambda, \log \ell, \log s)$, $m = nd \cdot \mathrm{poly}(\lambda)$, where $0 < \epsilon < 1$ and d is the maximum depth of the functions in $\mathcal{F}$. We set $\sigma' = \sigma^2 n m^2 \lambda \cdot 2^{\omega(\log \lambda)}$, $\sigma_1 = m^{d \cdot \log \lambda}$.

Let the pre-constrained point be $\mathbf{x}^* \in \{0,1\}^\ell$. For simplicity, in setup we write the point constraint as $\mathbf{x}^*$ instead of $C_{\mathbf{x}^*}$ and let $C_{\mathbf{x}^*}(f) = 1$ if $f(\mathbf{x}^*) = 1$. We construct a laconic pre-constrained ABE scheme for point constraints as follows.

$\mathsf{Setup}(1^\lambda, \mathbf{x}^*) \to (\mathsf{mpk}, \mathsf{msk}_{\mathbf{x}^*})$.

– Sample $\bar{\mathbf{B}} \leftarrow \mathbb{Z}_q^{\frac{n}{2} \times m}$, $\mathbf{S_B} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{\frac{n}{2} \times \frac{n}{2}}$, $\mathbf{E_B} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{\frac{n}{2} \times m}$ and define

$$\mathbf{B} := \begin{bmatrix} \bar{\mathbf{B}} \\ \mathbf{S_B} \cdot \bar{\mathbf{B}} + \mathbf{E_B} \end{bmatrix} \in \mathbb{Z}_q^{n \times m}.$$

– Sample $(\mathbf{W}, \mathbf{T_W}) \leftarrow \mathsf{TrapGen}(1^{2m^2 n}, 1^m, q)$ and compute $\mathbf{T} \leftarrow [\mathbf{I}_{2m^2} \otimes \mathbf{B} \;\; \mathbf{W}]^{-1}(\mathbf{I}_{2m^2} \otimes \mathbf{G})$ using $\mathbf{T_W}$.

- Set $\mathsf{pp} := (\mathbf{B}, \mathbf{W}, \mathbf{T})$, compute $\mathbf{C}_{\mathbf{x}^*} := \mathsf{Com}(\mathsf{pp}, \mathbf{x}^*)$ and $\mathbf{Z}_{\mathbf{x}^*} := \mathsf{Open}(\mathsf{pp}, \mathbf{x}^*)$.
- Sample $\mathbf{U} \leftarrow \{0,1\}^{m\times m}$, and compute $\mathbf{B}_1 := \mathbf{B} \cdot \mathbf{U} - \mathbf{C}_{\mathbf{x}^*} \in \mathbb{Z}_q^{n\times m}$.
- Output $\mathsf{mpk} := (\mathbf{B}, \mathbf{B}_1, \mathbf{W}, \mathbf{T}_{\mathbf{W}}, \mathbf{T})$ and $\mathsf{msk}_{\mathbf{x}^*} := (\mathbf{x}^*, \mathbf{C}_{\mathbf{x}^*}, \mathbf{Z}_{\mathbf{x}^*}, \mathbf{U})$.

$\mathsf{KeyGen}(\mathsf{msk}_{\mathbf{x}^*}, f) \to \mathsf{sk}_f$. If $f(\mathbf{x}^*) \neq 1$, abort, else

- Compute $\mathbf{V}_\ell = \mathsf{Ver}(\mathsf{pp}, \ell)$, where $\mathsf{pp} = (\mathbf{B}, \mathbf{W}, \mathbf{T})$ and set $\mathbf{A} := -\mathbf{B}_1\mathbf{V}_\ell$.
- Compute $\mathbf{A}_f := \mathsf{EvalF}(\mathbf{A}, f)$ and $\mathbf{H}_{\mathbf{A},f,\mathbf{x}^*} := \mathsf{EvalFX}(\mathbf{A}, f, \mathbf{x}^*)$.
- Sample $\mathbf{D} \leftarrow \mathsf{SamplePre}\left([\mathbf{B}\ \ \mathbf{A}_f], \begin{pmatrix} (\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}\mathbf{V}_\ell)\cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}^*} \\ \mathbf{I}_m \end{pmatrix}, \mathbf{0}, \sigma_1\right)$.

  Here, we use the fact that $[\mathbf{B}\ \ \mathbf{A}_f]\begin{pmatrix} (\mathbf{Z}_{\mathbf{x}^*} + \mathbf{U}\mathbf{V}_\ell)\cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}^*} \\ \mathbf{I}_m \end{pmatrix} = f(\mathbf{x}^*)\mathbf{G}$ to apply $\mathsf{SamplePre}$.
- Output $\mathsf{sk}_f := \mathbf{D}$.

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, m) \to \mathsf{ct}$.

- Sample $\mathbf{S}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n\times\frac{n}{2}}$, $\mathbf{S}_2 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{n\times\frac{n}{2}}$, $\mathbf{E}_0 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n\times m}$, $\mathbf{E}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n\times m}$, $\mathbf{r} \leftarrow \{0,1\}^\lambda$.
- Set $\mathbf{S} = [\mathbf{S}_1\ \ \mathbf{S}_2]$ and compute $\mathbf{C}_{\mathbf{x}} = \mathsf{Com}(\mathsf{pp}, \mathbf{x})$.
- Output ciphertext $\mathsf{ct} := (\mathbf{C}_0, \mathbf{C}_1, \mathbf{C}_2, \mathbf{r})$ where $\mathbf{C}_0 := \mathbf{S}\cdot\mathbf{B} + \mathbf{E}_0$, $\mathbf{C}_1 := \mathbf{S}\cdot(\mathbf{B}_1 + \mathbf{C}_{\mathbf{x}}) + \mathbf{E}_1$, $\mathbf{C}_2 := \mathsf{Ext}(\mathbf{S}_2, \mathbf{r}) \oplus m$.

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, \mathsf{ct}) \to m'$.

- Parse $\mathsf{mpk} = (\mathbf{B}, \mathbf{B}_1, \mathbf{W}, \mathbf{T}_{\mathbf{W}}, \mathbf{T})$, $\mathsf{sk}_f = \mathbf{D}$, and $\mathsf{ct} = (\mathbf{C}_0, \mathbf{C}_1, \mathbf{C}_2, \mathbf{r})$. Set $\mathsf{pp} = (\mathbf{B}, \mathbf{W}, \mathbf{T})$.
- Compute $\mathbf{V}_\ell = \mathsf{Ver}(\mathsf{pp}, \ell)$, $\mathbf{Z}_{\mathbf{x}} = \mathsf{Open}(\mathsf{pp}, \mathbf{x})$, $\mathbf{A} = -\mathbf{B}_1\mathbf{V}_\ell$ and $\mathbf{H}_{\mathbf{A},f,\mathbf{x}} := \mathsf{EvalFX}(\mathbf{A}, f, \mathbf{x})$.
- Compute $\mathbf{C}_{f,\mathbf{x}} := [\,\mathbf{C}_0\ \ \mathbf{C}_1\,]\begin{pmatrix} -\mathbf{Z}_{\mathbf{x}} \\ -\mathbf{V}_\ell \end{pmatrix}\cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}}$.
- Compute $\mathbf{E}_{f,\mathbf{x}}$ by solving the equation: $\mathbf{E}_{f,\mathbf{x}}\cdot\mathbf{D} = [\,\mathbf{C}_0\ \ \mathbf{C}_{f,\mathbf{x}}\,]\cdot\mathbf{D} \mod q$.
- Compute $\mathbf{S}$ by solving: $\mathbf{S}\cdot[\,\mathbf{B}\ \ \mathbf{A}_f\,] = [\,\mathbf{C}_0\ \ \mathbf{C}_{f,\mathbf{x}}\,] - \mathbf{E}_{f,\mathbf{x}} \mod q$.
- Parse $\mathbf{S} = [\mathbf{S}_1\ \ \mathbf{S}_2]$, and output $m' = \mathbf{C}_2 \oplus \mathsf{Ext}(\mathbf{S}_2, \mathbf{r})$.

Succinct Master Public key. We have $|\mathsf{mpk}| = \mathrm{poly}(\lambda, d)$, which is independent of the length of the punctured attribute.

Correctness We show that our scheme is correct for any function f such that $f(\mathbf{x}^*) = 1$ and attribute $\mathbf{x}$ such that $f(\mathbf{x}) = 0$, with secret key $\mathsf{sk}_f = \mathbf{D}$ and ciphertext $\mathsf{ct} = (\mathbf{C}_0, \mathbf{C}_1, \mathbf{C}_2, \mathbf{r}) \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, m)$. In the decryption algorithm we have

$$\begin{aligned}
\mathbf{C}_{f,\mathbf{x}} &= [\,\mathbf{C}_0\ \ \mathbf{C}_1\,]\cdot\begin{pmatrix} -\mathbf{Z}_{\mathbf{x}} \\ -\mathbf{V}_\ell \end{pmatrix}\cdot\mathbf{H}_{\mathbf{A},f,\mathbf{x}} \\
&= (\mathbf{S}\,[\mathbf{B}\ \ \mathbf{B}_1 + \mathbf{C}_{\mathbf{x}}] + [\mathbf{E}_0\ \ \mathbf{E}_1])\cdot\begin{pmatrix} -\mathbf{Z}_{\mathbf{x}} \\ -\mathbf{V}_\ell \end{pmatrix}\cdot\mathbf{H}_{\mathbf{A},f,\mathbf{x}} \\
&= \mathbf{S}\,[\mathbf{B}\ \ \mathbf{B}_1 + \mathbf{C}_{\mathbf{x}}]\cdot\begin{pmatrix} -\mathbf{Z}_{\mathbf{x}} \\ -\mathbf{V}_\ell \end{pmatrix}\cdot\mathbf{H}_{\mathbf{A},f,\mathbf{x}} + \mathbf{E}'_{f,\mathbf{x}} \\
\text{(using Equation (2))} &= \mathbf{S}(\underbrace{-\mathbf{B}_1\mathbf{V}_\ell}_{\mathbf{A}} - \mathbf{x}\otimes\mathbf{G})\cdot\mathbf{H}_{\mathbf{A},f,\mathbf{x}} + \mathbf{E}'_{f,\mathbf{x}} \\
\text{(using Equation (1))} &= \mathbf{S}(\mathbf{A}_f - f(\mathbf{x})\mathbf{G}) + \mathbf{E}'_{f,\mathbf{x}}
\end{aligned}$$

Since $f(x) = 0$, the above implies $\mathbf{C}_{f,\mathbf{x}} = \mathbf{S}\mathbf{A}_f + \mathbf{E}'_{f,\mathbf{x}}$. Using this we have $[\mathbf{C}_0 \;\; \mathbf{C}_{f,\mathbf{x}}] = \mathbf{S}[\mathbf{B} \;\; \mathbf{A}_f] + [\mathbf{E}_0 \;\; \mathbf{E}'_{f,\mathbf{x}}]$. Set $\mathbf{E}_{f,\mathbf{x}} = [\mathbf{E}_0 \;\; \mathbf{E}'_{f,\mathbf{x}}]$. Next, note that $[\mathbf{C}_0 \;\; \mathbf{C}_{f,\mathbf{x}}] \cdot \mathbf{D} = \mathbf{S}[\mathbf{B} \;\; \mathbf{A}_f] \cdot \mathbf{D} + \mathbf{E}_{f,\mathbf{x}} \cdot \mathbf{D} = \mathbf{E}_{f,\mathbf{x}} \cdot \mathbf{D}$. Assuming $\mathbf{D}$ is full rank with all but negligible probability [10] , we solve for $\mathbf{E}_{f,\mathbf{x}}$ from $[\mathbf{C}_0 \;\; \mathbf{C}_{f,\mathbf{x}}] \cdot \mathbf{D} = \mathbf{E}_{f,\mathbf{x}} \cdot \mathbf{D}$ and for $\mathbf{S}$ from $[\mathbf{C}_0 \;\; \mathbf{C}_{f,\mathbf{x}}] = \mathbf{S}[\mathbf{B} \;\; \mathbf{A}_f] + \mathbf{E}_{f,\mathbf{x}}$. Parse $\mathbf{S} = [\mathbf{S}_1 \;\; \mathbf{S}_2]$ and compute $\mathrm{Ext}(\mathbf{S}_2, \mathbf{r}) \oplus \mathbf{C}_2 = \mathrm{Ext}(\mathbf{S}_2, \mathbf{r}) \oplus \mathrm{Ext}(\mathbf{S}_2, \mathbf{r}) \oplus m = m$, and hence the correctness.

4.2 Security

Theorem 4.1. (Security Against Authority) The above construction is secure against a semi-malicious authority (Definition 2.2).

Proof. Consider an adversary $\mathcal{A}$ that outputs the master public key mpk, randomness r, point constraint $\mathbf{x}^*$, such that $f(\mathbf{x}^*) = 1$ for all $f \in \mathcal{F}$, challenge attribute $\mathbf{x}$, and two messages m_0, m_1. We consider the non-trivial scenario where $\mathbf{x}^* = \mathbf{x}$. To prove unconditional security against the authority we need to show $\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}^*, m_0) \approx_s \mathsf{Enc}(\mathsf{mpk}, \mathbf{x}^*, m_1)$. We prove the above via the following sequence of hybrids.

Hyb_0. This is the real world where the challenger encrypts the message m_0.

Hyb_1. This hybrid is same as the previous hybrid except the following changes

- Rewrite $\mathbf{C}_0 = \mathbf{S} \cdot \mathbf{B} + \mathbf{E}_0$ as
$$\mathbf{C}_0 := [\mathbf{S}_1 + \mathbf{S}_2\mathbf{S}_\mathbf{B}]\bar{\mathbf{B}} + [\mathbf{S}_2\mathbf{E}_\mathbf{B} + \mathbf{E}_0].$$
- Rewrite $\mathbf{C}_1 = \mathbf{S} \cdot (\mathbf{B}_1 + \mathbf{C}_{\mathbf{x}^*}) + \mathbf{E}_1$, as
$$\mathbf{C}_1 := [\mathbf{S}_1 + \mathbf{S}_2\mathbf{S}_\mathbf{B}]\bar{\mathbf{B}}\mathbf{U} + [\mathbf{S}_2\mathbf{E}_\mathbf{B}\mathbf{U} + \mathbf{E}_1].$$

Hyb_0 is identical to Hyb_1, since we only substitute $\mathbf{B}$ with $\begin{pmatrix} \bar{\mathbf{B}} \\ \mathbf{S}_\mathbf{B} \cdot \bar{\mathbf{B}} + \mathbf{E}_\mathbf{B} \end{pmatrix}$ and $\mathbf{B}_1 + \mathbf{C}_{\mathbf{x}^*}$ with $\mathbf{B}\mathbf{U}$.

Hyb_2. In this hybrid we substitute $[\mathbf{S}_1 + \mathbf{S}_2\mathbf{S}_\mathbf{B}]$ with a freshly sampled secret matrix $\mathbf{S}' \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n \times n/2}$. Similarly we substitute $[\mathbf{S}_2\mathbf{E}_\mathbf{B} + \mathbf{E}_0]$ with $\mathbf{E}'_0 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n \times m}$ and $[\mathbf{S}_2\mathbf{E}_\mathbf{B}\mathbf{U} + \mathbf{E}_1]$ with $\mathbf{E}'_1 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n \times m}$. $\mathsf{Hyb}_1 \approx_s \mathsf{Hyb}_2$ using the noise flooding lemma (Lemma 2.18). To see this we note that $\|\mathbf{S}_2\mathbf{S}_\mathbf{B}\| \leq \|\mathbf{S}_2\|\|\mathbf{S}_\mathbf{B}\| \leq n^2\sigma^2\lambda$. Similarly, $\|\mathbf{S}_2\mathbf{E}_\mathbf{B}\| \leq nm\sigma^2\lambda$, $\|\mathbf{S}_2\mathbf{E}_\mathbf{B}\mathbf{U}\| \leq nm^2\sigma^2\lambda$. We set $\sigma' = \sigma^2 nm^2\lambda \cdot 2^{\omega(\log\lambda)}$ and thus we have $\mathsf{SD}((\mathbf{S}_1 + \mathbf{S}_2\mathbf{S}_\mathbf{B}), \mathbf{S}'), \mathsf{SD}((\mathbf{S}_2\mathbf{E}_\mathbf{B} + \mathbf{E}_0), \mathbf{E}'_0), \mathsf{SD}((\mathbf{S}_2\mathbf{E}_\mathbf{B}\mathbf{U} + \mathbf{E}_1), \mathbf{E}'_1) \leq \mathsf{negl}(\lambda)$. Hence, $\mathsf{Hyb}_1 \approx_s \mathsf{Hyb}_2$. We get
$$\mathbf{C}_0 = \mathbf{S}'\bar{\mathbf{B}} + \mathbf{E}'_0, \;\; \mathbf{C}_1 = \mathbf{S}'\bar{\mathbf{B}}\mathbf{U} + \mathbf{E}'_1, \;\; \mathbf{C}_2 := \mathrm{Ext}(\mathbf{S}_2, \mathbf{r}) \oplus m_0.$$

Hyb_3. Here we modify the way $\mathbf{C}_2$ is computed by replacing $\mathrm{Ext}(\mathbf{S}_2, \mathbf{r})$ with an independently uniform random string $u \leftarrow \{0,1\}^\lambda$. Note that $\mathbf{S}_2$ does not appear elsewhere in the ciphertext and is not used anywhere else in Hyb_2. So we have $\mathbf{C}_2 = u \oplus m_0$, which is a uniformly random bit due the randomness of u.

Next we unroll the hybrids to get to the real world where we encrypt the message m_1. □

Constraint Hiding Next, we show that our construction satisfies constraint hiding.

Theorem 4.2. The above construction satisfies constraint hiding Definition 2.3.

Proof. Let the point constraints chosen by the adversary be $\mathbf{x}_0, \mathbf{x}_1$ such that $f(\mathbf{x}_0) = f(\mathbf{x}_1)$. We want to prove

$$(\mathsf{mpk}_0 : (\mathsf{mpk}_0, \mathsf{msk}_0) \leftarrow \mathsf{Setup}(1^\lambda, \mathbf{x}_0)) \approx_c (\mathsf{mpk}_1 : (\mathsf{mpk}_1, \mathsf{msk}_1) \leftarrow \mathsf{Setup}(1^\lambda, \mathbf{x}_1)) \quad (5)$$

where the adversary also has the oracle access to $\mathsf{KeyGen}(\mathsf{msk}_0, \cdot)$ in the L.H.S of Equation (5) and to $\mathsf{KeyGen}(\mathsf{msk}_1, \cdot)$ in the R.H.S of Equation (5). We prove the theorem using the following sequence of hybrids.

[10] We let the KeyGen sample a $\mathbf{D}$ such that it is full rank.

Hyb_0. This is the real world where we run the setup using $\mathbf{x}_0$.

Hyb_1. This hybrid is same as the previous hybrid, except that in the key generation algorithm we abort if $f(\mathbf{x}_1) \neq 1$. Hyb_1 is identical to Hyb_2 since $f(\mathbf{x}_0) = f(\mathbf{x}_1)$.

Hyb_2. This hybrid is same as the previous hybrid except that in the setup we sample $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times m}$. $\mathsf{Hyb}_1 \approx_c \mathsf{Hyb}_2$ using LWE assumption (Assumption 2.20). To see this, we note that using LWE we can replace $\mathbf{S}_\mathbf{B} \cdot \bar{\mathbf{B}} + \mathbf{E}_\mathbf{B}$ with a uniformly sampled matrix from $\mathbf{B}' \leftarrow \mathbb{Z}_q^{\frac{n}{2} \times m}$. Thus we have $\mathbf{B} = (\bar{\mathbf{B}} \;\; \mathbf{B}')$ where $\bar{\mathbf{B}}, \mathbf{B}' \leftarrow \mathbb{Z}_q^{\frac{n}{2} \times m}$.

Hyb_3. This hybrid is same as the previous except that we sample $\mathbf{B}$ with a trapdoor, $(\mathbf{B}, \mathbf{T}_\mathbf{B}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$. $\mathsf{Hyb}_2 \approx_s \mathsf{Hyb}_3$ due to the properties of the trapdoor sampling (Lemma 2.16).

Hyb_4. This hybrid is same as the previous hybrid except that in the key generation, we sample $\mathbf{D}$ differently as $\mathbf{D} \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \;\; \mathbf{A}_f], \mathbf{T}_\mathbf{B}, \mathbf{0}, \sigma_1\right)$. $\mathsf{Hyb}_3 \approx_s \mathsf{Hyb}_4$ due to the properties of the trapdoor sampling (Lemma 2.16). In this hybrid, sampling $\mathbf{D}$ is independent of $\mathbf{x}_0$ and $\mathbf{x}_1$.

Hyb_5. This hybrid is same as the previous hybrid except that in the setup algorithm we sample $\mathbf{B}_1$ uniformly as $\mathbf{B}_1 \leftarrow \mathbb{Z}_q^{n \times m}$. $\mathsf{Hyb}_4 \approx_s \mathsf{Hyb}_5$ by Leftover Hash Lemma (Lemma 2.19). Note that in Hyb_4, we had $\mathbf{B}_1 := \mathbf{B} \cdot \mathbf{U} - \mathbf{C}_{\mathbf{x}_0}$, where $\mathbf{C}_{\mathbf{x}_0} = \mathsf{Com}(\mathsf{pp}, \mathbf{x}_0)$. By leftover hash lemma $\mathbf{B} \cdot \mathbf{U}$ is statistically close to a random matrix sampled from $\mathbb{Z}_q^{n \times m}$. Thus $\mathbf{B} \cdot \mathbf{U} - \mathbf{C}_{\mathbf{x}_0}$ is statistically close to a random matrix sampled from $\mathbb{Z}_q^{n \times m}$.

Henceforth we unroll the hybrids as above starting by computing $\mathbf{B}_1$ as $\mathbf{B}_1 := \mathbf{B} \cdot \mathbf{U} - \mathbf{C}_{\mathbf{x}_1}$, where $\mathbf{C}_{\mathbf{x}_1} = \mathsf{Com}(\mathsf{pp}, \mathbf{x}_1)$, and so on to get back to the real world where we run setup using $\mathbf{x}_1$. This completes the proof.

□

We summarize the result of this section using the following theorem.

Theorem 4.3. There exists a laconic pre-constrained attribute-based encryption (AB-PCE) scheme for point-constraints that satisfies semi-malicious security (as defined in Definition 2.2), under the LWE assumption with $|\mathsf{mpk}| = \mathrm{poly}(\lambda, d)$, where d denotes the maximum depth of the function class supported by the scheme.

The prior work by [AJJM22] constructed a AB-PCE scheme for point-constraints satisfying semi-malicious security assuming LWE and achieving $|\mathsf{mpk}| = \mathrm{poly}(\lambda, d, \ell)$, where d denotes the maximum depth of the function class supported by the scheme and ℓ denotes the attribute length.

Acknowledgement

References

[ABB10a] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572. Springer, Berlin, Heidelberg, May / June 2010. (Cited on page 16.)

[ABB10b] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 98–115. Springer, Berlin, Heidelberg, August 2010. (Cited on page 16.)

[ABD$^+$21] Navid Alamati, Pedro Branco, Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, and Sihang Pu. Laconic private set intersection and applications. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part III*, volume 13044 of *LNCS*, pages 94–125. Springer, Cham, November 2021. (Cited on page 3.)

[ADD$^+$23] Divesh Aggarwal, Nico Döttling, Jesko Dujmovic, Mohammad Hajiabadi, Giulio Malavolta, and Maciej Obremski. Algebraic restriction codes and their applications. *Algorithmica*, pages 1–47, 2023. (Cited on page 5, 18.)

[AGVW13] Shweta Agrawal, Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption: New perspectives and lower bounds. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 500–518. Springer, Berlin, Heidelberg, August 2013. (Cited on page 8, 27.)

[AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 119–135. Springer, Berlin, Heidelberg, May 2001. (Cited on page 5, 18.)

[AJJM20] Prabhanjan Ananth, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Multi-key fully-homomorphic encryption in the plain model. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 28–57. Springer, Cham, November 2020. (Cited on page 4.)

[AJJM21] Prabhanjan Ananth, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Unbounded multi-party computation from learning with errors. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 754–781. Springer, Cham, October 2021. (Cited on page 4.)

[AJJM22] Prabhanjan Ananth, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Pre-constrained encryption. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022. (Cited on page 2, 3, 4, 5, 6, 9, 10, 11, 12, 19, 41, 52, 53.)

[AMVY21] Shweta Agrawal, Monosij Maitra, Narasimha Sai Vempati, and Shota Yamada. Functional encryption for Turing machines with dynamic bounded collusion from LWE. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 239–269, Virtual Event, August 2021. Springer, Cham. (Cited on page 8, 9, 13, 28.)

[AV19] Prabhanjan Ananth and Vinod Vaikuntanathan. Optimal bounded-collusion secure functional encryption. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part I*, volume 11891 of *LNCS*, pages 174–198. Springer, Cham, December 2019. (Cited on page 8, 14.)

[BD18] Zvika Brakerski and Nico Döttling. Two-message statistically sender-private OT from LWE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 370–390. Springer, Cham, November 2018. (Cited on page 5, 17, 18, 19.)

[BD20] Zvika Brakerski and Nico Döttling. Lossiness and entropic hardness for ring-LWE. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 1–27. Springer, Cham, November 2020. (Cited on page 7.)

[BF03] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. (Cited on page 2.)

[BF22] Nir Bitansky and Sapir Freizeit. Statistically sender-private OT from LPN and derandomization. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 625–653. Springer, Cham, August 2022. (Cited on page 5, 18.)

[BGG$^+$14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 533–556. Springer, Berlin, Heidelberg, May 2014. (Cited on page 2, 6, 9, 17.)

[BGJP23] James Bartusek, Sanjam Garg, Abhishek Jain, and Guru-Vamsi Policharla. End-to-end secure messaging with traceability only for illegal content. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 35–66. Springer, Cham, April 2023. (Cited on page 3, 6, 10, 11, 53, 57, 58, 59, 61.)

[BGJS16] Saikrishna Badrinarayanan, Vipul Goyal, Aayush Jain, and Amit Sahai. Verifiable functional encryption. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 557–587. Springer, Berlin, Heidelberg, December 2016. (Cited on page 2.)

[BGMM20] James Bartusek, Sanjam Garg, Daniel Masny, and Pratyay Mukherjee. Reusable two-round MPC from DDH. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 320–348. Springer, Cham, November 2020. (Cited on page 4.)

[BGSZ22] James Bartusek, Sanjam Garg, Akshayaram Srinivasan, and Yinuo Zhang. Reusable two-round MPC from LPN. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part I*, volume 13177 of *LNCS*, pages 165–193. Springer, Cham, March 2022. (Cited on page 4.)

[BHY09] Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 1–35. Springer, Berlin, Heidelberg, April 2009. (Cited on page 48.)

[BJKL21] Fabrice Benhamouda, Aayush Jain, Ilan Komargodski, and Huijia Lin. Multiparty reusable non-interactive secure computation from LWE. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 724–753. Springer, Cham, October 2021. (Cited on page 4.)

[BL20] Fabrice Benhamouda and Huijia Lin. Mr NISC: Multiparty reusable non-interactive secure computation. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 349–378. Springer, Cham, November 2020. (Cited on page 4.)

[BLP$^+$13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 575–584. ACM Press, June 2013. (Cited on page 16.)

[BV18] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. *Journal of the ACM*, 65(6):39:1–39:37, 2018. (Cited on page 8, 26, 27.)

[CCH$^+$19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-Shamir: from practice to theory. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1082–1090. ACM Press, June 2019. (Cited on page 47, 48.)

[Cha07] Melissa Chase. Multi-authority attribute based encryption. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 515–534. Springer, Berlin, Heidelberg, February 2007. (Cited on page 2.)

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, Berlin, Heidelberg, May / June 2010. (Cited on page 16.)

[DGHM18] Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, and Daniel Masny. New constructions of identity-based and key-dependent message secure encryption schemes. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 3–31. Springer, Cham, March 2018. (Cited on page 8, 14, 15, 49.)

[DGI$^+$19] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 3–32. Springer, Cham, August 2019. (Cited on page 5, 18.)

[FJK23] Rex Fernando, Aayush Jain, and Ilan Komargodski. Maliciously-secure MrNISC in the plain model. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 98–128. Springer, Cham, April 2023. (Cited on page 4.)

[FPS$^+$18] Jonathan Frankle, Sunoo Park, Daniel Shaar, Shafi Goldwasser, and Daniel J Weitzner. Audit: Practical accountability of secret processes. *Cryptology ePrint Archive*, 2018. (Cited on page 11.)

[GHM$^+$19] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, Ahmadreza Rahimi, and Sruthi Sekar. Registration-based encryption from standard assumptions. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 63–93. Springer, Cham, April 2019. (Cited on page 2.)

[GHMR18] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ahmadreza Rahimi. Registration-based encryption: Removing private-key generator from IBE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 689–718. Springer, Cham, November 2018. (Cited on page 2.)

[GIKM00] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. *J. Comput. Syst. Sci.*, 60(3):592–629, 2000. (Cited on page 11.)

[GKVL21] Matthew Green, Gabriel Kaptchuk, and Gijs Van Laer. Abuse resistant law enforcement access systems. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 553–583. Springer, 2021. (Cited on page 11.)

[GLSW08] Vipul Goyal, Steve Lu, Amit Sahai, and Brent Waters. Black-box accountable authority identity-based encryption. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 2008*, pages 427–436. ACM Press, October 2008. (Cited on page 2.)

[Goy07] Vipul Goyal. Reducing trust in the PKG in identity based cryptosystems. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 430–447. Springer, Berlin, Heidelberg, August 2007. (Cited on page 2.)

[GP17] Shafi Goldwasser and Sunoo Park. Public accountability vs. secret laws: can they coexist? a cryptographic proposal. In *Proceedings of the 2017 on Workshop on Privacy in the Electronic Society*, pages 99–110, 2017. (Cited on page 11.)

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309. (Cited on page 2.)

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. (Cited on page 16.)

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Berlin, Heidelberg, August 2013. (Cited on page 17, 19.)

[GSW23] Vipul Goyal, Akshayaram Srinivasan, and Mingyuan Wang. Reusable secure computation in the plain model. In *Annual International Cryptology Conference*, pages 427–458. Springer, 2023. (Cited on page 5, 11.)

[GV20] Rishab Goyal and Satyanarayana Vusirikala. Verifiable registration-based encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 621–651. Springer, Cham, August 2020. (Cited on page 2.)

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 162–179. Springer, Berlin, Heidelberg, August 2012. (Cited on page 20.)

[HK12] Shai Halevi and Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. *Journal of Cryptology*, 25(1):158–193, January 2012. (Cited on page 5, 7, 18.)

[IK02] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan Eidenbenz, and Ricardo Conejo, editors, *ICALP 2002*, volume 2380 of *LNCS*, pages 244–256. Springer, Berlin, Heidelberg, July 2002. (Cited on page 7, 25.)

[IKSS23] Yuval Ishai, Dakshita Khurana, Amit Sahai, and Akshayaram Srinivasan. Black-box reusable NISC with random oracles. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 68–97. Springer, Cham, April 2023. (Cited on page 4.)

[KLN23] Markulf Kohlweiss, Anna Lysyanskaya, and An Nguyen. Privacy-preserving blueprints. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 594–625. Springer, Cham, April 2023. (Cited on page 10.)

[KN08] Gillat Kol and Moni Naor. Cryptography and game theory: Designing protocols for exchanging information. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 320–339. Springer, Berlin, Heidelberg, March 2008. (Cited on page 48.)

[KNT21] Fuyuki Kitagawa, Ryo Nishimaki, and Keisuke Tanaka. Simple and generic constructions of succinct functional encryption. *Journal of Cryptology*, 34(3):25, July 2021. (Cited on page 8, 26.)

[KNYY19] Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Exploring constructions of compact NIZKs from various assumptions. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 639–669. Springer, Cham, August 2019. (Cited on page 48.)

[KO04] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 335–354. Springer, Berlin, Heidelberg, August 2004. (Cited on page 4.)

[LPST16a] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation with non-trivial efficiency. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part II*, volume 9615 of *LNCS*, pages 447–462. Springer, Berlin, Heidelberg, March 2016. (Cited on page 8, 27.)

[LPST16b] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Output-compressing randomized encodings and applications. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 96–124. Springer, Berlin, Heidelberg, January 2016. (Cited on page 8, 26, 27.)

[LW11] Allison Lewko and Brent Waters. Unbounded hibe and attribute-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 547–567. Springer, 2011. (Cited on page 2.)

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Berlin, Heidelberg, April 2012. (Cited on page 16.)

[NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In S. Rao Kosaraju, editor, *12th SODA*, pages 448–457. ACM-SIAM, January 2001. (Cited on page 5, 18.)

[OPP14] Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 536–553. Springer, Berlin, Heidelberg, August 2014. (Cited on page 5, 7, 19.)

[OSEH13] Kazuma Ohara, Yusuke Sakai, Keita Emura, and Goichiro Hanaoka. A group signature scheme with unbounded message-dependent opening. In Kefei Chen, Qi Xie, Weidong Qiu, Ninghui Li, and Wen-Guey Tzeng, editors, *ASIACCS 13*, pages 517–522. ACM Press, May 2013. (Cited on page 11.)

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 89–114. Springer, Cham, August 2019. (Cited on page 47, 48.)

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, Berlin, Heidelberg, August 2008. (Cited on page 48.)

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):34:1–34:40, 2009. (Cited on page 48.)

[Sav18] Stefan Savage. Lawful device access without mass surveillance risk: A technical design discussion. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 1761–1774, 2018. (Cited on page 11.)

[Wee25] Hoeteck Wee. Almost optimal KP and CP-ABE for circuits from succinct LWE. Cryptology ePrint Archive, Paper 2025/509, 2025. (Cited on page 9, 17.)

[WWW22] Brent Waters, Hoeteck Wee, and David J. Wu. Multi-authority ABE from lattices without random oracles. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 651–679. Springer, Cham, November 2022. (Cited on page 16.)

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations extended abstract. In *23rd FOCS*, volume 28, 1982. (Cited on page 21.)

APPENDIX

A Additional Preliminaries

A.1 Dual-Mode Non-Interactive Zero-Knowledge Proof Systems

Let L be a language in NP, and let R be the associated binary relation such that for every $x \in \{0,1\}^*$, $x \in L$ if and only if there exists a witness w such that $(x, w) \in R$. A dual-mode non-interactive proof system for R consists of the following probabilistic polynomial-time algorithms.

$\mathsf{Hsetup}(1^\lambda) \to \mathsf{crs}$. This algorithm takes as input the security parameter λ and outputs a hiding common reference string crs.

$\mathsf{Bsetup}(1^\lambda) \to (\mathsf{crs}, \mathsf{td}_{\mathsf{ext}})$. This algorithm takes as input the security parameter λ and outputs a binding common reference string crs and an extraction trapdoor $\mathsf{td}_{\mathsf{ext}}$.

$\mathsf{Prove}(\mathsf{crs}, x, w) \to \pi$. This algorithm takes as input a crs, a statement x, and a witness w, and outputs a proof π.

$\mathsf{Verify}(\mathsf{crs}, x, \pi) \to 0/1$. The verification algorithm takes as input a crs, a statement x, and a proof π, and outputs either 1 (accept) or 0 (reject).

We require the dual mode NIZK proof system to satisfy the following properties.

Definition A.1 (Perfect completeness in both modes). A dual-mode NIZK proof system for the language L and the associated binary relation R is said to satisfy perfect completeness if for any $\lambda \in \mathbb{N}$, any statement $x \in L$ and corresponding witness w, the following holds

$$\Pr[\mathsf{Verify}(\mathsf{crs}, x, \mathsf{Prove}(\mathsf{crs}, x, w)) = 1] = 1$$

where $\mathsf{crs} \leftarrow \mathsf{Hsetup}(1^\lambda)$ or $(\mathsf{crs}, \mathsf{td}_{\mathsf{ext}}) \leftarrow \mathsf{Bsetup}(1^\lambda)$.

Definition A.2 (CRS indistinguishability). A dual-mode NIZK proof system for the language L and the associated binary relation R is said to satisfy CRS indistinguishability if there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, the following two distribution ensembles are computationally indistinguishable

$$\{\mathsf{crs} \mid \mathsf{crs} \leftarrow \mathsf{Hsetup}(1^\lambda)\} \approx_c \{\mathsf{crs} \mid (\mathsf{crs}, \mathsf{td}_{\mathsf{ext}}) \leftarrow \mathsf{Bsetup}(1^\lambda)\}.$$

Definition A.3 (Statistical zero-knowledge in hiding mode). A dual-mode NIZK proof system for the language L and the associated binary relation R is said to satisfy statistical zero-knowledge if there exists a probabilistic polynomial-time simulator Sim such that for any $(x, w) \in R$, the following two distribution ensembles are statistically indistinguishable for any $\lambda \in \mathbb{N}$

$$\{(\mathsf{crs}, \pi) : (\mathsf{crs}, \pi) \leftarrow \mathsf{Sim}(1^\lambda, x)\} \approx_s \{(\mathsf{crs}, \mathsf{Prove}(\mathsf{crs}, x, w)) : \mathsf{crs} \leftarrow \mathsf{Hsetup}(1^\lambda)\}.$$

Definition A.4 (Common random string). If crs output by Hsetup (resp. Bsetup) is uniformly random, we call it common random string in the hiding (resp. binding) mode.

Definition A.5 (Knowledge extraction in the binding mode). A dual-mode NIZK proof system for the language L and the associated binary relation R is said to satisfy knowledge extraction if there exists an extractor Ext, such that for any PPT adversary $\mathcal{A}$, the following holds

$$\Pr\left[\begin{array}{ll} & (\mathsf{crs}, \mathsf{td}_{\mathsf{ext}}) \leftarrow \mathsf{Bsetup}(1^\lambda); \\ \mathsf{Verify}(\mathsf{crs}, x, \pi) = 0 \vee (x, w) \in R : & (x, \pi) \leftarrow \mathcal{A}(\mathsf{crs}); \\ & w \leftarrow \mathrm{Ext}(\mathsf{td}_{\mathsf{ext}}, x, \pi) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Remark A.6. In the standard definition of dual-mode NIZK, we do not require that the CRS is a uniformly random string in the hiding mode. However, the construction based on the LWE assumption [CCH$^+$19, PS19] has this property.

The dual-mode NIZK by Canetti et al. [CCH+19] satisfies the following adaptive soundness instead of Definition A.5 (in the binding mode).

Definition A.7 (Adaptive Soundness). A dual-mode NIZK proof system for the language L and the associated binary relation R is said to satisfy adaptive soundness if for any PPT adversary $\mathcal{A}$, the following holds

$$\Pr\left[\ \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \wedge x \notin L : \begin{array}{l} (\mathsf{crs}, \mathsf{td}_{\mathsf{ext}}) \leftarrow \mathsf{Bsetup}(1^\lambda); \\ (x, \pi) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array} \right] \leq \mathsf{negl}(\lambda).$$

We can upgrade adaptive soundness to knowledge extraction by the following theorem.

Theorem A.8 ([KNYY19]). If a NIZK proof system is adaptively sound, and there exists PKE, we can convert the NIZK proof system into a NIZK that has the knowledge extraction property.

It is easy to see that this conversion preserves the dual-mode property and the uniformly random CRS property in the hiding mode if we use lossy PKE [PVW08, BHY09, KN08] with uniformly random lossy public keys [CCH+19] for the conversion. This is because the conversion adds a public key to the NIZK CRS, appends a ciphertext of a witness to a NIZK proof for proving the knowledge and the ciphertext is an encryption of the witness.

Theorem A.9 ([Reg09]). The Regev PKE is lossy encryption with uniformly random lossy public keys under the LWE assumption.

Theorem A.10 ([CCH+19, PS19]). There exists a dual-mode non-interactive zero-knowledge proof system for any NP language, based on the plain LWE problem with (small) polynomial approximation factors, satisfying statistical zero-knowledge with common random string in the hiding mode and adaptive soundness in the binding mode.

We can obtain the following corollary from the theorems above.

Corollary A.11. There exists a dual-mode non-interactive zero-knowledge proof system for any NP language, based on the plain LWE problem with (small) polynomial approximation factors, satisfying statistical zero-knowledge with common random string in the hiding mode and knowledge extraction in the binding mode.

A.2 One-Way Relation

Next, we provide the definition of one-way relation.

A one-way relation for a relation $\mathcal{R}$ consists of two algorithms $(\mathsf{Gen}, \mathsf{Sample})$ with the following syntax.

$\mathsf{Gen}(1^\lambda) \to \mathsf{pp}$. The generation algorithm takes as input the security parameter and outputs the public parameter pp.

$\mathsf{Sample}(\mathsf{pp}) \to (x, w)$. The sample algorithm outputs an instance witness pair (x, w).

A one-way relation satisfies the following properties.

Definition A.12 (Correctness). A one-way relation for a relation $\mathcal{R}$ is said to be correct if for any $\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda)$, the following holds

$$\Pr[(\mathsf{pp}, x, w) \in \mathcal{R} \mid (x, w) \leftarrow \mathsf{Sample}(\mathsf{pp})] \geq 1 - \mathsf{negl}(\lambda).$$

Definition A.13 (Security). A one-way relation for a relation $\mathcal{R}$ is said to be secure if for any PPT adversary $\mathcal{A}$, the following holds

$$\Pr\left[\ (\mathsf{pp}, x, w') \in \mathcal{R} : \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda); \\ (x, w) \leftarrow \mathsf{Sample}(\mathsf{pp}); \\ w' \leftarrow \mathcal{A}(\mathsf{pp}, x) \end{array} \right] \leq \mathsf{negl}(\lambda).$$

A.3 Digital Signatures

A digital signature scheme for a message space $\mathcal{M}$ consists of three algorithms $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ with the following syntax.

$\mathsf{Gen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$. The key generation algorithm takes as input the security parameter λ and outputs a verification key vk and a signing key sk.

$\mathsf{Sign}(\mathsf{sk}, m) \to \sigma$. The signing algorithm takes as input the signing key sk and a message $m \in \mathcal{M}$, and outputs a signature σ.

$\mathsf{Verify}(\mathsf{vk}, m, \sigma) \to 0/1$. The verification algorithm takes as input a verification key vk, a message $m \in \mathcal{M}$ and a signature σ, and outputs either 1 (accept) or 0 (reject).

A digital signature scheme satisfies the following properties.

Definition A.14 (Correctness). A digital signature scheme is sad to be correct if for any $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$, and any $m \in \mathcal{M}$, the following holds

$$\Pr[\mathsf{Verify}(\mathsf{vk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = 1] = 1.$$

Definition A.15 (EUF-CMA Security). A digital signature scheme is existentially unforgeable under chosen message attacks if for any PPT adversary $\mathcal{A}$, the following holds

$$\Pr\left[m \notin Q \wedge \mathsf{Verify}(\mathsf{vk}, m, \sigma) = 1 : \begin{array}{l} (\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda); \\ (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}(\mathsf{vk}) \end{array} \right] \leq \mathsf{negl}(\lambda)$$

where $Q \subset \mathcal{M}$ is the set of messages for which $\mathcal{A}$ makes the signing queries to $\mathsf{Sign}(\mathsf{sk}, \cdot)$.

B Missing details from Section 3

B.1 Succinct Hash Encryption with Semi-Malicious security in ROM.

In this section, we provide a construction of hash encryption satisfying semi-malicious security. To begin, we observe that the construction of HE from LWE by Döttling et al. [DGHM18] already satisfies semi-*honest* security. To make it succinct and semi-malicious secure in the ROM, we make the following two modifications:

1. Succinctness: We use a pseudorandom generator PRG to generate key.
2. Semi-malicious security: Let H be a hash function modelled as the random oracle. Let R be the (potentially bad) randomness sampled by the adversary in the semi-malicious game. Then, the key of HE is set R, using which the parties compute $H(\mathsf{PRG}(R))$.

We also note that for our purpose we need a hash encryption scheme with hash domain $\{-1, 0, 1\}^m$ satisfying semi-malicious security. We use the HE scheme for binary hash domain from [DGHM18] as a black box to construct our hash encryption scheme with hash domain $\{-1, 0, 1\}^m$ satisfying semi-malicious security.

Building blocks. We use the following ingredients for our construction.

1. A pseudorandom generator $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^*$.
2. A random oracle $H : \{0,1\}^* \to \mathbb{Z}_p^{2m \times \lambda}$.
3. A hash encryption scheme $\mathsf{HE} = (\mathsf{HE.Gen}, \mathsf{HE.Hash}, \mathsf{HE.Enc}, \mathsf{HE.Dec})$ satisfying semi-honest security for the hash domain $\{0,1\}^{2m}$ and message space $\mathcal{M}$. This can be instantiated from LWE assumption (Theorem 2.11).
4. A mapping $\phi : \mathcal{X} \to \{0,1\}^{2m}$, where, $\mathcal{X} \subseteq \{-1, 0, 1\}^m$ and $\phi(x_1 \ldots x_m) = y_1 \ldots y_{2m}$ where $y_{2i-1} = y_{2i} = 0$ if $x_i = 0$, $y_{2i-1} = y_{2i} = 1$ if $x_i = 1$, and $y_{2i-1} = 1, y_{2i} = 0$ if $x_i = -1$.

Construction. We describe our construction below.

$\mathsf{Gen}(1^\lambda, m) \to \mathsf{key}$. The setup algorithm does the following.

1. Sample $r \leftarrow \{0,1\}^\lambda$.
2. Compute $\mathbf{A} = H(\mathsf{PRG}(r))$ and set $\mathsf{HE.key} = \mathbf{A}$.
3. Output $\mathsf{key} = r$ and set the CRS as $\mathsf{HE.key}$.

$\mathsf{Hash}(\mathsf{key}, x \in \{-1,0,1\}^m) \to h$. The hash algorithm does the following.

1. Parse $\mathsf{key} = r$ and compute $\mathsf{HE.key} := H(\mathsf{PRG}(r))$.
2. Let $\phi(x) = y$ and compute $\mathsf{HE.hash} \leftarrow \mathsf{HE.Hash}(\mathsf{HE.key}, y)$.
3. Output $h = \mathsf{HE.hash}$.

$\mathsf{Enc}(\mathsf{key}, (h, i \in [m], c \in \{-1,0,1\}), \mu)$. The encryption algorithm does the following.

1. Parse $\mathsf{key} = r$, $h = \mathsf{HE.hash}$ and compute $\mathsf{HE.key} := H(\mathsf{PRG}(r))$.
2. Sample random $s_0 \leftarrow \mathcal{M}$ and set $s_1 = \mu \oplus s_0$.
3. Let $\phi(c) = c_0 c_1$ and compute $\mathsf{HE.ct}_0 \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key}, (\mathsf{HE.hash}, 2i-1, c_0), s_0)$ and $\mathsf{HE.ct}_1 \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key}, (\mathsf{HE.hash}, 2i, c_1), s_1)$.
4. Output $\mathsf{ct} = (\mathsf{HE.ct}_0, \mathsf{HE.ct}_1)$.

$\mathsf{Dec}(\mathsf{key}, x, \mathsf{ct}) \to \{0,1\}$. The decryption algorithm does the following.

1. Parse $\mathsf{key} = r$ and compute $\mathsf{HE.key} := H(\mathsf{PRG}(r))$.
2. Let $\phi(x) = y$.
3. Parse $\mathsf{ct} = (\mathsf{HE.ct}_0, \mathsf{HE.ct}_1)$ and compute $s'_0 \leftarrow \mathsf{HE.Dec}(\mathsf{HE.key}, y, \mathsf{HE.ct}_0)$ and $s'_1 \leftarrow \mathsf{HE.Dec}(\mathsf{HE.key}, y, \mathsf{HE.ct}_1)$.
4. Output $\mu' = s_0 \oplus s_1$.

Collusion resistance. First we note that ϕ is a one to one mapping, i.e. if $x_1 \neq x_2 \implies \phi(x_1) \neq \phi(x_2)$. This along with the fact that $\mathsf{HE.Hash}$ is a collusion resistant hash, implies that the Hash algorithm in the above construction is a collusion resistant hash for hash domain $\{-1,0,1\}^m$.

Correctness. For any $\mathsf{Enc}(\mathsf{key}, (h, i, c), \mu) = \mathsf{ct} = (\mathsf{HE.ct}_0, \mathsf{HE.ct}_1)$, where $\mathsf{HE.ct}_0 \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key}, (\mathsf{HE.hash}, 2i-1, c_0), s_0)$ and $\mathsf{HE.ct}_1 \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key}, (\mathsf{HE.hash}, 2i, c_1), s_1)$ and for any x such that $\mathsf{Hash}(\mathsf{key}, x) = h = \mathsf{HE.Hash}(\mathsf{HE.key}, y)$, where $y = \phi(x)$, we observe that with all but negligible probability

1. if $y_{2i-1} = c_0$,

$$\mathsf{HE.Dec}(\mathsf{HE.key}, y, \mathsf{HE.ct}_0) = \mathsf{HE.Dec}(\mathsf{HE.key}, y, \mathsf{HE.Enc}(\mathsf{HE.key}, (\mathsf{HE.hash}, 2i-1, c_0), s_0)) = s_0$$

2. and if $y_{2i} = c_1$

$$\mathsf{HE.Dec}(\mathsf{HE.key}, y, \mathsf{HE.ct}_1) = \mathsf{HE.Dec}(\mathsf{HE.key}, y, \mathsf{HE.Enc}(\mathsf{HE.key}, (\mathsf{HE.hash}, 2i, c_1), s_1)) = s_1$$

by the correctness of the underlying HE scheme. Now, if $x_i = c$, it implies that $\phi(x_i) = \phi(c) \implies y_{2i-1} y_{2i} = c_0 c_1$. So we recover s_0, s_1 in Step 3 of the decryption algorithm and $s_0 \oplus s_1 = \mu$ in the Step 4 by the correctness of secret sharing. Hence, the correctness follows.

Semi-malicious security. We show that the above construction of a hash encryption scheme is secure via following theorem.

Theorem B.1. Assume that HE is a secure hash encryption scheme for hash domain $\{0,1\}^{2m}$ in the semi-honest setting (Definition 2.8). Then the above construction is a secure hash encryption scheme in the semi-malicious setting for hash domain $\{-1,0,1\}^m$.

Proof. Recall that to prove the security we need to show that

$$\mathsf{Enc}(\mathsf{key},(h,i,c_i),\mu_0) \approx_c \mathsf{Enc}(\mathsf{key},(h,i,c_i),\mu_1)$$

where $h = \mathsf{Hash}(\mathsf{key},x)$ for some $x \in \{-1,0,1\}^m$ and $x_i \neq c_i$ for some $c_i \in \{-1,0,1\}$.
We consider the following sequence of hybrid games.

Hyb_0. This is the real world where the ciphertext is computed for message μ_0, We write the complete game to setup notations and for easy reference in the later hybrids.

1. The adversary outputs $x \in \{-1,0,1\}^m$ and a randomness r.
2. The challenger computes $y = \phi(x)$, generates $\mathbf{A} \leftarrow \mathsf{HE.Gen}(\mathsf{HE.key},1^{2m})$ and sets $\mathbf{A} := H(\mathsf{PRG}(r))$. It sets $\mathsf{key} = r$.
3. The adversary outputs an index $i \in [m]$, two messages μ_0, μ_1 and a $c_i \in \{-1,0,1\}$ such that $x_i \neq c_i$. The challenger does the following.
   (a) It computes $\phi(c_i) = d_{2i-1}d_{2i}$, where $d_{2i-1}, d_{2i} \in \{0,1\}$.
   (b) It samples $s_0^{(2i-1)}, s_1^{(2i-1)} \leftarrow \mathcal{M}$ and sets $s_0^{(2i)} = \mu_0 \oplus s_0^{(2i-1)}$, $s_1^{(2i)} = \mu_1 \oplus s_1^{(2i-1)}$ ($s_b^{(\mathsf{ind})}$ denotes the share of message μ_b at position ind).
   (c) It computes $\mathsf{HE.ct}_{(2i-1)} \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key},(\mathsf{HE.Hash}(\mathsf{HE.key},y),2i-1,d_{2i-1}),s_0^{(2i-1)})$ and $\mathsf{HE.ct}_{(2i)} \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key},(\mathsf{HE.Hash}(\mathsf{HE.key},y),2i,d_{2i}),s_0^{(2i)})$.
4. The challenger returns $\mathsf{ct} = (\mathsf{HE.ct}_{(2i-1)},\mathsf{HE.ct}_{(2i)})$ to the adversary.
5. The adversary outputs a guess bit β.

Hyb_1. In this hybrid we change the way the ciphertext is generated. In particular for $j \in \{2i-1,2i\}$ such that $y_j \neq d_j$ (since $x_i \neq c_i$, then by the definition of ϕ, it implies there exists such an index j), the challenger computes $\mathsf{HE.ct}_{(j)} \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key},(\mathsf{HE.Hash}(\mathsf{HE.key},y),j,d_j,s_1^{(j)})$.

Hyb_2. This hybrid is same as the previous hybrid except that we set $s_0^{(j')} := s_1^{(j)} \oplus \mu_1$, where $j' \in \{2i-1,2i\}$ such that $j' \neq j$.
We note that this hybrid is the real world where the ciphertext is computed for message μ_1.

Indistinguishability of hybrids. Now we show that the above consecutive hybrids are indistinguishable.

Claim B.2. Assume that HE is a secure hash encryption scheme in the semi-honest setting. Then $\mathsf{Hyb}_0 \approx_c \mathsf{Hyb}_1$.

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ who can distinguish between the two hybrids with non-negligible advantage ϵ, then there exists a PPT adversary $\mathcal{B}$ against the semi-honest security of the underlying HE scheme with the same advantage ϵ. The reduction is as follows.

1. $\mathcal{B}$ first runs $\mathcal{A}$. The adversary outputs $x \in \{-1,0,1\}^m$ and a randomness r.

2. $\mathcal{B}$ computes $y = \phi(x)$ and sends y to the HE challenger. The challenger generates $\mathbf{A} \leftarrow \mathsf{HE.Gen}(\mathsf{HE.key},1^{2m})$ and returns the $\mathsf{HE.key} = \mathbf{A}$ to $\mathcal{B}$.

3. $\mathcal{B}$ sets the $H(\mathsf{PRG}(r)) = \mathbf{A}$ and returns $\mathbf{A}$ to the adversary $\mathcal{A}$.

4. $\mathcal{A}$ outputs an index $i \in [m]$, two messages μ_0, μ_1 and a $c_i \in \{-1, 0, 1\}$ such that $x_i \neq c_i$. $\mathcal{B}$ does the following.
   - It computes $\phi(c_i) = d_{2i-1}d_{2i}$, where $d_{2i-1}, d_{2i} \in \{0, 1\}$.
   - It samples $s_0^{(2i-1)}, s_1^{(2i-1)} \leftarrow \mathcal{M}$ and sets $s_0^{(2i)} = \mu_0 \oplus s_0^{(2i-1)}$, $s_1^{(2i)} = \mu_1 \oplus s_1^{(2i-1)}$.
   - For $j \in \{2i-1, 2i\}$ such that $y_j \neq d_j$, $\mathcal{B}$ sends $(j, (s_0^{(j)}, s_1^{(j)}), d_j)$ as the ciphertext challenge to the HE challenger. The HE challenger samples a bit $\beta \leftarrow \{0, 1\}$ and computes $\mathsf{HE.ct}_{(j)} \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key}, (\mathsf{HE.Hash}(\mathsf{HE.key}, y), j, d_j), s_\beta^{(j)})$ and returns $\mathsf{HE.ct}_{(j)}$ to $\mathcal{B}$.
   - Let $j' \in \{2i-1, 2i\}$ s.t $j' \neq j$. $\mathcal{B}$ computes $\mathsf{HE.ct}_{(j')} \leftarrow \mathsf{HE.Enc}(\mathsf{HE.key}, (\mathsf{HE.Hash}(\mathsf{HE.key}, y), j', d_{j'}), s_0^{(j')})$.
   - $\mathcal{B}$ sets $\mathsf{ct} = (\mathsf{HE.ct}_{(j)}, \mathsf{HE.ct}_{(j')})$ if $j = 2i-1$, else it sets $\mathsf{ct} = (\mathsf{HE.ct}_{(j')}, \mathsf{HE.ct}_{(j)})$ and sends ct to $\mathcal{A}$.
5. In the end $\mathcal{A}$ outputs a bit β'. $\mathcal{B}$ forwards β' to the HE challenger.

We observe that if the HE challenger samples $\beta = 0$, then $\mathcal{B}$ simulated the real world where μ_0 is encrypted , else the real world where μ_1 is encrypted with $\mathcal{A}$. Hence, advantage of $\mathcal{B}$ = $|\Pr(\beta' = 1|\beta = 0) - \Pr(\beta' = 1|\beta = 1)|$ = $|\Pr(\beta' = 1|\mathsf{Hyb}_0) - \Pr(\beta' = 1|\mathsf{Hyb}_1)|$ = ϵ (by assumption). □

Claim B.3. Hyb_1 and Hyb_2 are statistically indistinguishable.

Proof. We note that $s_0^{(j')}$ is a uniformly random string, since each share of μ_0 is distributed uniformly at random. Also, since the other share of μ_0, which is $s_0^{(j)}$, does not appear in Hyb_1 and Hyb_2, we can set $s_0^{(j')} = s_1^{(j)} \oplus \mu_1$, which is again a uniformly random string due to the randomness of $s_1^{(j)}$ and the guarantee of secret sharing. We also note that now $\mathsf{HE.ct}_{(j')}$ encrypts the message $s_1^{(j)} \oplus \mu_1$. □

□

C Laconic Identity-Based PCE for General Constraints

In this section we present a construction for a IB-PCE scheme for general constraints. Following [AJJM22], we also note that with simple modifications to the AB-PCE construction in Section 4.1, we can achieve a IB-PCE scheme for general constraints. Moreover, our construction achieves a succinct master public key.

We describe our construction below.

$\mathsf{Setup}(1^\lambda, C) \rightarrow (\mathsf{mpk}, \mathsf{msk}_C)$. The setup algorithm parses the constraint C as an ℓ bit string and does the following.

- Sample $\bar{\mathbf{B}} \leftarrow \mathbb{Z}_q^{\frac{n}{2} \times m}$, $\quad \mathbf{S_B} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{\frac{n}{2} \times \frac{n}{2}}$, $\quad \mathbf{E_B} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{\frac{n}{2} \times m}$ and define
$$\mathbf{B} := \begin{bmatrix} \bar{\mathbf{B}} \\ \mathbf{S_B} \cdot \bar{\mathbf{B}} + \mathbf{E_B} \end{bmatrix} \in \mathbb{Z}_q^{n \times m}.$$
- Sample $(\mathbf{W}, \mathbf{T_W}) \leftarrow \mathsf{TrapGen}(1^{2m^2 n}, 1^m, q)$ and compute $\mathbf{T} \leftarrow [\mathbf{I}_{2m^2} \otimes \mathbf{B} \ \ \mathbf{W}]^{-1}(\mathbf{I}_{2m^2} \otimes \mathbf{G})$ using $\mathbf{T_W}$.
- Set $\mathsf{pp} := (\mathbf{B}, \mathbf{W}, \mathbf{T})$, compute $\mathbf{C}_C := \mathsf{Com}(\mathsf{pp}, C)$ and $\mathbf{Z}_C := \mathsf{Open}(\mathsf{pp}, C)$.
- Sample $\mathbf{U} \leftarrow \{0, 1\}^{m \times m}$, and compute $\mathbf{B}_1 := \mathbf{B} \cdot \mathbf{U} - \mathbf{C}_C \in \mathbb{Z}_q^{n \times m}$.
- Output $\mathsf{mpk} := (\mathbf{B}, \mathbf{B}_1, \mathbf{W}, \mathbf{T_W}, \mathbf{T})$ $\quad$ and $\quad$ $\mathsf{msk}_C := (C, \mathbf{C}_C, \mathbf{Z}_C, \mathbf{U})$.

$\mathsf{KeyGen}(\mathsf{msk}_C, \mathsf{id}) \rightarrow \mathsf{sk}_{\mathsf{id}}$. The key generation algorithm constructs a universal circuit $U[\mathsf{id}]$,with id hardwired, and does the following.

- Parse $\mathsf{msk}_C = (C, \mathbf{C}_C, \mathbf{Z}_C, \mathbf{U})$. If $C(\mathsf{id}) \neq 1$, abort.

- Compute $\mathbf{V}_\ell = \mathsf{Ver}(\mathsf{pp}, \ell)$, where $\mathsf{pp} = (\mathbf{B}, \mathbf{W}, \mathbf{T})$ and set $\mathbf{A} := -\mathbf{B}_1\mathbf{V}_\ell$.
- Parse C as an ℓ bit string and compute $\mathbf{A}_{\mathsf{id}} := \mathsf{EvalF}(\mathbf{A}, U[\mathsf{id}])$ and $\mathbf{H}_{\mathbf{A},U[\mathsf{id}],C} := \mathsf{EvalFX}(\mathbf{A}, U[\mathsf{id}], C)$.
- Sample $\mathbf{D} \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \;\; \mathbf{A}_{\mathsf{id}}],\ \begin{pmatrix} (\mathbf{Z}_C + \mathbf{U}\mathbf{V}_\ell) \cdot \mathbf{H}_{\mathbf{A},U[\mathsf{id}],C} \\ \mathbf{I}_m \end{pmatrix},\ \mathbf{0},\ \sigma_1\right)$.

  It is easy to note that $\left((\mathbf{Z}_C + \mathbf{U}\mathbf{V}_\ell) \cdot \mathbf{H}_{\mathbf{A},U[\mathsf{id}],C} \;\; \mathbf{I}_m\right)^{\mathsf{T}}$ is a gadget trapdoor for $[\mathbf{B} \;\; \mathbf{A}_{\mathsf{id}}]$.
- Output $\mathsf{sk}_{\mathsf{id}} := \mathbf{D}$.

$\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, m) \to \mathsf{ct}$. The encryption algorithm constructs a universal circuit $U[\mathsf{id}]$,with id hardwired, and does the following.

- Compute $\mathbf{V}_\ell = \mathsf{Ver}(\mathsf{pp}, \ell)$, where $\mathsf{pp} = (\mathbf{B}, \mathbf{W}, \mathbf{T})$ and set $\mathbf{A} := -\mathbf{B}_1\mathbf{V}_\ell$. Compute $\mathbf{A}_{\mathsf{id}} := \mathsf{EvalF}(\mathbf{A}, U[\mathsf{id}])$.
- Sample $\mathbf{S}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n \times \frac{n}{2}}$, $\mathbf{S}_2 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{n \times \frac{n}{2}}$, $\mathbf{E}_0 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n \times m}$, $\mathbf{E}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n \times m}$, $\mathbf{r} \leftarrow \{0,1\}^\lambda$.
- Set $\mathbf{S} = [\mathbf{S}_1 \;\; \mathbf{S}_2]$ and output ciphertext $\mathsf{ct} := (\mathbf{C}_0, \mathbf{C}_1, \mathbf{r})$ where

$$\mathbf{C}_0 := \mathbf{S} \cdot [\mathbf{B} \;\; \mathbf{A}_{\mathsf{id}}] + \mathbf{E}_0, \quad \mathbf{C}_1 := \mathrm{Ext}(\mathbf{S}_2, \mathbf{r}) \oplus m.$$

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, \mathsf{ct}) \to m'$.

- Parse $\mathsf{mpk} = (\mathbf{B}, \mathbf{B}_1, \mathbf{W}, \mathbf{T}_{\mathbf{W}}, \mathbf{T})$, $\mathsf{sk}_f = \mathbf{D}$, and $\mathsf{ct} = (\mathbf{C}_0, \mathbf{C}_1, \mathbf{r})$. Set $\mathsf{pp} = (\mathbf{B}, \mathbf{W}, \mathbf{T})$.
- Compute $\mathbf{E}_{\mathsf{id}}$ by solving the equation:

$$\mathbf{E}_{\mathsf{id}} \cdot \mathbf{D} = \mathbf{C}_0 \cdot \mathbf{D} \mod q.$$

- Compute $\mathbf{S}$ by solving:

$$\mathbf{S} \cdot [\,\mathbf{B} \;\; \mathbf{A}_{\mathsf{id}}\,] = \mathbf{C}_0 - \mathbf{E}_{\mathsf{id}} \mod q.$$

- Parse $\mathbf{S} = [\mathbf{S}_1 \;\; \mathbf{S}_2]$, and output $m' = \mathbf{C}_1 \oplus \mathrm{Ext}(\mathbf{S}_2, \mathbf{r})$.

Succinct Master Public Key. We have $\mathsf{mpk} = \mathrm{poly}(\lambda)$, which is independent of the constraint size.

The correctness, unconditional security against semi-malicious authority and the constraint hiding are simple adaptations of the respective proofs in Section 4.1, hence omitted. We summarize the result of this section using the following theorem.

Theorem C.1. There exists an identity-based pre-constrained encryption (IB-PCE) scheme for general constraints that satisfies semi-malicious security (as defined in Definition 2.2), under the LWE assumption, and achieves a succinct master public key of size $|\mathsf{mpk}| = \mathrm{poly}(\lambda)$.

The prior work by [AJJM22] constructed a IB-PCE scheme for general constraints satisfying semi-malicious security assuming LWE and achieving $|\mathsf{mpk}| = \mathrm{poly}(\lambda, \ell)$, where ℓ denotes the constraint size.

D Pre-Constrained Group Signatures

In this section, we provide our definition and construction for pre-constrained group signatures. Our definition largely follows the definition of [BGJP23] except that we generalize the constraint of database membership to arbitrary constraints, and favour simpler game based definitions for security against authorities as compared to the simulation style definitions of [BGJP23]. Another difference is that their definition also includes the step of authorizing the database while ours does not. We note that such an authorization can be performed via a separate protocol (using zero-knowledge or multiparty computation protocols).

D.1 Definition

A pre-constrained group signature (PCGS) scheme for a circuit family $\mathcal{C} = \{C : \{0,1\}^n \to \{0,1\}\}$ for $n = n(\lambda)$, a message space $\mathcal{M} = \{0,1\}^n$, an identity space $\mathcal{ID}$ consists of five algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Open})$ with the following syntax.

$\mathsf{Setup}(1^\lambda, C) \to (\mathsf{mpk}, \mathsf{msk})$. The setup algorithm, run by the group manager GM, takes as input the security parameter λ and a circuit $C \in \mathcal{C}$, and outputs a master public key mpk and a master secret key msk.

$\mathsf{KeyGen}\langle \mathsf{GM}(\mathsf{msk}), U\rangle \to (\mathsf{id}, \mathsf{sk}_{\mathsf{id}})$. This is an interactive protocol between the group manager GM with msk and the user U. It delivers an identity $\mathsf{id} \in \mathcal{ID}$ to both GM and U and a user secret signing key $\mathsf{sk}_{\mathsf{id}}$ to U.

$\mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, m) \to \sigma$. The signing algorithm takes as input the master public key mpk, the user signing key $\mathsf{sk}_{\mathsf{id}}$ and a message m, and outputs a signature σ.

$\mathsf{Verify}(\mathsf{mpk}, m, \sigma) \to \{0,1\}$. The verification algorithm takes as input the master public key mpk, a message m and a signature σ, and outputs a bit indicating accept or reject.

$\mathsf{Open}(\mathsf{msk}, \sigma) \to \{\mathsf{id}, \perp\}$. The opening algorithm on input the master secret key msk and a signature σ outputs either an identity $\mathsf{id} \in \mathcal{ID}$ or $\perp$.

Definition D.1 (Correctness). A PCGS scheme is said to be correct if for any $C \in \mathcal{C}$, $m \in \mathcal{M}$, the following holds

$$\Pr[\mathsf{Verify}(\mathsf{mpk}, m, \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, m)) = 1] \geq 1 - \mathsf{negl}(\lambda)$$

where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, C)$ and $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}}) \leftarrow \mathsf{KeyGen}\langle \mathsf{GM}(\mathsf{msk}), U\rangle$.

Definition D.2 (Constraint-Hiding). A PCGS scheme is said to satisfy constraint-hiding security if for any PPT adversary $\mathcal{A}$, the following holds

$$\Pr\left[\beta' = \beta : \begin{array}{l} C_0, C_1 \leftarrow \mathcal{A}; \\ \beta \leftarrow \{0,1\}; (\mathsf{mpk}_\beta, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, C_\beta); \\ \beta' \leftarrow \mathcal{A}(\mathsf{mpk}_\beta) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ is admissible if $|C_0| = |C_1|$ and $C_0, C_1 \in \mathcal{C}$.

Definition D.3 (Traceability). For a PCGS scheme and an adversary $\mathcal{A}$, let us define the traceability experiment as follows.

1. $\mathcal{A}$ outputs a challenge circuit C.
2. The challenger generates $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, C)$. It sends mpk to $\mathcal{A}$.
3. $\mathcal{A}$ can make the following queries to the challenger.
   (a) H-KeyGen. $\mathcal{A}$ can request the joining of an user U to the group. The challenger executes the KeyGen protocol where it acts both as the group manager and the user. The challenger receives an $\mathsf{id} \in \mathcal{ID}$ and the respective signing key $\mathsf{sk}_{\mathsf{id}}$. It forwards id to $\mathcal{A}$ and maintains a set of these identities H_{id}. It also maintains a list H_{key} of the identity and the respective signing key pair $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}})$ for the queried identities.
   (b) C-KeyGen. $\mathcal{A}$ can request to join the group as an user U. Then the challenger and the adversary executes the KeyGen protocol where the challenger acts as the group manager and $\mathcal{A}$ as the user U. The challenger and $\mathcal{A}$ receives an $\mathsf{id} \in \mathcal{ID}$ and $\mathcal{A}$ additionally receives the signing key $\mathsf{sk}_{\mathsf{id}}$. The challenger maintains a set of these identities C_{id}.
   (c) Sign. $\mathcal{A}$ can request a signature on message m. It sends (m, id) to the challenger, where the id is the user identity corresponding to some user in the group. If $\mathsf{id} \notin H_{\mathsf{id}}$, the challenger outputs $\perp$ else it returns $\sigma \leftarrow \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, m)$ to $\mathcal{A}$. The challenger maintains a list S_{id} of (m, id) for which it returns the signature to $\mathcal{A}$.

(d) Open. $\mathcal{A}$ can request to open a signature σ. The challenger runs $\mathsf{Open}(\mathsf{msk}, \sigma)$ and forwards the output to $\mathcal{A}$.

4. $\mathcal{A}$ outputs a message m^* and a signature σ^*.

$\mathcal{A}$ wins if the following conditions hold

1. $\mathsf{Verify}(\mathsf{mpk}, m, \sigma) = 1$ and $C(m) = 1$.
2. $\mathsf{id} \leftarrow \mathsf{Open}(\mathsf{msk}, \sigma) \wedge ((m, \mathsf{id}) \notin S_{\mathsf{id}}) \wedge (\mathsf{id} \notin \mathsf{C}_{\mathsf{id}})$.

We say that a PCGS scheme is traceable if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the traceability experiment is $\mathsf{negl}(\lambda)$.

Definition D.4 (Unframeability). For a PCGS scheme and an adversary $\mathcal{A}$, let us define the unframeability experiment as follows.

1. $\mathcal{A}$ outputs a circuit $C \in \mathcal{C}$.
2. On input $1^\lambda, C$, the challenger generates $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, C)$. It sends $(\mathsf{mpk}, \mathsf{msk})$ to $\mathcal{A}$.
3. $\mathcal{A}$ can make the H-KeyGen and Sign queries to the challenger as defined in Definition D.3. The challenger maintains the list H_{id} and S_{id} for the respective queries.
4. $\mathcal{A}$ outputs a message m^* and a signature σ^*.

$\mathcal{A}$ wins if the following conditions hold

1. $\mathsf{Verify}(\mathsf{mpk}, m, \sigma) = 1$.
2. $\mathsf{id} \leftarrow \mathsf{Open}(\mathsf{msk}, \sigma) \wedge ((m, \mathsf{id}) \notin S_{\mathsf{id}}) \wedge (\mathsf{id} \in H_{\mathsf{id}})$.

We say that a PCGS scheme satisfies unframeability if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the unframeability experiment is $\mathsf{negl}(\lambda)$.

Definition D.5 (Client-Authority Anonymity against Malicious Authority). For a PCGS scheme and an adversary $\mathcal{A}$, let us define the experiment for anonymity against malicious authority $\mathsf{Expt}^{\mathsf{CAA}}_{\beta,\mathcal{A}}(1^\lambda)$ as follows.

1. $\mathcal{A}$ outputs the challenge master public key mpk^*, a message m^* and two identities $(\mathsf{id}^*_0, \mathsf{id}^*_1)$ and the respective signing keys $(\mathsf{sk}^*_{\mathsf{id}_0}, \mathsf{sk}^*_{\mathsf{id}_1})$.
2. The challenger samples a bit $\beta \leftarrow \{0, 1\}$ and computes $\sigma_\beta \leftarrow \mathsf{Sign}(\mathsf{mpk}^*, \mathsf{sk}^*_{\mathsf{id}_\beta}, m^*)$ and returns σ_β to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a bit β^* as the output of the experiment.

We say that an adversary $\mathcal{A}$ is *admissible* if (i) $C \in \mathcal{C}$ where $C \leftarrow \mathrm{Ext}(1^\lambda, \mathsf{mpk}^*)$, and (ii) $C(m^*) = 0$.
We define the advantage of $\mathcal{A}$ in the above experiment as

$$\mathsf{Adv}^{\mathsf{CAA}}_{\mathcal{A}}(\lambda) = \left| \Pr\left[\mathsf{Expt}^{\mathsf{CAA}}_{0,\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{CAA}}_{1,\mathcal{A}}(1^\lambda) = 1\right] \right|.$$

We say that a PCGS scheme satisfies anonymity against malicious authority if there exists an (possibly inefficient) extractor Ext such that for any admissible PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CAA}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

Definition D.6 (Unconditional Client-Authority Anonymity against Malicious Authority). We say that a PCGS scheme satisfies unbounded anonymity against a malicious authority if for *any* (unbounded) admissible adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CAA}}_{\mathcal{A}}(\lambda)$ (as defined in Definition D.5) is negligible in the security parameter.

Definition D.7 (Client-Authority Unlinkability against Malicious Authority). For a PCGS scheme and an adversary $\mathcal{A}$, let us define the experiment for unlinkability against malicious authority $\mathsf{Expt}^{\mathsf{CAU}}_{\beta,\mathcal{A}}(1^\lambda)$ as follows.

1. $\mathcal{A}$ outputs the challenge master public key mpk^*, two messages (m_0^*, m_1^*) and two identities $(\mathsf{id}_0^*, \mathsf{id}_1^*)$ and the respective signing keys $(\mathsf{sk}_{\mathsf{id}_0}^*, \mathsf{sk}_{\mathsf{id}_1}^*)$.
2. The challenger computes $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_0}^*, m_0^*)$, samples a bit $\beta \leftarrow \{0,1\}$ and computes $\sigma_1 \leftarrow \mathsf{Sign}(\mathsf{mpk}^*, \mathsf{sk}_{\mathsf{id}_\beta}^*, m_1^*)$ and returns (σ_0, σ_1) to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a bit β^* as the output of the experiment.

We say that an adversary $\mathcal{A}$ is *admissible* if (i) $C \in \mathcal{C}$ where $C \leftarrow \mathrm{Ext}(1^\lambda, \mathsf{mpk}^*)$, and (ii) $C(m_0^*) = C(m_1^*) = 0$. We define the advantage of $\mathcal{A}$ in the above experiment as

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{CAU}}(\lambda) = \left|\Pr\left[\mathsf{Expt}_{0,\mathcal{A}}^{\mathsf{CAU}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}_{1,\mathcal{A}}^{\mathsf{CAU}}(1^\lambda) = 1\right]\right|.$$

We say that a PCGS scheme satisfies unlinkability against malicious authority if there exists an (possibly inefficient) extractor Ext such that for any admissible PPT adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{CAU}}(\lambda) \leq \mathsf{negl}(\lambda)$.

Definition D.8 (Unconditional Client-Authority Unlinkability against Malicious Authority). We say that a PCGS scheme satisfies unbounded unlinkability against a malicious authority if for *any* (unbounded) admissible adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{CAU}}(\lambda)$ (as defined in Definition D.7) is negligible in the security parameter.

Definition D.9 (Client-Client Anonymity). For a PCGS scheme and an adversary $\mathcal{A}$, let us define the experiment for client-client anonymity $\mathsf{Expt}_{\beta,\mathcal{A}}^{\mathsf{CCA}}(1^\lambda)$ as follows.

1. $\mathcal{A}$ outputs a circuit $C \in \mathcal{C}$.
2. The challenger generates $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, C)$. It sends the master public key mpk to $\mathcal{A}$.
3. $\mathcal{A}$ can make the H-KeyGen, C-KeyGen and Sign queries to the challenger as defined in Definition D.3. The challenger maintains the list H_{id}, H_{key}, C_{id} and S_{id} for the respective queries.
4. $\mathcal{A}$ outputs a message m^* and two user identities $(\mathsf{id}_0, \mathsf{id}_1)$. If $\mathsf{id}_0 \vee \mathsf{id}_1 \in C_{\mathsf{id}}$ then abort, else the challenger samples $\beta \leftarrow \{0,1\}$ and computes $\sigma_\beta \leftarrow \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_\beta}, m^*)$, where $\mathsf{sk}_{\mathsf{id}_\beta}$ is the signing key corresponding to id_β from the list H_{key}, and returns this to the adversary.
5. $\mathcal{A}$ outputs a bit β' as the output of the experiment.

We define the advantage of $\mathcal{A}$ in the above experiment as

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{CCA}}(\lambda) = \left|\Pr\left[\mathsf{Expt}_{0,\mathcal{A}}^{\mathsf{CCA}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}_{1,\mathcal{A}}^{\mathsf{CCA}}(1^\lambda) = 1\right]\right|.$$

We say that a PCGS scheme satisfies client-client anonymity if for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{CCA}}(\lambda) \leq \mathsf{negl}(\lambda)$.

Definition D.10 (Client-Client Unlinkability). For a PCGS scheme and an adversary $\mathcal{A}$, let us define the experiment for client-client unlinkability $\mathsf{Expt}_{\beta,\mathcal{A}}^{\mathsf{CCU}}(1^\lambda)$ as follows.

1. $\mathcal{A}$ outputs a circuit $C \in \mathcal{C}$.
2. The challenger generates $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, C)$. It sends the master public key mpk to $\mathcal{A}$.
3. $\mathcal{A}$ can make the H-KeyGen, C-KeyGen and Sign queries to the challenger as defined in Definition D.3. The challenger maintains the list H_{id}, H_{key}, C_{id} and S_{id} for the respective queries.
4. $\mathcal{A}$ outputs messages m_0, m_1 and two user identities $\mathsf{id}_0, \mathsf{id}_1$. If $\mathsf{id}_0 \in C_{\mathsf{id}}$ or $\mathsf{id}_1 \in C_{\mathsf{id}}$ then abort, else the challenger computes $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_0}, m_0)$, samples $\beta \leftarrow \{0,1\}$ and computes $\sigma_1 \leftarrow \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_\beta}, m_1)$, where $\mathsf{sk}_{\mathsf{id}_\beta}$ is the signing key corresponding to id_β from the list H_{key}. It returns (σ_0, σ_1) to $\mathcal{A}$.

5. $\mathcal{A}$ outputs a bit β' as the output of the experiment.

We define the advantage of $\mathcal{A}$ in the above experiment as

$$\mathsf{Adv}^{\mathsf{CCU}}_{\mathcal{A}}(\lambda) = \left|\Pr\left[\mathsf{Expt}^{\mathsf{CCU}}_{0,\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{CCU}}_{1,\mathcal{A}}(1^\lambda) = 1\right]\right|.$$

We say that a PCGS scheme satisfies client-client unlinkability if for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CCU}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

D.2 Construction

Our construction of PCGS follows from the compiler provided by [BGJP23] and is secure in ROM. As we will see below, we can use our LWE based sPCE scheme achieving unconditional security together with dual-mode NIZK to obtain unconditional anonymity against malicious authority.

Building Blocks We use the following ingredients for our construction.

1. A sPCE scheme $\mathsf{sPCE} = \mathsf{sPCE}.(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$. This can be instantiated from a variety of assumptions as described in Theorems 3.15, 3.18 and 3.32.
2. A dual-mode NIZK proof system $\mathsf{ZK} = (\mathsf{ZK.Hsetup}, \mathsf{ZK.Bsetup}, \mathsf{ZK.Prove}, \mathsf{ZK.Verify})$ satisfying statistical zero-knowledge in. This can be instantiated from LWE (Corollary A.11).
3. A one-way relation $\mathcal{R} = (\mathcal{R}.\mathsf{Gen}, \mathcal{R}.\mathsf{Sample})$ for a set of tuples $\mathcal{R}$.
4. A digital signature scheme $\mathcal{S} = (\mathcal{S}.\mathsf{Setup}, \mathcal{S}.\mathsf{Sign}, \mathcal{S}.\mathsf{Verify})$.
5. A random oracle H.

Construction. We now describe the construction of PCGS scheme, adapted from [BGJP23].

$\mathsf{Setup}(1^\lambda, C) \to (\mathsf{mpk}, \mathsf{msk})$. The setup algorithm does the following.

- Generate $\mathcal{R}.\mathsf{pp} \leftarrow \mathcal{R}.\mathsf{Gen}(1^\lambda)$ and $(\mathcal{S}.\mathsf{vk}, \mathcal{S}.\mathsf{sk}) \leftarrow \mathcal{S}.\mathsf{Gen}(1^\lambda)$.
- Define circuit $U[C]$ as follows: On input (m, id), $U[C](m, \mathsf{id}) = \mathsf{id}$ if $C(m) = 1$, $\perp$ otherwise. Compute $(\mathsf{sPCE.pk}, \mathsf{sPCE.sk}) \leftarrow \mathsf{sPCE.Gen}(1^{\breve{}}, \mathsf{U}[\mathsf{C}])$.
- Output $\mathsf{mpk} = (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk})$ and $\mathsf{msk} = (\mathsf{sPCE.sk}, \mathcal{S}.\mathsf{sk})$.

$\mathsf{KeyGen}\langle \mathsf{GM}(\mathsf{msk}), U\rangle \to (\mathsf{id}, \mathsf{sk}_{\mathsf{id}})$. The key generation protocol is as follows.

- The user U samples $s \leftarrow \{0,1\}^r$ and computes an instance-witness pair $(\mathsf{id}, w) = \mathcal{R}.\mathsf{Sample}(\mathcal{R}.\mathsf{pp}; s)$. It sends id to the group manager GM.
- The group manager parses $\mathsf{msk} = (\mathsf{sPCE.sk}, \mathcal{S}.\mathsf{sk})$ and computes $\sigma_{\mathsf{id}} \leftarrow \mathcal{S}.\mathsf{Sign}(\mathcal{S}.\mathsf{sk}, \mathsf{id})$ and returns it to the user.
- The user sets $\mathsf{sk}_{\mathsf{id}} = (s, \sigma_{\mathsf{id}})$.

$\mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, m) \to \sigma$. The signing algorithm does the following.

- Parse $\mathsf{mpk} = (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk})$ and $\mathsf{sk}_{\mathsf{id}} = (s, \sigma_{\mathsf{id}})$.
- Compute $(\mathsf{id}, w) = \mathcal{R}.\mathsf{Sample}(\mathcal{R}.\mathsf{pp}; s)$.
- Sample $r \leftarrow \{0,1\}^\lambda$ and compute $\mathsf{sPCE.ct} = \mathsf{sPCE.Enc}(\mathsf{sPCE.pk}, \mathsf{m}, \mathsf{id}; \mathsf{r})$.
- Let $\mathsf{crs} \leftarrow H(m, \mathsf{sPCE.ct})$, and compute $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}, \mathsf{sPCE.ct}), (\mathsf{id}, \mathsf{s}, \mathsf{w}, \text{œ}_{\mathsf{id}}, \mathsf{r}))$ for the relation that checks that:
  1. $\mathsf{sPCE.ct} = \mathsf{sPCE.Enc}(\mathsf{sPCE.pk}, \mathsf{m}, \mathsf{id}; \mathsf{r})$.

2. $(\mathsf{id}, w) = \mathcal{R}.\mathsf{Sample}(\mathcal{R}.\mathsf{pp}; s)$.
3. $\mathcal{S}.\mathsf{Verify}(\mathcal{S}.\mathsf{vk}, \mathsf{id}, \sigma_{\mathsf{id}})$

- Output $\sigma = (\mathsf{sPCE.ct}, \mathsf{crs}, \text{ß})$.

$\mathsf{Verify}(\mathsf{mpk}, m, \sigma) \to \{0, 1\}$. The verification algorithm does the following.

- Parse $\mathsf{mpk} = (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk})$ and $\sigma = (\mathsf{sPCE.ct}, \mathsf{crs}, \text{ß})$.
- Check $H(m, \mathsf{sPCE.ct}) = \mathsf{crs}$. If true, output $\mathsf{ZK.Verify}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}, \mathsf{sPCE.ct}), \text{ß})$.

$\mathsf{Open}(\mathsf{msk}, \sigma) \to \{\mathsf{id}, \bot\}$. The open algorithm does the following.

- Parse $\mathsf{msk} = (\mathsf{sPCE.sk}, \mathcal{S}.\mathsf{sk})$ and $\sigma = (\mathsf{sPCE.ct}, \mathsf{crs}, \text{ß})$.
- Output $\mathsf{sPCE.Dec}(\mathsf{sPCE.sk}, \mathsf{sPCE.ct})$.

Correctness. We now show that the above construction is correct via the following theorem.

Theorem D.11. Suppose that $\mathcal{S}$ is a correct digital signature scheme and ZK satisfies completeness (Definition A.1). Then the above construction of PCGS satisfies correctness (Definition D.1).

Proof. We observe that for $\sigma = (\mathsf{sPCE.ct}, \mathsf{crs}, \text{ß})$, we have $\mathsf{sPCE.ct} = \mathsf{sPCE.Enc}(\mathsf{sPCE.pk}, \mathsf{m}, \mathsf{id}; \mathsf{r})$, $\mathsf{crs} = H(m, \mathsf{sPCE.ct})$ and $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}, \mathsf{sPCE.ct}), (\mathsf{id}, \mathsf{s}, \mathsf{w}, \text{œ}_{\mathsf{id}}, \mathsf{r}))$, where $(\mathsf{id}, w) = \mathcal{R}.\mathsf{Sample}(\mathcal{R}.\mathsf{pp}; s)$ and $\sigma_{\mathsf{id}} \leftarrow \mathcal{S}.\mathsf{Sign}(\mathcal{S}.\mathsf{sk}, \mathsf{id})$.
By the correctness of the signature scheme, we have with all but negligible probability, $\mathcal{S}.\mathsf{Verify}(\mathcal{S}.\mathsf{vk}, \mathsf{id}, \sigma_{\mathsf{id}}) = 1$. Hence, the completeness of ZK scheme implies $\mathsf{ZK.Verify}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}, \mathsf{sPCE.ct}), \text{ß}) = 1$ with all but negligible probability.
So, $\mathsf{Verify}(\mathsf{mpk}, m, \sigma) = (\mathsf{crs} = H(m, \mathsf{sPCE.ct})) \wedge (\mathsf{ZK.Verify}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}, \mathsf{sPCE.ct}), \text{ß})) = 1$ with all but negligible probability . Hence the above construction satisfies correctness. □

Instantiation Below we describe the properties inherited by the resulting PCGS scheme for each instantiation of sPCE scheme that we construct.

1. If we use sPCE scheme as in Theorem 3.15, we achieve anonymity and unlinkability against a malicious authority for general constraints. For constraints in NC^1, we can achieve unconditional anonymity and unlinkability against a malicious authority.

2. If we use sPCE scheme as in Theorem 3.18, we can achieve unconditional anonymity and unlinkability against a malicious authority for general constraints.

3. If we use the succinct sPCE scheme as in Theorem 3.32, we achieve a succinct PCGS scheme achieving security guarantees against a semi-malicious PPT authority.

Next, we prove the security properties of the above PCGS scheme.

Constraint-Hiding. This follows immediately from the constraint-hiding property of the underlying sPCE scheme.

Traceability. We show that the above scheme satisfies traceability using the following theorem.

Theorem D.12. Suppose that sPCE scheme satisfies perfect correctness (Definition 3.1) and security against outsiders (Definition 3.7), ZK satisfies statistical zero-knowledge in the hiding mode (Definition A.3) and knowledge extraction in the binding mode (Definition A.5), one way relation $\mathcal{R}$ is secure (Definition A.13) and the signature scheme $\mathcal{S}$ satisfies EUF-CMA security. Then the above construction of PCGS satisfies traceability (Definition D.3) in the random oracle model.

Proof. The proof is identical to the proof of traceability in Theorem 4, [BGJP23] and is hence omitted. □

Unframeability. We show that the above scheme satisfies unframeability using the following theorem.

Theorem D.13. Suppose that sPCE scheme satisfies perfect correctness (Definition 3.1) and security against outsiders (Definition 3.7), ZK satisfies statistical zero-knowledge with common random string in the hiding mode (Definition A.3) and knowledge extraction in the binding mode (Definition A.5), and the one way relation $\mathcal{R}$ is secure (Definition A.13). Then the above construction of PCGS satisfies unframeability (Definition D.4) in the random oracle model.

Proof. The proof is identical to the proof of unframeability in Theorem 4, [BGJP23] and is hence omitted. □

Client-Authority Anonymity against Malicious Authority.

Theorem D.14. Suppose that sPCE scheme satisfies (computational/unconditional) security against malicious authority (Definition 3.6) and ZK satisfies statistical zero-knowledge with common random string in the hiding mode (Definition A.3). Then the above construction of PCGS satisfies (computational/unconditional) client-authority anonymity against a malicious authority (Definition D.6) in the random oracle model.

Proof. Recall that in the client-authority anonymity against a malicious authority, we want to show that

$$\mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_0}, m) \approx_s \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_1}, m)$$

where $C(m) = 0$, for the circuit C associated with the, possibly malformed, master public key mpk.
Here we let the extractor Ext of PCGS scheme to be the extractor of the underlying sPCE scheme, say sPCE.Ext, which is secure against a malicious authority. The proof proceeds via the following sequence of hybrid games between the challenger and an unbounded adversary $\mathcal{A}$.

Hyb_0. This is the real world with $\beta = 0$, i.e., the challenge signature is computed using the signing key $\mathsf{sk}_{\mathsf{id}_0}$ associated with the identity id_0. We write the complete game here to set up the notations and easy reference in later hybrids.

1. $\mathcal{A}$ outputs the challenge master public key mpk, a message m and two identities $(\mathsf{id}_0, \mathsf{id}_1)$ and the respective signing keys $(\mathsf{sk}_{\mathsf{id}_0}, \mathsf{sk}_{\mathsf{id}_1})$.
2. The challenger parses $\mathsf{mpk} = (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk})$ and $\mathsf{sk}_{\mathsf{id}_0} = (s_0, \sigma_{\mathsf{id}_0})$. It computes $\mathsf{sPCE.ct} = \mathsf{sPCE.Enc}(\mathsf{sPCE.pk}, \mathsf{m}, \mathsf{id}_0; \mathsf{r}_0)$, $\mathsf{crs} \leftarrow H(m, \mathsf{sPCE.ct})$ and proof $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}, \mathsf{sPCE.ct}), (\mathsf{id}_0, \mathsf{s}_0, \mathsf{w}_0, œ_{\mathsf{id}_0}, \mathsf{r}_0))$ as in the Sign algorithm of the construction. It returns $\sigma = (\mathsf{sPCE.ct}, \mathsf{crs}, ß)$ to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a bit β as the output of the experiment.

Hyb_1. This hybrid is same as the previous hybrid except that the random oracle H is lazily sampled, i.e., the challenger generates (crs, π) using ZK.Sim on the instance $(\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}, \mathsf{sPCE.ct})$ and sets $H(m, \mathsf{sPCE.ct}) = \mathsf{crs}$.
This hybrid is statistically indistinguishable from previous one due to statistical zero-knowledge in hiding mode of the underlying ZK scheme.

Hyb_2. This hybrid is same as the previous hybrid except that the challenger computes sPCE.ct differently as $\mathsf{sPCE.ct} \leftarrow \mathsf{sPCE.Enc}(\mathsf{sPCE.pk}, \mathsf{m}, \mathsf{id}_1)$.
This hybrid is statistically indistinguishable from previous one due to the unconditional SIM security against malicious authority of the underlying sPCE scheme.

Hyb_3. This hybrid is same as the previous hybrid except that the challenger generates (crs, π) differently . It computes $\mathsf{crs} \leftarrow H(m, \mathsf{sPCE.ct})$ and the proof $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}, \mathsf{sPCE.ct}), (\mathsf{id}_1, \mathsf{s}_1, \mathsf{w}_1, œ_{\mathsf{id}_1}, \mathsf{r}_1))$ as in the Sign algorithm of the construction. This is the real world with $\beta = 1$.
This hybrid is statistically indistinguishable from previous one due to statistical zero-knowledge in hiding mode of the underlying ZK scheme.

□

Client-Authority Unlinkability against Malicious Authority.

Theorem D.15. Suppose that sPCE scheme satisfies (computational/unconditional) security against malicious authority (Definition 3.6) and ZK satisfies statistical zero-knowledge with common random string in the hiding mode (Definition A.3). Then the above construction of PCGS satisfies (computational/unconditional) client-authority unlinkability against a malicious authority (Definition D.7) in the random oracle model.

Proof. Recall that in the client-authority unlinkability against a malicious authority, we want to show that

$$\mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_0}, m_0) \approx_s \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_b}, m_1)$$

where $b \in \{0, 1\}$ and $C(m_0) = 0 = C(m_1)$, for the circuit C associated with the, possibly malformed, master public key mpk.

Here we let the extractor Ext of PCGS scheme to be the extractor of the underlying sPCE scheme, say $\mathsf{sPCE.Ext}$, which is secure against a malicious authority. We prove the above for $b = 0$ and $b = 1$ separately.

1. $\mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_0}, m_0) \approx_s \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_0}, m_1)$.
   The proof proceeds via the following sequence of hybrid games between the challenger and an unbounded adversary $\mathcal{A}$.

   Hyb_0. This is the real world with $\beta = 0$, i.e., the challenge signature is computed for message m_0 using the signing key $\mathsf{sk}_{\mathsf{id}_0}$ associated with the identity id_0. We write the complete game here to set up the notations and easy reference in later hybrids.

   (a) $\mathcal{A}$ outputs the challenge master public key mpk, two messages m_0, m_1 and two identities $(\mathsf{id}_0, \mathsf{id}_1)$ and the respective signing keys $(\mathsf{sk}_{\mathsf{id}_0}, \mathsf{sk}_{\mathsf{id}_1})$.

   (b) The challenger parses $\mathsf{mpk} = (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk})$ and $\mathsf{sk}_{\mathsf{id}_0} = (s_0, \sigma_{\mathsf{id}_0})$.
   It computes $\mathsf{sPCE.ct} = \mathsf{sPCE.Enc}(\mathsf{sPCE.pk}, \mathsf{m}_0, \mathsf{id}_0; \mathsf{r}_0)$, $\mathsf{crs} \leftarrow H(m_0, \mathsf{sPCE.ct})$ and proof $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}_0, \mathsf{sPCE.ct}), (\mathsf{id}_0, \mathsf{s}_0, \mathsf{w}_0, \mathsf{œ}_{\mathsf{id}_0}, \mathsf{r}_0))$ as in the Sign algorithm of the construction. It returns $\sigma = (\mathsf{sPCE.ct}, \mathsf{crs}, \mathsf{ß})$ to $\mathcal{A}$.

   (c) $\mathcal{A}$ outputs a bit β as the output of the experiment.

   Hyb_1. This hybrid is same as the previous hybrid except that the random oracle H is lazily sampled, i.e., the challenger generates (crs, π) using $\mathsf{ZK.Sim}$ on the instance $(\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}_0, \mathsf{sPCE.ct})$ and sets $H(m_0, \mathsf{sPCE.ct}) = \mathsf{crs}$. We note that $\mathsf{Hyb}_0 \approx_s \mathsf{Hyb}_1$ using the statistical zero-knowledge with common random string of the underlying ZK scheme.

   Hyb_2. This hybrid is same as the previous hybrid except that the challenger computes $\mathsf{sPCE.ct}$ differently as $\mathsf{sPCE.ct} \leftarrow \mathsf{sPCE.Enc}(\mathsf{sPCE.pk}, \mathsf{m}_1, \mathsf{id}_0)$ and the corresponding proof using ZK simulator as $(\mathsf{crs}, \pi) \leftarrow \mathsf{ZK.Sim}(\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}_1, \mathsf{sPCE.ct})$. $\mathsf{Hyb}_1 \approx_s \mathsf{Hyb}_2$ using the unconditional security of the underlying sPCE scheme against a malicious authority

   Hyb_3. This hybrid is same as the previous hybrid except that the challenger generates (crs, π) differently. It computes $\mathsf{crs} \leftarrow H(m_1, \mathsf{sPCE.ct})$ and the proof $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}_1, \mathsf{sPCE.ct}), (\mathsf{id}_0, \mathsf{s}_0, \mathsf{w}_0, \mathsf{œ}_{\mathsf{id}_0}, \mathsf{r}_0))$ as in the Sign algorithm of the construction. This is the real world with $\beta = 1$. We note that $\mathsf{Hyb}_2 \approx_s \mathsf{Hyb}_3$ using the statistical zero-knowledge with common random string of the underlying ZK scheme.

2. $\mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_0}, m_0) \approx_s \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}_1}, m_1)$. The indistinguishability of these two distribution follows from the similar sequence of hybrids as above except the following changes:

   - In Hyb_2 we generate $\mathsf{sPCE.ct}$ differently as $\mathsf{sPCE.ct} \leftarrow \mathsf{sPCE.Enc}(\mathsf{sPCE.pk}, \mathsf{m}_1, \mathsf{id}_1)$.
   - In Hyb_3 we generate the proof as $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, (\mathcal{R}.\mathsf{pp}, \mathsf{sPCE.pk}, \mathcal{S}.\mathsf{vk}, \mathsf{m}_1, \mathsf{sPCE.ct}), (\mathsf{id}_1, \mathsf{s}_1, \mathsf{w}_1, \mathsf{œ}_{\mathsf{id}_1}, \mathsf{r}_1))$.

□

Client-Client Anonymity and Unlinkability.

Theorem D.16. Suppose that sPCE scheme satisfies security against outsiders (Definition 3.7) and ZK satisfies statistical zero-knowledge in the hiding mode (Definition A.3). Then the above construction of PCGS satisfies client-client anonymity (Definition D.9) and client-client unlinkability (Definition D.10) in the random oracle model.

Proof. The proof is identical to the proof of client-client anonymity and client-client unlinkability (Definition D.10) in Theorem 4, [BGJP23] and is hence omitted. □

E Pre-Constrained Input Obfuscation

We introduce the notion of pre-constrained input obfuscation (PCIO) in this section. This notion is a relaxation of virtual black-box obfuscation or indistinguishability obfuscation. A notable difference is that we can compute outputs of obfuscated circuits only for pre-determined polynomially many inputs.

E.1 Definition

A pre-constrained input obfuscation for circuit family $\mathcal{C} = \{C : \mathcal{X} \to \mathcal{Y}\}$ with input space $\mathcal{X}$ and output space $\mathcal{Y}$ consists of three algorithms $(\mathsf{Setup}, \mathcal{O}, \mathsf{Eval})$ defined as follows.

$\mathsf{Setup}(1^\lambda, \mathcal{S}) \to (\mathsf{pk}, \mathsf{ek})$. The setup algorithm takes as input a set of input $\mathcal{S}$ where $\mathcal{S} \subset \mathcal{X}$ and $|\mathcal{S}| = \mathrm{poly}(\lambda)$, and outputs a public key pk and evaluation key ek.

$\mathcal{O}(\mathsf{pk}, C) \to \widetilde{C}$. The obfuscation algorithm takes as input the public key pk and a circuit C, and outputs an obfuscated circuit $\widetilde{C}$.

$\mathsf{Eval}(\mathsf{ek}, \widetilde{C}, x) \to y$. The evaluation algorithm takes as input the evaluation key ek, obfuscated circuit $\widetilde{C}$, and an input $x \in \mathcal{X}$, and outputs an $y \in \mathcal{Y}$.

Here, pk is reusable for multiple C.

Definition E.1 (Correctness). A PCIO scheme is correct if for any $\mathcal{S} \subset \mathcal{X}$, $|\mathcal{S}| = \mathrm{poly}(\lambda)$, and $(\mathsf{pk}, \mathsf{ek}) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{S})$, the following holds

1. $\forall x \in \mathcal{S}, \Pr[\mathsf{Eval}(\mathsf{ek}, \mathcal{O}(\mathsf{pk}, C), x) = C(x)] \geq 1 - \mathsf{negl}(\lambda)$.
2. $\forall x \in \mathcal{X} \setminus \mathcal{S}, \Pr[\mathsf{Eval}(\mathsf{ek}, \mathcal{O}(\mathsf{pk}, C), x) = \bot] \geq 1 - \mathsf{negl}(\lambda)$.

Definition E.2 (Input-Set-Hiding). A PCIO scheme satisfies input-set-hiding security if for any PPT adversary $\mathcal{A}$, the following holds

$$\Pr\left[\ \mathcal{A}(\mathsf{pk}_\beta) = \beta : \begin{array}{l} (\mathcal{S}_0, \mathcal{S}_1) \leftarrow \mathcal{A}; \\ \beta \leftarrow \{0,1\}; (\mathsf{pk}_\beta, \mathsf{ek}_\beta) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{S}_\beta) \end{array}\ \right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ is admissible if $\mathcal{S}_b \subset \mathcal{X}$ and $\mathcal{S}_b = \mathrm{poly}(\lambda)$ for $b \in \{0,1\}$.

Definition E.3 (Virtual Black-Box Security against Malicious Authority). For a PPT adversary $\mathcal{A}$ and a PPT simulator Sim, consider the following experiment $\mathsf{Exp}^{\mathsf{PC\text{-}VBB}}_{\beta,\mathcal{A},\mathsf{Sim}}(1^\lambda)$.

1. $\mathcal{A}$ outputs the public key pk and circuit C.
2. On input (pk, C), the challenger samples a random bit β uniformly and generates $\widetilde{C} \leftarrow \mathcal{O}(\mathsf{pk}, C)$ if $\beta = 0$. Otherwise, the challenger generates $\widetilde{C} \leftarrow \mathsf{Sim}(1^\lambda, 1^{|C|}, \mathsf{pk}, C(x_1), \ldots, C(x_q))$ where $(x_1, \ldots, x_q) \leftarrow \mathsf{Ext}(1^\lambda, \mathsf{pk})$ and Ext is an (possibly inefficient) extractor. It returns $\widetilde{C}$ to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a guess bit β' as the output of the experiment.

We say that a PCIO scheme satisfies the virtual black-box security against malicious authority if there exists an (possibly inefficient) extractor Ext such that for any PPT adversary $\mathcal{A}$,

$$\left|\Pr\left[\mathsf{Exp}^{\mathsf{PC\text{-}VBB}}_{0,\mathcal{A},\mathsf{Sim}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{PC\text{-}VBB}}_{1,\mathcal{A},\mathsf{Sim}}(1^\lambda) = 1\right]\right| = \mathsf{negl}(\lambda).$$

Definition E.4 (Indistinguishability against Malicious Authority). A PCIO scheme satisfies input-set-hiding security if there exists an admissible (possibly inefficient) extractor Ext such that for any PPT and admissible adversary $\mathcal{A}$, the following holds

$$\Pr\left[\ \mathcal{A}(\mathsf{ct}_\beta) = \beta : \begin{array}{l} \mathsf{pk}, (C_0, C_1) \leftarrow \mathcal{A}; \\ \beta \leftarrow \{0,1\}; \mathsf{ct}_\beta \leftarrow \mathsf{Enc}(\mathsf{pk}, C_\beta) \end{array}\ \right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ is admissible if (i) $\mathcal{S} \subset \mathcal{X}$ and $|\mathcal{S}| = \mathrm{poly}(\lambda)$ where $\mathcal{S} \leftarrow \mathrm{Ext}(1^\lambda, \mathsf{pk})$, and (ii) $C_b \in \mathcal{C}$ for $b \in \{0,1\}$, $|C_0| = |C_1|$, and $\forall x \in \mathcal{S}, C_0(x) = C_1(x)$. We say that Ext is admissible if for every $\mathcal{S} = (x_1, \ldots, x_Q)$ and randomness r, we have that $(x_1, \ldots, x_Q) = (x'_1, \ldots, x'_Q)$, where (i) $(\mathsf{pk}, \mathsf{ek}_{x_1}, \ldots, \mathsf{ek}_{x_Q}) \leftarrow \mathsf{Setup}(1^\lambda, \{x_1, \ldots, x_Q\}; r)$ and (ii) $\mathcal{S}' = (x'_1, \ldots, x'_Q) \leftarrow \mathrm{Ext}(1^\lambda, \mathsf{pk})$.
In addition, if $\mathcal{A}$ is unbounded, we say that PCIO is unconditional IND secure.

Definition E.5 (Succinct Public Key). We say that a PCIO scheme has succinct public keys when the size of the public key is sublinear in $\mathcal{S}$, that is $|\mathsf{pk}| = O(|\mathcal{S}|^{1-\gamma})$ for some $0 < \gamma < 1$ where $(\mathsf{pk}, \mathsf{ek}) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{S})$.

E.2 Construction and Implication

E.2.1 From sPCE to PCIO.

We construct a PCIO scheme, for circuit family $\mathcal{C} = \{C : \mathcal{X} \to \mathcal{Y}\}$, from a $\mathsf{sPCE} = (\mathsf{sPCE.Setup}, \mathsf{sPCE.Enc}, \mathsf{sPCE.Dec})$ scheme as defined in Section 3.

$\mathsf{Setup}(1^\lambda, \mathcal{S}) \to (\mathsf{pk}, \mathsf{ek})$**.** The setup algorithm does the following.

- Parse $\mathcal{S} = (x_1, \ldots, x_Q)$.
- Let $U[x]$ be a universal circuit, which on input a circuit C, outputs $C(x)$.
- Run $(\mathsf{fe.pk}, \mathsf{fe.sk}_1, \ldots, \mathsf{fe.sk}_Q) \leftarrow \mathsf{sPCE.Setup}(1\check{}, \mathsf{U}[\mathsf{x}_1], \ldots, \mathsf{U}[\mathsf{x}_\mathsf{Q}])$.
- Output $\mathsf{pk} := \mathsf{fe.pk}$ and $\mathsf{ek} := (\mathsf{fe.sk}_1, \ldots, \mathsf{fe.sk}_Q)$. (We assume that $\mathsf{fe.sk}_i$ includes x_i.)

$\mathcal{O}(\mathsf{pk}, C) \to \widetilde{C}$**.** The obfuscating algorithm does the following.

- Parse $\mathsf{pk} = \mathsf{fe.pk}$ and run $\mathsf{fe.ct} \leftarrow \mathsf{sPCE.Enc}(\mathsf{fe.pk}, \mathsf{C})$.
- Output $\widetilde{C} := \mathsf{fe.ct}$.

$\mathsf{Eval}(\mathsf{ek}, \widetilde{C}, x) \to y$**.** The evaluation algorithm does the following.

- Parse $\mathsf{ek} = (\mathsf{fe.sk}_1, \ldots, \mathsf{fe.sk}_Q)$ and $\widetilde{C} = \mathsf{fe.ct}$.
- Find $\mathsf{fe.sk}_i$ corresponding to x. If there is no such sk_i, output $\perp$.
- Otherwise, run and output $y \leftarrow \mathsf{sPCE.Dec}(\mathsf{fe.sk_i}, \mathsf{fe.ct})$.

Correctness. We show the correctness of our PCIO scheme via the following theorem.

Theorem E.6. *Suppose* sPCE *satisfies correctness as defined in Definition 3.1. Then the above construction satisfies correctness as defined in Definition E.1.*

Proof. From the correctness of sPCE, it holds that $y = U[x_i](C)$ where $y = \mathsf{sPCE.Dec}(\mathsf{fe.sk_i}, \mathsf{fe.ct})$. By the definition of $U[x]$, it holds that $y = C(x_i)$. Next, if $x \in \mathcal{X} \setminus \mathcal{S}$, there is no sk_i corresponding to x in ek. In this case, the decryption algorithm outputs $\perp$. Thus, the theorem holds. □

Input-set-hiding. We show that the above construction satisfies the input-set-hiding property.

Theorem E.7. Suppose sPCE satisfies the function-hiding property as defined in Definition 3.2. Then the above construction satisfies input-set-hiding as defined in Definition E.2.

Proof. We construct an adversary $\mathcal{B}$ of the function-hiding game for sPCE by using an adversary $\mathcal{A}$ of the input-set-hiding game for PCIO. $\mathcal{B}$ works as follows.

1. When $\mathcal{A}$ sends $\mathcal{S}_0 = (x_1^{(0)}, \ldots, x_Q^{(0)})$ and $\mathcal{S}_1 = (x_1^{(1)}, \ldots, x_Q^{(1)})$, $\mathcal{B}$ sets $f_i^{(b)} := U[x_i^{(b)}]$ for $b \in \{0, 1\}$ and sends $:= (f_1^{(0)}, f_Q^{(1)}), \ldots, (f_Q^{(0)}, f_Q^{(1)})$ to its challenger.

2. $\mathcal{B}$ receives $\mathsf{fe.pk}$ from its challenger and it sends $\mathsf{pk} := \mathsf{fe.pk}$ to $\mathcal{A}$.

3. $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

$\mathcal{B}$ perfectly simulates the input-set-hiding game for $\mathcal{A}$. Thus, if $\mathcal{A}$ breaks the input-set-hiding property, $\mathcal{B}$ also breaks the function-hiding game. This completes the proof. □

Security against malicious authority. We show that the above construction satisfies security against malicious authority.

Theorem E.8. Suppose sPCE satisfies security SIM security (resp. indistinguishability) against malicious authority. Then the above construction satisfies VBB security (resp. indistinguishability) against malicious authority. In addition, if sPCE has unconditional (SIM or IND) security, the above construction also has unconditional (VBB or IND) security.

We focus on the proof for the simulation-based security. The proof for the indistinguishability-based definition is similar to the simulation-based one, and we omit it.

Proof. We define $\mathrm{Ext}(1^\lambda, \mathsf{pk})$ as follows.

- Parse $\mathsf{pk} = \mathsf{fe.pk}$ and run $(f_1, \ldots, f_Q) \leftarrow \mathsf{sPCE.Ext}(1^{\breve{}}, \mathsf{fe.pk})$.
- Interpret f_i as a universal circuit $U[x_i']$ for all $i \in [Q]$.
- Output $(x_1', \ldots, x_Q')$.

We also define $\mathsf{Sim}(1^\lambda, 1^{|C|}, \mathsf{pk}, C(x_1'), \ldots, C(x_Q'))$ as follows.

- Parse $\mathsf{pk} = \mathsf{fe.pk}$ and run $\mathsf{fe.ct}_1 \leftarrow \mathsf{sPCE.Sim}(\mathsf{fe.pk}, 1^{|\mathsf{C}|}, \mathsf{C}(\mathsf{x}_1'), \ldots, \mathsf{C}(\mathsf{x}_\mathsf{Q}'))$.
- Output $\widetilde{C} := \mathsf{fe.ct}_1$.

We construct an adversary $\mathcal{B}$ of the SIM security for sPCE by using an adversary $\mathcal{A}$ of the VBB security for PCIO. $\mathcal{B}$ works as follows.

1. When $\mathcal{A}$ sends pk and C, $\mathcal{B}$ sends $\mathsf{fe.pk} := \mathsf{pk}$ and $x := C$ to its challenger.

2. $\mathcal{B}$ receives a challenge ciphertext $\mathsf{fe.ct}$ from its challenger and it sends $\widetilde{C} := \mathsf{fe.ct}$ to $\mathcal{A}$.

3. $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

If $\beta = 0$, $\mathcal{B}$ receives $\mathsf{fe.ct}_0 \leftarrow \mathsf{sPCE.Enc}(\mathsf{fe.pk}, \mathsf{C})$. If $\beta = 1$, $\mathcal{B}$ receives $\mathsf{fe.ct}_1 \leftarrow \mathsf{sPCE.Sim}(\mathsf{fe.pk}, 1^{|\mathsf{C}|}, \mathsf{U}[\mathsf{x}_1'](\mathsf{C}), \ldots, \mathsf{U}[\mathsf{x}_\mathsf{Q}'](\mathsf{C}))$ where $(U[x_1'], \ldots . U[x_Q']) \leftarrow \mathsf{sPCE.Ext}(1^{\breve{}}, \mathsf{FE.pk})$. So, $\mathcal{B}$ perfectly simulates the VBB security game for $\mathcal{A}$. Thus, if $\mathcal{A}$ breaks the VBB security, $\mathcal{B}$ also breaks the SIM security. This completes the proof. □

From PCIO to sPCE. It is easy to see that we can construct a sPCE scheme from a PCIO scheme. Let $\mathsf{PCIO} := \mathsf{PCIO}.(\mathsf{Setup}, \mathcal{O}, \mathsf{Eval})$ be a PCIO scheme.

$\mathsf{Setup}(1^\lambda, f_1, \ldots, f_Q) \to (\mathsf{pk}, \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_Q})$**.** The setup algorithm does the following.

- Set $\mathcal{S} := (f_1, \ldots, f_Q)$ and run $(\mathsf{PCIO.pk}, \mathsf{PCIO.ek}) \leftarrow \mathsf{PCIO.Setup}(1^\lambda, \mathcal{S})$.
- Output $\mathsf{pk} := \mathsf{PCIO.pk}$ and $\mathsf{sk}_{f_i} := (\mathsf{PCIO.ek}, f_i)$ for all $i \in [Q]$.

$\mathsf{Enc}(\mathsf{pk}, x) \to \mathsf{ct}$**.** The encryption algorithm does the following.

- Parse $\mathsf{pk} = \mathsf{PCIO.pk}$ and run $\widetilde{U} \leftarrow \mathsf{PCIO}.\mathcal{O}(\mathsf{PCIO.pk}, U[x])$ where $U[x]$ is a universal circuit that takes as input f, and outputs $f(x)$.
- Output $\mathsf{ct} := \widetilde{U}$.

$\mathsf{Dec}(\mathsf{sk}_{f_i}, \mathsf{ct}) \to y$**.** The decryption algorithm does the following.

- Parse $\mathsf{sk}_{f_i} = (\mathsf{PCIO.ek}, f_i)$ and $\mathsf{ct} = \widetilde{U}$.
- Compute and output $\mathsf{PCIO.Eval}(\mathsf{PCIO.ek}, \widetilde{U}, f_i)$.

Correctness. Correctness of the above scheme follows from the correctness of the sPCE scheme.

Theorem E.9. Suppose PCIO satisfies correctness as defined in Definition E.1. Then the above construction satisfies correctness as defined in Definition 3.1.

Input-set-hiding. The input-set-hiding property of the above scheme follows from the function-hiding of the sPCE scheme.

Theorem E.10. Suppose PCIO satisfies input-set-hiding as defined in Definition E.2. Then the above construction satisfies the function-hiding property as defined in Definition 3.2.

Security against malicious authority. This follows from the security against malicious authority of the sPCE scheme.

Theorem E.11. Suppose PCIO satisfies VBB security (resp. indistinguishability) against malicious authority. Then the above construction satisfies security SIM security (resp. indistinguishability) against malicious authority. In addition, if PCIO has unconditional (VBB or IND) security, the above construction also has unconditional (SIM or IND) security.

The proofs of Theorems E.9 to E.11 are almost the same as those of Theorems E.6 to E.8, so we omit them.