Collusion-Resistant Asymmetric Anamorphic Encryption: Framework, Generic Construction, and Concrete Instantiations

Zhikang Xie[1], Rupeng Yang[2], Man Ho Au[3], Zuoxia Yu[2], and Willy Susilo[2]

[1] The University of Hong Kong, Hong Kong, China
zkxiecs@gmail.com
[2] University of Wollongong, Wollongong, Australia
{rupengy,zyu,wsusilo}@uow.edu.au
[3] The Hong Kong Polytechnic University, Hong Kong, China
mhaau@polyu.edu.hk

Abstract. (Receiver) anamorphic encryption enables covert communication through innocent-looking ciphertexts, even against a dictator that compels users to surrender their private decryption keys. In this work, we study the asymmetric form, where the covert channel operates in a manner analogous to PKE. However, a generic construction achieving collusion-resistance (where the dictator can corrupt a sender to additionally obtain the sender double key) under only mild restrictions on the underlying PKE remains lacking.

To address the problem, we develop a general framework for constructing collusion-resistant asymmetric anamorphic encryptions from a new cryptographic abstraction called witness PRF for PKE. The framework works in the anamorphic extension setting, which enables covert communication from (already deployed) public keys. Importantly, our security model addresses the shortcomings of prior collusion models.

The first instantiation of our framework yields a generic asymmetric anamorphic extension for any PKE scheme with high min-entropy ciphertexts. In contrast to prior generic constructions of fully asymmetric anamorphic encryption, which rely on indistinguishability obfuscation and do not consider collusion, our construction achieves collusion-resistance under assumptions believed to be weaker. We also give practical instantiations for widely deployed PKE schemes, including ElGamal, Regev, Linear, and CSIDH-based encryptions, without relying on heavy cryptographic mechanisms. Notably, some of these extend the range of PKE schemes known to admit practical asymmetric anamorphism.

Keywords: Asymmetric anamorphic encryption · Anamorphic extension · Indistinguishability against collusion · Witness PRF for PKE.

1 Introduction

(Receiver) Anamorphic Encryption (AE) was introduced by [26] to technically demonstrate that the efforts of adversarial authorities to inspect private communications by compelling users to surrender their private decryption keys may

be futile. In particular, it enables the transmission of covert messages through innocent-looking ciphertexts, which remains undetectable even against a powerful adversary with access to the decryption keys.

Informally speaking, an AE scheme consists of a standard PKE scheme $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ together with an *anamorphic triplet* $\Sigma = (\mathsf{aKGen}, \mathsf{aEnc}, \mathsf{aDec})$. The system can operate either in the normal mode, using Π, or in the anamorphic mode, using Σ. In the anamorphic mode, aKGen outputs an anamorphic public-secret key pair $(\mathsf{apk}, \mathsf{ask})$, together with a pair $(\mathsf{sdk}, \mathsf{rdk})$ that consists of a *sender double key* and a *receiver double key*. The sender double key sdk enables the anamorphic encryption algorithm aEnc to generate an anamorphic ciphertext act that simultaneously carries two messages: an innocuous message m and a covert message am. The receiver double key rdk enables the anamorphic decryption algorithm aDec to recover am from act. The security is defined against a powerful adversary called the *dictator* that models a strong government with access to the user secret key. Concretely, the dictator should be unable to distinguish the normal mode from the anamorphic mode, even given a public-secret key pair (generated by KGen or aKGen, respectively) and oracle access to encryptions (implemented by Enc or aEnc, respectively). Since its introduction, three main notions have emerged in the literature, which we briefly review as follows.

1. **SAE.** The original notion of AE is now referred to as Symmetric AE (SAE), where $\mathsf{sdk} = \mathsf{rdk} = \mathsf{dk}$, with the shared key dk simply called *double key*.
2. **AAE.** The notion of Asymmetric AE (AAE) was introduced in [12], where $(\mathsf{sdk}, \mathsf{rdk})$ forms an asymmetric key pair.
3. **PKAE.** The Public-Key AE (PKAE) proposed by [27] can be viewed as a special case of AAE where the sender double key sdk is empty. Namely, anamorphic encryption can be directly performed using the anamorphic public key apk. This eliminates the need to privately distribute a sender double key, an interaction that may itself appear suspicious to the dictator.

In this paper, we investigate AAE in the setting of anamorphic extensions introduced by [4], where double keys can be generated directly from (even already deployed) public keys. Compared to SAE, AAE makes it possible to seek collusion-resistance where the dictator compels an anamorphic sender to disclose the (sender) double key [14]. In addition, no anamorphic extension for PKAE is known in the literature, and it seems very difficult to construct public-key anamorphic extension as observed in [27]. These considerations motivate our study of AAE, with prior work reviewed as follows.

Security Models of AAE. A commonly studied notion of AAE is *Fully Asymmetric Anamorphic Encryption* (F-AAE) [12, 13, 16], which is defined by two security requirements. The first is identical to the standard security requirement for the original AE notion discussed above: the dictator should be unable to distinguish the normal mode from the anamorphic mode, even given the public and secret keys. The second requirement concerns security against the anamorphic sender. Specifically, in the anamorphic mode, even an anamorphic sender given the anamorphic public key and the sender double key should learn no informa-

tion about both the normal message and the covert message from an anamorphic ciphertext.

A key limitation of F-AAE is that the dictator is not allowed to obtain the sender double key. Consequently, if the sender double key is compromised, a scenario termed *collusion* (between the dictator and an anamorphic sender) in [14], then F-AAE provides no further security guarantee.

To address this issue, Choi *et al.* [14] introduced a security notion termed *CIND-xCPA1*. Informally, it requires that, after the anamorphic mode is deployed, even if the dictator obtains both the anamorphic secret key and the sender double key, it still cannot determine whether a ciphertext was generated by the normal encryption algorithm or the anamorphic one. However, this notion still does not rule out the possibility that, once given the sender double key, the dictator can infer that the receiver has indeed deployed the anamorphic mode (rather than being uncertain if the sender double key was forged by the corrupted anamorphic sender), which may itself be incriminating.

Another drawback of the security definitions in [14] is their lack of parsimony. Even when restricted to CPA-security, capturing a sufficiently strong security guarantee for AAE requires three distinct security models: *STE-xCPA1* against the dictator, *SIND-nCPA* against the anamorphic sender, and *CIND-xCPA1* against collusion.

Constructions of AAE. As emphasized in [26], constructing entirely new PKE schemes tailored specifically for friendly covert communication may be of limited practical value, since a sufficiently powerful government could simply prohibit their use. A more standard approach is therefore to identify anamorphic modes within existing, especially widely deployed PKE schemes that were not originally designed to support covert channels.

Several works in the asymmetric setting follow this approach. Catalano *et al.* [12] show that the Cramer-Shoup Lite encryption scheme [15] and the Naor-Yung paradigm [25] achieve fully asymmetric anamorphism. Deo and Libert [16] study lattice-based PKE and establish fully asymmetric anamorphism for schemes including Regev [28] and dual Regev [21]. This line of work, however, proceeds on a scheme-by-scheme basis, with each design tailored to a particular PKE scheme or encryption structure.

An alternative line of research considers generic constructions of asymmetric anamorphic triplets. A representative result in this direction is due to Catalano *et al.* [13]. In particular, they construct an anamorphic triplet which satisfies F-AAE for any PKE scheme, as long as the ciphertext distribution for any message has sufficiently high entropy, which is a very mild restriction. However, the drawback of this approach lies in the reliance on a strong primitive, namely Indistinguishability Obfuscation (iO), the limited practical efficiency, and insufficient security guarantees (collusion is not considered).

Motivation. In summary, existing treatments of AAE have the following limitations. Firstly, the security models for collusion-resistance remain insufficiently strong. Secondly, the treatment of CPA-security lacks conciseness, requiring several separate notions. Thirdly, a generic collusion-resistant construction under

mild restrictions (e.g., high min-entropy ciphertexts) on the underlying PKE scheme is still lacking.

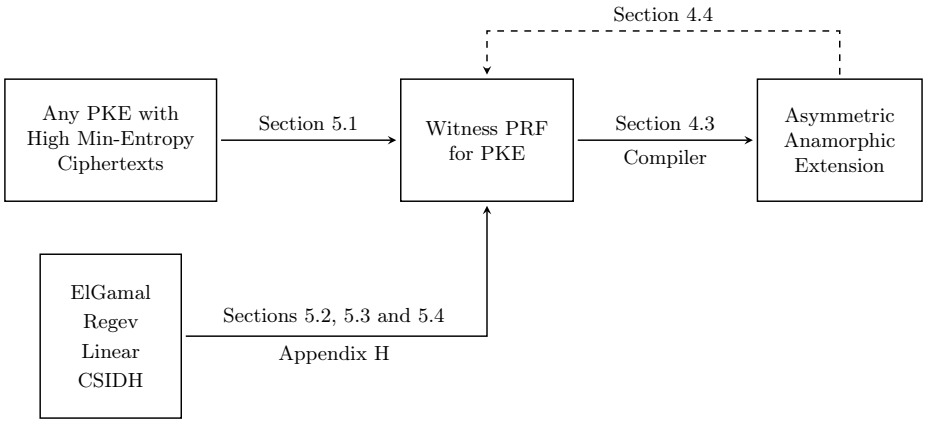


Fig. 1: Our Constructions of Asymmetric Anamorphic Extensions

1.1 Contributions

To address the above limitations, we make the following main contributions, with our constructions illustrated in Fig. 1 for ease of exposition.

1. **Definition.** We formalize the security of collusion-resistance for asymmetric anamorphic extensions by a security model termed *indistinguishability against collusion*. This model addresses the incompleteness of prior collusion models. Moreover, together with the CPA-security of the underlying PKE, this single model subsumes the existing CPA-security requirements for AAE, yielding a stronger and more concise security formulation.
2. **Framework.** We introduce a new primitive, called *witness PRF for PKE*, as an intermediate layer between the PKE scheme and the goal of asymmetric anamorphism. Accordingly, we give a compiler that transforms this intermediate layer into a collusion-resistant asymmetric anamorphic extension. In this way, establishing asymmetric anamorphism for a PKE scheme is reduced to constructing an appropriate witness PRF for it.
   We also demonstrate that the intermediate layer of witness PRF for PKE is essential in every asymmetric anamorphic extension in our setting. Concretely, given any PKE scheme, we prove that an asymmetric anamorphic extension for it gives rise to a witness PRF for it. Together with the compiler, this converse result shows that the witness PRF for PKE is a precise cryptographic abstraction for our task.
3. **Instantiations.** We provide several instantiations of our framework.

(a) Our first instantiation is a generic construction.[4] As with the iO-based constructions of [13], our generic construction applies to any PKE scheme whose ciphertext distribution for any message has high entropy, which is a very mild restriction. Specifically, we construct a witness PRF for any such PKE scheme, thereby realizing the intermediate layer generically. Then with our compiler, we obtain an asymmetric anamorphic extension for any PKE scheme with high min-entropy ciphertexts.
Compared to the iO-based constructions of [13], our generic construction is proven secure against collusion between the dictator and an anamorphic sender, relies on the assumption of witness PRF for any NP-relation which is possibly weaker than iO [35], and avoids complexity leveraging without requiring perfect correctness of the underlying PKE scheme.

(b) From a practical perspective, we give direct constructions of witness PRFs for the ElGamal, Regev, Linear, and CSIDH-based encryption schemes, without relying on the previous strong primitive of a witness PRF for any NP-relation. These instantiations for specific PKE schemes further illustrate the versatility of our framework.
To our knowledge, these yield the first practical, zero-latency asymmetric anamorphic extensions for ElGamal, Linear, and CSIDH-based encryptions. In particular, the anamorphic ElGamal using multi-message extension from [3] requires two ciphertexts to recover a covert message, thus destroying zero-latency [17], whereas our construction recovers it from a single ciphertext.

1.2 Additional Related Work

AE was introduced by [26] and is related to several earlier cryptographic notions, including key escrow [24, 6, 20], deniable encryption [9], kleptography [32–34], (public-key) steganography [29, 31], and subvertible backdoored encryption [22]. A detailed comparison among these notions is provided in the original work [26].

The models for AE have evolved through a sequence of works. In the original proposal of [26], both sender AE and receiver AE were introduced. Among these, receiver AE subsequently became the dominant line of research. Accordingly, we focus on it in this paper and write AE when no ambiguity arises. The notions of anamorphic extension and robustness were introduced by [4]. Anamorphic extension allows covert channels to be opened even after the deployment of a normal public key, while robustness enables the receiver to determine whether a given ciphertext indeed contains covert information. The original notion of AE is symmetric (SAE), in the sense that the two parties engaged in covert communication share the same double key. The notion of AAE was proposed by [12], where the double key is split into a sender double key for embedding covert messages and a receiver double key for recovering them. PKAE was introduced

[4] We emphasize that the later instantiations for specific PKE schemes are not special cases of this generic construction. All of them should instead be viewed as separate instantiations of our framework.

by [27], where no sender double key is required, so the anamorphic receiver need not interact with the anamorphic sender in advance.

At the construction level, designing PKE schemes specifically for convenient covert communication may be of limited interest, since a sufficiently powerful dictator could simply prohibit their use [26]. Consequently, a line of research has focused on the anamorphism of existing and widely deployed PKE schemes that were not originally designed to support covert channels. Examples include ElGamal [23, 4], primal and dual Regev [16], RSA-OAEP [23, 4], Cramer-Shoup [23, 4, 12], and the Naor-Yung paradigm [26, 12].

One generic approach [23, 14] requires the underlying PKE scheme to satisfy randomness recovery. In this approach, covert transmission is achieved by replacing the encryption randomness with a ciphertext of the covert message under another PKE scheme with pseudorandom-ciphertext security. However, this method imposes relatively strong requirements on the underlying PKE scheme. Another generic approach is the rejection sampling technique proposed by [26], which can equip any PKE scheme having high min-entropy ciphertexts (a mild requirement) with a symmetric anamorphic extension. A concrete security analysis was later given by [13]. Building on iO (essentially through the use of Sahai-Waters encryption [30]), together with rejection sampling, Catalano *et al.* [13] obtained a generic construction that equips any PKE scheme having high min-entropy ciphertexts with asymmetric anamorphic extensions.

On the negative side, Catalano *et al.* [13] showed that generic stateless black-box AE requires weaker security or restrictions on the underlying PKE. F-AAE remains unachievable by generic black-box constructions even with high min-entropy ciphertexts and semi-adaptive security. Moreover, several recent works [18, 10, 2] have investigated anamorphic-resistant encryption schemes.

2 Technical Overview

For simplicity, throughout this section we assume that the covert message am carried by each ciphertext is a single bit, that is, $\mathsf{am} \in \{0, 1\}$. We write aS, aR, and D for the anamorphic sender, anamorphic receiver, and dictator, respectively.

2.1 Prior Generic Construction for Symmetric Anamorphism

We first recall the generic rejection sampling technique [26], which yields a symmetric anamorphic extension for any PKE scheme $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ (with high min-entropy ciphertexts [13]).

In this approach, the double key dk shared by the anamorphic sender aS and receiver aR for covert communication is a key K of some secure PRF, denoted by F with output space $\{0, 1\}$. To transmit a covert message $\mathsf{am} \in \{0, 1\}$, aS repeatedly generates ciphertexts of an innocent-looking message m under the public key pk of aR, as $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$, until getting a ciphertext ct such that $\mathsf{am} = F(K, \mathsf{ct})$. It then sends the resulting ciphertext ct to aR. On the receiver side, the covert message am is recovered simply by computing $F(K, \mathsf{ct})$.

2.2 Our Generic Construction for Asymmetric Anamorphism

The above approach is naturally suited to symmetric anamorphism, since aS and aR have to share the same PRF key. To achieve asymmetric anamorphism for any PKE scheme $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ (with high min-entropy ciphertexts), we can instead use witness PRFs [35], in the above rejection sampling procedure.

Informally speaking, a witness PRF for an NP-language L is a function associated with a pair of public evaluation key ek and secret function key fk. On the one hand, the function value on any statement x can be efficiently computed using fk, denoted by $\mathsf{F}(\mathsf{fk}, \mathsf{x}) \to 0/1$. On the other hand, given ek, anyone that knows a witness w such that $\mathsf{x} \in L$ is also able to evaluate the function value on x, denoted by $\mathsf{Eval}(\mathsf{ek}, \mathsf{w}, \mathsf{x}) \to 0/1$.

First Attempt. A natural approach is to define the NP-language L as the set of ciphertexts under the public key pk of aR:

$$L = \{\mathsf{ct} : \exists\ (\mathsf{r}, \mathsf{m}) \text{ s.t. } \mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{r})\}\,.$$

Suppose that we adopt a witness PRF and obtain $(\mathsf{ek}, \mathsf{fk}, \mathsf{Eval}, \mathsf{F})$ for the language L. The anamorphic sender aS and receiver aR can then extract the same bit from a ciphertext ct as follows:

$$\mathsf{aS} : \mathsf{Eval}(\mathsf{ek}, \mathsf{w} = (\mathsf{r}, \mathsf{m}), \mathsf{x} = \mathsf{ct}), \quad \mathsf{aR} : \mathsf{F}(\mathsf{fk}, \mathsf{x} = \mathsf{ct}),$$

where ek and fk serve as the sender double key sdk and receiver double key rdk respectively. Besides, on the sender side, aS also uses rejection sampling to select a proper ciphertext such that the evaluation by aR yields am.

Problems. This construction is incomplete for the following two reasons.

First, we may consider an extreme case in which the underlying PKE scheme satisfies randomness recovery. Intuitively, this property means that the encryption randomness r used to generate a ciphertext ct can be recovered from ct using the secret key sk. Now assume a powerful dictator D that knows sk and further corrupts an anamorphic sender to obtain sdk. It can recover the encryption randomness r and the message m using sk, and subsequently invoke $\mathsf{Eval}(\mathsf{ek}, (\mathsf{r}, \mathsf{m}), \mathsf{ct})$, which is undesirable, since we require the derived bit to remain pseudorandom to D.

Next, a witness PRF only guarantees pseudorandomness of function values on statements outside the underlying language, which is insufficient in our setting. Indeed, we require that the function values on random ciphertexts generated by aS be pseudorandom, whereas any correctly generated ciphertext necessarily lies in the language L.

Second Attempt. To address the above issue, we exploit a secure PRG G. In particular, aS manipulates the random tape of Enc, and generates ciphertexts as

$$\mathsf{ct} = \mathsf{wEnc}(\mathsf{pk}, \mathsf{m}; \mathsf{s}) := \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s})),$$

where s is a uniformly random seed. Again, we leverage a witness PRF scheme to establish the keys and algorithms $(\mathsf{ek}, \mathsf{fk}, \mathsf{Eval}, \mathsf{F})$, with the difference that the targeted language is now switched to

$$L = \{\mathsf{ct} : \exists\ (\mathsf{s}, \mathsf{m}) \text{ s.t. } \mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))\}\,.$$

Moreover, using ek and fk as sdk and rdk, respectively, aS and aR can extract the same bit from a ciphertext ct as follows:

$$\mathsf{aS} : \mathsf{Eval}(\mathsf{ek}, \mathsf{w} = (\mathsf{s}, \mathsf{m}), \mathsf{x} = \mathsf{ct}), \quad \mathsf{aR} : \mathsf{F}(\mathsf{fk}, \mathsf{x} = \mathsf{ct}).$$

We now return to the problems raised earlier. First, assume that the underlying PKE scheme Π satisfies randomness recovery. By the security of PRG, even if D can recover the randomness r such that $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{r})$, it still cannot recover the seed s for $\mathsf{r} = \mathsf{G}(\mathsf{s})$, and hence may no longer be able to invoke Eval as before.

Next, the purpose of PRG is to ensure that a random ciphertext ct generated by Enc lies outside the language L with overwhelming probability. Intuitively, if the seed length of G is much shorter than its output length (i.e., the length of random coins used by Enc), then for a random ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$, with overwhelming probability there exists no seed s such that $\mathsf{ct} = \mathsf{wEnc}(\mathsf{pk}, \mathsf{m}; \mathsf{s}) = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$. If our intuition is correct, then the function value of ct is pseudorandom. Furthermore, by the pseudorandomness of G, the function values on random ciphertexts generated by aS via wEnc are also pseudorandom.

Problem. Unfortunately, if the underlying PKE scheme Π is not perfectly correct, we cannot rule out the possibility that there exists a bad message m^* such that, for $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}^*)$, with non-negligible probability, there exist a seed s' and a different message $\mathsf{m}' \neq \mathsf{m}^*$ for which $\mathsf{ct}^* = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}'; \mathsf{G}(\mathsf{s}'))$, meaning that $\mathsf{ct}^* \in L$.

Final Construction. To address the above problem, we view the pair $(\mathsf{m}, \mathsf{ct})$, instead of ct alone, as a statement. Accordingly, the witness is a seed s such that $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$. Similarly, we exploit a witness PRF scheme to obtain the keys and algorithms $(\mathsf{ek}, \mathsf{fk}, \mathsf{Eval}, \mathsf{F})$, for the current language

$$L = \{(\mathsf{m}, \mathsf{ct}) : \exists\ \mathsf{s} \text{ s.t. } \mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))\}.$$

Using ek and fk as sdk and rdk, respectively, aS and aR can again derive the same bit from a ciphertext ct as follows:

$$\mathsf{aS} : \mathsf{Eval}(\mathsf{ek}, \mathsf{w} = \mathsf{s}, \mathsf{x} = (\mathsf{m}, \mathsf{ct})), \quad \mathsf{aR} : \mathsf{F}(\mathsf{fk}, \mathsf{x} = (\mathsf{m}, \mathsf{ct})).$$

We emphasize that given any message m, for a suitable seed length (refer to Section 5.1 for details) and $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$, with overwhelming probability there exists no s such that $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$. This means that, in our current setting, the statement $(\mathsf{m}, \mathsf{ct})$ lies outside L with overwhelming probability, and our goal for pseudorandomness follows.

Also notice that, on the receiver side of our generic construction, since the receiver does not obtain m directly, as part of the statement, it should first decrypt ct using the secret key sk to recover m before invoking the algorithm F.

For completeness, we give a schematic description of our generic construction in Fig. 2, where we highlight in blue the changes relative to the rejection sampling for symmetric anamorphism.

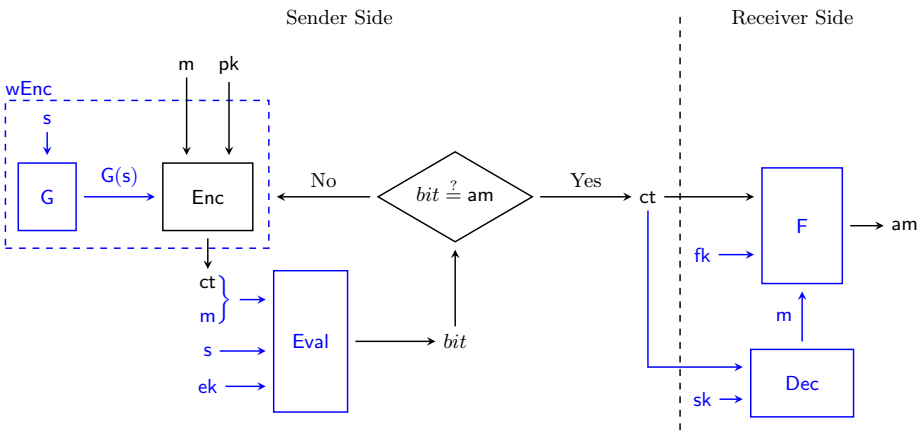


Fig. 2: Generic Construction for Asymmetric Anamorphic Extension. Here, $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is a PKE scheme with high min-entropy ciphertexts, where $(\mathsf{pk}, \mathsf{sk})$ denotes the public-secret key pair of the receiver, and ct denotes the ciphertext of message m. The anamorphic message sent to the receiver is denoted as am. wEnc refers to the encryption process that encrypts message m into ct, where the randomness used in Enc is generated by a PRG G with a uniform seed s. $(\mathsf{F}, \mathsf{Eval})$ are algorithms of the witness PRF scheme, where $(\mathsf{ek}, \mathsf{fk})$ are public evaluation and secret function keys, respectively.

2.3 Observation from ElGamal Encryption

We observe that, for ElGamal encryption [19], we can directly construct a form of witness PRF (which can then be combined with rejection sampling to obtain an asymmetric anamorphic extension), without relying on primitives as strong as witness PRFs for arbitrary NP-relations.

Recall that an ElGamal ciphertext has the form

$$\mathsf{ct} = (c_0, c_1) = (g^s, h^s \cdot m),$$

where $h = g^x$ is the public key and x is the secret key.

Our sole goal is to enable two parties to extract from ct the same pseudorandom group element, which can be further converted into a pseudorandom bit with a randomness extractor.

In particular, we sample $\widehat{x} \xleftarrow{\$} \mathbb{Z}_q$, where q is the group order, then set the evaluation key and function key as $\mathsf{ek} = \widehat{h} = g^{\widehat{x}}$ and $\mathsf{fk} = \widehat{x}$, respectively.

- Eval : On the sender side, given $\mathsf{ek} = \widehat{h}$ and the encryption randomness s, the target element is computed as $R = \widehat{h}^s$ (the element $g^{\widehat{x}s}$), whose pseudorandomness is due to the Decisional Diffie-Hellman (DDH) assumption.
- F : On the receiver side, given $\mathsf{fk} = \widehat{x}$, the target can be calculated from the ciphertext as $R = c_0^{\widehat{x}}$, resulting in the same element $g^{\widehat{x}s}$.

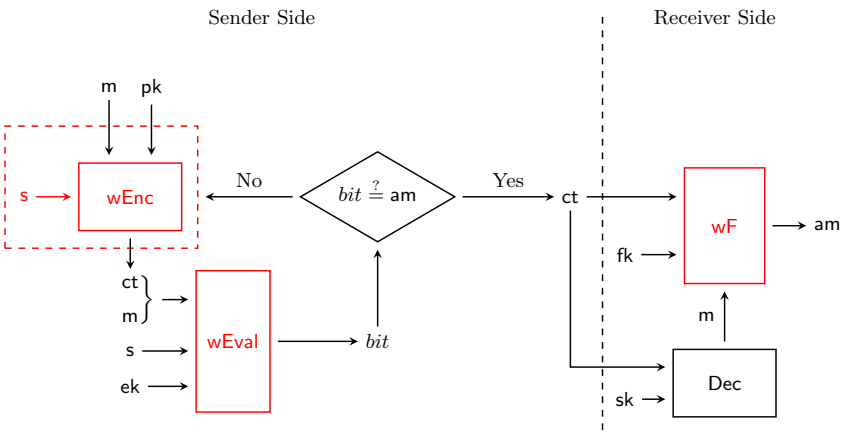


Fig. 3: The Compiler for Asymmetric Anamorphic Extensions. Here, $\mathsf{wPRF} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wF}, \mathsf{wEval})$ is a witness PRF for some PKE $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$.

2.4 Framework

From the above observation on ElGamal encryption and for practical purposes, constructing an asymmetric anamorphic extension for a specific PKE scheme does not necessarily require a strong primitive as witness PRF for arbitrary NP-relation. Instead, it suffices to have a form of witness PRF, for a particular NP-relation related to the PKE.

Intermediate Layer. We abstract the previously seen requirements into a new primitive, called *witness PRF for PKE*, which isolates an intermediate layer from the task of identifying asymmetric anamorphism. Intuitively, a witness PRF for a PKE scheme $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is defined by a pair of public evaluation key ek and secret function key fk, intended to serve as the sender double key sdk and the receiver double key rdk, respectively. As discussed above, it includes a new encryption algorithm $\mathsf{wEnc}(\mathsf{pk}, \mathsf{m})$ with internal encryption randomness denoted by s. The encryption procedure of wEnc may differ from that of Enc, provided that the resulting ciphertexts are indistinguishable from those generated by Enc. Unlike a general witness PRF for an arbitrary NP-relation, this primitive is tailored to the PKE scheme Π, and more precisely to the relation induced by wEnc under the public key pk of anamorphic receiver. Concretely, the algorithm wEval (to be used by the anamorphic sender) takes as input the evaluation key ek, a statement $\mathsf{x} = (\mathsf{m}, \mathsf{ct})$ consisting of a message m and a ciphertext ct, along with the related witness s such that $\mathsf{ct} = \mathsf{wEnc}(\mathsf{pk}, \mathsf{m}; \mathsf{s})$, then outputs the function value on $(\mathsf{m}, \mathsf{ct})$. For security, we require that the function values on statements, where for each one $\mathsf{x} = (\mathsf{m}, \mathsf{ct})$, ct is a random ciphertext generated by wEnc of the specific message m, be pseudorandom, which constitutes a form of weak pseudorandomness. Finally, the algorithm wF (to be used by the anamorphic receiver) can leverage the secret function key fk to efficiently compute the function value on any statement $\mathsf{x} = (\mathsf{m}, \mathsf{ct})$.

Compiler. The intermediate layer can be compiled into an asymmetric anamorphic extension, with a process similar to our former generic construction. The process is shown in Fig. 3, with the main algorithms of witness PRF for PKE highlighted in red. Under our framework, the task of designing an asymmetric anamorphic mode for a PKE scheme reduces to constructing an associated witness PRF. Accordingly, in the instantiations that follow, it is sufficient to present suitable constructions of witness PRFs for the given PKE schemes.

Remark. Our earlier generic construction can be reinterpreted as using a general witness PRF to construct a witness PRF for any PKE scheme (with high min-entropy ciphertexts). Therefore, it can also be regarded as an instantiation of our framework. Also note that, in the instantiations for specific PKE schemes that follow, we will not rely on primitives as strong as a general witness PRF. Moreover, the decryption step on the receiver side can be omitted, since the message m no longer needs to participate in the computations of function values.

2.5 Regev Encryption

We can construct a witness PRF for Regev encryption scheme [28]. For $\mathbb{Z}/q\mathbb{Z}$ where $q > 2$ is a prime, we use the representatives $\mathbb{Z}_q = \left[-\frac{q-1}{2}, \frac{q-1}{2}\right]$. Recall that a ciphertext encrypting a message $\mu \in \{0,1\}$ has the form

$$\mathsf{ct} = \begin{bmatrix} \mathbf{A}\mathbf{s} \\ \mathbf{b}^t\mathbf{s} + \mu \cdot \lfloor \frac{q}{2} \rceil \end{bmatrix} \in \mathbb{Z}_q^n \times \mathbb{Z}_q,$$

where $\left(\mathbf{A} \in \mathbb{Z}_q^{n\times m}, \mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e} \in \mathbb{Z}_q^m\right)$ is the public key, $\mathbf{x} \in \mathbb{Z}_q^n$ is the secret key, and $\mathbf{s} \in \{0,1\}^m$ is the encryption randomness.

Similar to the construction for ElGamal encryption, we set the secret function key and the public evaluation key to $\mathsf{fk} = \widehat{\mathbf{x}} \in \mathbb{Z}_q^n$ and $\mathsf{ek} = \widehat{\mathbf{b}} = \mathbf{A}^t\widehat{\mathbf{x}} + \widehat{\mathbf{e}} \in \mathbb{Z}_q^m$, respectively, following the Learning With Errors (LWE) problem. Given $\mathsf{ek} = \widehat{\mathbf{b}}$ and the encryption randomness $\mathbf{s}$ of a ciphertext

$$\mathsf{ct} = \begin{bmatrix} \mathbf{c} \\ c \end{bmatrix} = \begin{bmatrix} \mathbf{A}\mathbf{s} \\ \mathbf{b}^t\mathbf{s} + \mu \cdot \lfloor \frac{q}{2} \rceil \end{bmatrix},$$

one can compute the value

$$r = \widehat{\mathbf{b}}^t\mathbf{s} = \widehat{\mathbf{x}}^t\mathbf{A}\mathbf{s} + \widehat{\mathbf{e}}^t\mathbf{s} \in \mathbb{Z}_q,$$

which is pseudorandom (by appropriate parameters). On the other hand, given $\mathsf{fk} = \widehat{\mathbf{x}}$, one can compute

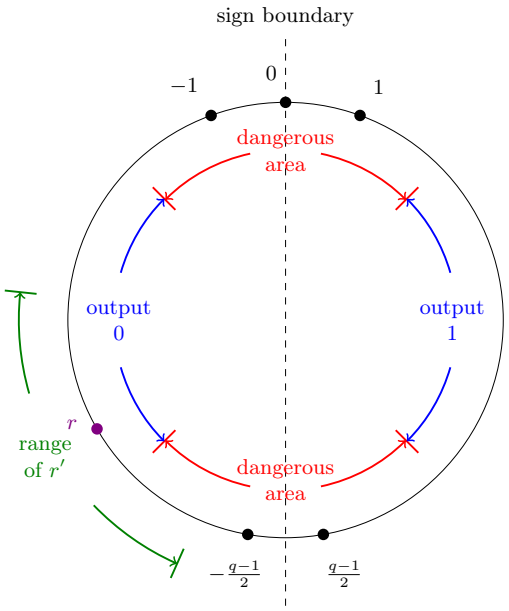
$$r' = \widehat{\mathbf{x}}^t\mathbf{c} = \widehat{\mathbf{x}}^t\mathbf{A}\mathbf{s},$$

which is close to r. Their difference is

$$\Delta = r - r' = \widehat{\mathbf{e}}^t\mathbf{s}.$$

To make wEval and wF output a pseudorandom bit, a natural idea is to use the sign of r and r': output 0 when the value is negative, and output 1 otherwise, for now just ignoring the boundary case where the value is equal to 0.

Problem. In the Regev encryption scheme, the modulus q is typically polynomially large in the security parameter. Consequently, the probability that r takes a value close to the sign boundary (0 and $\pm\frac{q-1}{2}$) is non-negligible, causing that the signs of r and r' may differ with non-negligible probability. For example, when $r = 1$, an extremely small error term $\Delta = 2$ already suffices to flip the sign: $r' = r - \Delta = -1$.



Notes: 1. In our construction, an upper bound on the value $|\Delta|$, where Δ treated as an element in $\mathbb{Z}$ denotes the difference between r (computed by wEval) and r' (computed by wF), is $q/10$.
2. Given this upper bound on $|\Delta|$, it suffices to let each dangerous region, namely the neighborhoods of 0 or $\pm\frac{q-1}{2}$ in the above circle, occupy roughly 1/4 of $\mathbb{Z}_q$.
3. As illustrated in the above circle, whenever r lies outside the dangerous regions, wEval can safely output 0 or 1 according to the sign of r. In this case, the possible range of r' does not cross the sign boundary, and hence wF will output the same bit.

Fig. 4: Outputs of wEval According to Range of r for Regev Encryption

Solution. To address this issue, we let the wEval algorithm output a special symbol $\perp \notin \{0, 1\}$ whenever r falls into such a dangerous region. For clarity, we give a detailed description of this procedure in Fig. 4. Returning to the framework in Fig. 3, this means that on the sender side if wEval outputs $\perp$ on a ciphertext ct in some trial, then the sender simply rejects ct (also generates a

fresh ciphertext for the next attempt), even if the sign of r matches the covert bit it intends to transmit. This ensures that at the receiver side an opposite bit will not be obtained when applying wF.

2.6 Linear Encryption

Finally, we give a witness PRF for Linear encryption [7] under the Decisional LINear (DLIN) assumption rather than DDH. This distinction is meaningful since DLIN may remain hard even when DDH is efficiently decidable. Recall the DLIN assumption: for independent uniform group elements $u, v, h \in \mathbb{G}$ and exponents $r, s, t \in \mathbb{Z}_q$,

$$(u, v, h, u^r, v^s, h^{r+s}) \approx (u, v, h, u^r, v^s, h^t),$$

which directly gives the ciphertext structure of Linear encryption:

$$\mathsf{ct} = (c_0, c_1, c_2) = \left(u^r, v^s, h^{r+s} \cdot m\right).$$

Here, the public key is $\mathsf{pk} = (u, v, h)$, and the secret key $\mathsf{sk} = (x, y)$ specifies the discrete logarithm relations $u^x = v^y = h$. These enable the receiver to recover the mask $h^{r+s} = c_0^x c_1^y$ and decrypt by division.

First Attempt. As in the ElGamal construction, it is sufficient to see how the sender and receiver derive the same pseudorandom group element. Our goal is to embed a DLIN tuple

$$\left(g_1, g_2, g_3, g_1^\sigma, g_2^\tau, g_3^{\sigma+\tau}\right),$$

where $R = g_3^{\sigma+\tau}$ is our target pseudorandom group element, the bases g_1, g_2, g_3 are public, and the receiver should be able to derive the two shares g_1^σ, g_2^τ from the ciphertext. Besides, the receiver should know the secret discrete logarithm relations among these bases, as a trapdoor for computing R. A natural attempt is to retain ordinary encryption with encryption randomness (σ, τ), resulting in $\mathsf{ct} = (c_0, c_1, c_2) = (u^\sigma, v^\tau, h^{\sigma+\tau} \cdot m)$. Specifically, we choose $\widehat{a}, \widehat{b} \xleftarrow{\$} \mathbb{Z}_q$ as the function key fk and set

$$g_1 = h, \quad g_2 = h^{\widehat{a}}, \quad g_3 = h^{\widehat{a}\widehat{b}}.$$

On the sender side, the target element $R = g_3^{\sigma+\tau}$ can be computed directly using the encryption randomness (σ, τ). With $\mathsf{sk} = (x, y)$ and $\mathsf{fk} = (\widehat{a}, \widehat{b})$, the receiver first derives the shares $g_1^\sigma = c_0^x$ and $g_2^\tau = c_1^{y \cdot \widehat{a}}$ from ct, and then computes the target as $R = (g_1^\sigma)^{\widehat{a} \cdot \widehat{b}} \cdot (g_2^\tau)^{\widehat{b}}$.

Problem. The dictator also knows sk and hence can recover $h^{\sigma+\tau} = c_0^x c_1^y$. As a result, the dictator essentially faces a DDH tuple

$$\left(h, g_3, h^{\sigma+\tau}, R\right) = \left(h, h^{\widehat{a}\widehat{b}}, h^{\sigma+\tau}, h^{\widehat{a}\widehat{b}(\sigma+\tau)}\right).$$

Thus, the pseudorandomness of R would rely on DDH, which DLIN alone does not guarantee.

Final Construction. To prevent the dictator from recovering the DDH share $h^{\sigma+\tau}$, we replace τ by $\widehat{a}\tau$ in the ciphertext:

$$\mathsf{ct} = (c_0, c_1, c_2) = \left(u^{\sigma}, v^{\widehat{a}\tau}, h^{\sigma+\widehat{a}\tau} \cdot m\right) = \mathsf{Enc}(\mathsf{pk}, m; (\sigma, \widehat{a}\tau)).$$

Essentially, on the sender side the ciphertext ct is computed with encryption randomness (σ, τ) through a modified encryption algorithm as $\mathsf{ct} = (u^{\sigma}, \widehat{v}^{\tau}, h^{\sigma} \cdot \widehat{h}^{\tau} \cdot m)$, where $\widehat{v} = v^{\widehat{a}}$ and $\widehat{h} = h^{\widehat{a}} = g_2$, elements included in the evaluation key ek. As before, the target element $R = g_3^{\sigma+\tau}$ can be computed directly using the randomness (σ, τ).

On the receiver side, with $\mathsf{sk} = (x, y)$ and $\mathsf{fk} = (\widehat{a}, \widehat{b})$, the receiver can first derive the shares $g_1^{\sigma} = c_0^x$ and $g_2^{\tau} = c_1^y$ from ct, and then compute the target element as $R = (g_1^{\sigma})^{\widehat{a}\cdot\widehat{b}} \cdot (g_2^{\tau})^{\widehat{b}}$, which is exactly $g_3^{\sigma+\tau}$.

2.7 Discussion

Our witness PRF for ElGamal easily extends to one for CSIDH-based ElGamal, by replacing exponentiation with the corresponding commutative group action. The resulting instantiation is presented in Appendix H.

For PKE schemes with randomness recovery, a more direct practical approach is to encrypt the covert message under an auxiliary PKE scheme with pseudorandom ciphertexts, then use the ciphertext as the encryption randomness [23, 14], rather than apply our framework. Despite this practical limitation, our framework remains compatible with such PKE schemes, as demonstrated by our generic construction (Section 5.1) and converse result (Section 4.4).

3 Preliminaries

3.1 Notations

In this paper, we use PPT as shorthand for Probabilistic Polynomial Time. For a finite set X, we write $x \xleftarrow{\$} X$ to denote uniformly sampling at random an element x from X. More generally, $d \hookleftarrow D$ denotes that d is sampled according to some distribution D. We write $y \leftarrow A(x)$ to represent an invocation of algorithm A on input x with output y. If A is randomized, we can equivalently represent it as a deterministic algorithm as $A(x; r)$, where r denotes its internal coin tosses. For integers a, b such that $a \leq b$, we use $[a, b]$ to denote the set $\{a, a+1, \cdots, b-1, b\}$. The symbol $\approx$ refers to computational or statistical indistinguishability. We use $\mathsf{negl}(\cdot)$ and $\mathsf{poly}(\cdot)$ to denote some negligible function and some polynomial, respectively.

3.2 Public Key Encryption

Definition 1 (Public Key Encryption). *A Public Key Encryption (PKE) is defined by a triple of efficient algorithms* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *as follows.*

- $\mathsf{KGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$. *The key generation algorithm takes as input a security parameter* λ*, then outputs a pair* $(\mathsf{pk}, \mathsf{sk})$ *of public and secret key.*
- $\mathsf{Enc}(\mathsf{pk}, \mathsf{m}) \to \mathsf{ct}$. *The encryption algorithm takes as input a public key* pk *and a message* $\mathsf{m} \in \mathcal{M}_{\mathsf{pk}}$ *where* $\mathcal{M}_{\mathsf{pk}}$ *denotes the message space, then computes a ciphertext* ct *of* m.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to \mathsf{m}$. *The decryption algorithm takes as input a secret key* sk *and a ciphertext* ct*, then recovers the corresponding plaintext* m.

Definition 2 (Correctness of PKE). *A PKE scheme* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *is correct, if for* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$*, for any* $\mathsf{m} \in \mathcal{M}_{\mathsf{pk}}$*, for* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$*, it holds that* $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = \mathsf{m}$*, except for a negligible probability.*

Definition 3 (IND-CPA Security). *A PKE scheme* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *is INDistinguishable against Chosen Plaintext Attacks (IND-CPA), if for any PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, its advantage is negligible in the security parameter:*

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

where the experiment ExpCPA *is defined in Fig. 5.*

$\mathsf{ExpCPA}_{\mathcal{A},b}(1^\lambda)$:

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{m}_0, \mathsf{m}_1, \mathsf{state}) \leftarrow \mathcal{A}_1(\mathsf{pk})$
3 $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}_b)$
4 **return** $\mathcal{A}_2(\mathsf{state}, \mathsf{ct})$

Fig. 5: The Experiment ExpCPA

For our generic construction, we require the underlying PKE scheme to have high min-entropy ciphertexts, defined as follows.

Definition 4 (High Min-Entropy Ciphertexts). *We say a PKE scheme* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *has high min-entropy ciphertexts, if for any* $(\mathsf{pk}, \mathsf{sk})$ *in the range of* KGen*, and any message* $\mathsf{m} \in \mathcal{M}_{\mathsf{pk}}$*, there is*

$$\mathsf{H}_\infty(\mathsf{Enc}(\mathsf{pk}, \mathsf{m})) = \Omega(\lambda).$$

For simplicity, we use $\mathsf{H}_\infty(\mathsf{Enc})$ *to denote the minimum, that is,*

$$\mathsf{H}_\infty(\mathsf{Enc}) = \min_{\mathsf{pk},\mathsf{m}} \left\{\mathsf{H}_\infty(\mathsf{Enc}(\mathsf{pk}, \mathsf{m}))\right\}.$$

3.3 Pseudorandom Generator

Definition 5 (Pseudorandom Generator). *Let $t(\cdot)$ be a polynomial, then a deterministic polynomial-time algorithm $\mathsf{G} : \{0,1\}^n \to \{0,1\}^{t(n)}$ is called a PRG if it satisfies the following two properties:*

1. *Expansion: $t(n) > n$ for every n.*
2. *Pseudorandomness: For any PPT adversary $\mathcal{A}$, we have*

$$\left| \Pr_{s \xleftarrow{\$} \{0,1\}^n} [\mathcal{A}(\mathsf{G}(s)) = 1] - \Pr_{r \xleftarrow{\$} \{0,1\}^{t(n)}} [\mathcal{A}(r) = 1] \right| \leq \mathsf{negl}(n).$$

3.4 Witness PRF

Definition 6 (Witness PRF [35]). *A witness PRF for NP-relations is a triple of algorithms $\mathsf{WPRF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Eval})$ defined as follows.*

- $\mathsf{Gen}(1^\lambda, R) \to (\mathsf{ek}, \mathsf{fk})$. *The key generation algorithm takes as input a security parameter λ and a relation circuit $R : \mathcal{W} \times \mathcal{X} \to \{0,1\}$, where $\mathcal{W}$ and $\mathcal{X}$ denote witness space and statement space, respectively, then outputs a public evaluation key ek and a secret function key fk.*
- $\mathsf{F}(\mathsf{fk}, \mathsf{x}) \to \mathsf{y}$. *The algorithm takes as input a function key fk and a statement $\mathsf{x} \in \mathcal{X}$, then computes the function value $\mathsf{y} \in \mathcal{Y}$ for some set $\mathcal{Y}$ on x.*
- $\mathsf{Eval}(\mathsf{ek}, \mathsf{w}, \mathsf{x}) \to \mathsf{y}$. *The algorithm takes as input an evaluation key ek, a witness $\mathsf{w} \in \mathcal{W}$, and a related statement $\mathsf{x} \in \mathcal{X}$, then outputs the function value $\mathsf{y} \in \mathcal{Y}$ on x.*

Definition 7 (Correctness of Witness PRF). *For any NP-relation R, $\mathsf{w} \in \mathcal{W}$ and $\mathsf{x} \in \mathcal{X}$, then for $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R)$, there is $\mathsf{Eval}(\mathsf{ek}, \mathsf{w}, \mathsf{x}) = \mathsf{F}(\mathsf{fk}, \mathsf{x})$ if $R(\mathsf{w}, \mathsf{x}) = 1$.*

Definition 8 (Pseudorandomness). *The pseudorandomness property of witness PRF is defined with respect to statements x out of the language $L = \{\mathsf{x} : \exists\ \mathsf{w}$ s.t. $R(\mathsf{w}, \mathsf{x}) = 1\}$,namely $\mathsf{x} \notin L$. A witness PRF $\mathsf{WPRF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Eval})$ is pseudorandom if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, its advantage is negligible in the security parameter. In particular,*

$$\mathsf{Adv}^{\mathsf{PR}}_{\mathcal{A},\mathsf{WPRF}}(\lambda) = \left| \Pr\left[\mathsf{ExpPR}^R_{\mathcal{A},0}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ExpPR}^R_{\mathcal{A},1}(1^\lambda) = 1\right] \right| \leq \mathsf{negl}(\lambda),$$

where the experiment ExpPR is defined in Fig. 6.

3.5 Assumptions

Definition 9 (Decisional Diffie-Hellman Assumption). *Let $\mathbb{G} = (G, q, g)$ be a cyclic group G of prime order q with generator g. The Decisional Diffie-Hellman (DDH) assumption states that (g, g^a, g^b, g^{ab}) and (g, g^a, g^b, g^c) are computationally indistinguishable, where a, b, c are sampled independently and uniformly from $\mathbb{Z}_q$.*

$\mathsf{ExpPR}^{R}_{\mathcal{A},b}(1^\lambda)$:

1 $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R)$
2 $\mathcal{T} \leftarrow \varnothing$
3 $(\mathsf{x}^*, \mathsf{state}) \leftarrow \mathcal{A}_1^{\mathcal{O}(\cdot)}(\mathsf{ek})$
4 **where** $\mathcal{O}(\cdot)$ on input x:
5 $\mathcal{T} \leftarrow \mathcal{T} \cup \{\mathsf{x}\}$
6 **return** $\mathsf{F}(\mathsf{fk}, \mathsf{x})$
7 $\mathsf{y}_0 \leftarrow \mathsf{F}(\mathsf{fk}, \mathsf{x}^*)$, $\mathsf{y}_1 \xleftarrow{\$} \mathcal{Y}$
8 $b' \leftarrow \mathcal{A}_2^{\mathcal{O}(\cdot)}(\mathsf{state}, \mathsf{y}_b)$
9 **where** $\mathcal{O}(\cdot)$ on input x:
10 $\mathcal{T} \leftarrow \mathcal{T} \cup \{\mathsf{x}\}$
11 **return** $\mathsf{F}(\mathsf{fk}, \mathsf{x})$
12 **return** $b' \wedge (\mathsf{x}^* \notin \mathcal{T}) \wedge (\mathsf{x}^* \notin L)$

Fig. 6: The Experiment ExpPR

Definition 10 (Decisional LWE Assumption). *The Learning With Errors (LWE) problem is parameterized by integers n, m, and modulus q, together with an error distribution χ over $\mathbb{Z}$. The decisional LWE assumption states that, given a random matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and a vector $\mathbf{b} \in \mathbb{Z}_q^m$, it is hard to distinguish between the following two cases:*

1. $\mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e}$, *where* $\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$ *and* $\mathbf{e} \hookleftarrow \chi^m$,
2. $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_q^m$.

For suitable noise distributions $\chi = \overline{\Psi}_\alpha$, the LWE problem is known to be as hard as standard worst-case lattice problems under a quantum reduction.

Definition 11. *For $\alpha \in (0,1)$ and a prime q, let $\overline{\Psi}_\alpha$ denote the distribution over $\mathbb{Z}_q$ obtained by sampling a Gaussian random variable X with mean 0 and standard deviation $\alpha/\sqrt{2\pi}$, and outputting $\lfloor qX \rceil \bmod q$.*

Theorem 1 (Regev's Reduction [28]). *Let $\alpha \in (0,1)$ and let q be a prime satisfying $q > 2\sqrt{n}/\alpha$. If there exists an efficient, possibly quantum, algorithm for deciding the $(\mathbb{Z}_q, n, \overline{\Psi}_\alpha)$-LWE problem, then there exists an efficient quantum algorithm for approximating* SIVP *and* GapSVP *in the worst case within a factor of $\tilde{O}(n/\alpha)$ in the ℓ_2 norm.*

Therefore, assuming that approximating SIVP or GapSVP on n-dimensional lattices within some factor polynomial in n is hard, it follows that deciding LWE is hard whenever n/α is polynomial in n.

Definition 12 (Decisional Linear Assumption). *Let $\mathbb{G} = (G, q, g)$ be a cyclic group G of prime order q. The Decisional LINear (DLIN) assumption*

states that $(u, v, h, u^r, v^s, h^{r+s})$ *and* (u, v, h, u^r, v^s, h^t) *are computationally indistinguishable, where* u, v, h *are sampled independently and uniformly from* G*, and* r, s, t *are sampled independently and uniformly from* $\mathbb{Z}_q$*.*

3.6 Other Useful Lemmas

Lemma 1 (Rejection Sampling with a Predicate [13, generalized from Lemma 4 in the full version]). *Let* D *be a distribution over some set* X*, let* $v \geq 1$ *be an integer, and let* $0 \leq \theta \leq 1$*. Sample* $c_1, \ldots, c_{v+1}$ *independently from* D*. Independently of these samples, draw* $b_1, \ldots, b_v$ *independently from* $\text{Bernoulli}(\theta)$*. Define* c *to be* c_i*, where* i *is the smallest index such that* $b_i = 1$*, and let* $c = c_{v+1}$ *if no such index exists. Then the distribution of* c *is identical to* D*.*

Lemma 2 ([1, Lemma 12 in the full version]). *Let* $\mathbf{e}$ *be any vector in* $\mathbb{Z}^m$ *and let* $\mathbf{y} \hookleftarrow \overline{\Psi}_\alpha^m$*. Then the quantity* $|\mathbf{e}^t\mathbf{y}|$*, where* $\mathbf{e}^t\mathbf{y}$ *is viewed as an element of* $\mathbb{Z}$*, satisfies*

$$|\mathbf{e}^t\mathbf{y}| \leq \|\mathbf{e}\|\, q\alpha\, \omega(\sqrt{\log m}) + \|\mathbf{e}\|\sqrt{m}/2$$

with all but negligible probability in m*.*

Lemma 3 (Leftover Hash Lemma [8, Theorem 8.11]). *Let* $\mathbf{R} \in \{0,1\}^{m\times k}$ *be a random matrix with independent columns, each having min-entropy at least* κ*. Sample* $\mathbf{A} \xleftarrow{\$} \{0,1\}^{n\times m}$ *and* $\mathbf{B} \xleftarrow{\$} \{0,1\}^{n\times k}$ *independently of one another and of* $\mathbf{R}$*. Then the statistical distance between* $(\mathbf{A}, \mathbf{AR})$ *and* $(\mathbf{A}, \mathbf{B})$ *is at most* $\frac{k}{2} \cdot 2^{(n-\kappa)/2}$*, where matrix multiplication is over* $\mathbb{F}_2$*. In particular, the two distributions are statistically close if* $k = \mathsf{poly}(\lambda)$ *and* $\kappa \geq n + \omega(\log\lambda)$*.*

Lemma 4 (Generalized Leftover Hash Lemma [1, Lemma 4]). *Suppose that* $m > (n+1)\log_2 q + \omega(\log n)$*. Let* $k = k(n)$ *be polynomial in* n*. Let* $\mathbf{R} \xleftarrow{\$} \{0,1\}^{m\times k}$*,* $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$*, and* $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{n\times k}$*. Then, for every* $\mathbf{w} \in \mathbb{Z}_q^m$*, the distribution* $(\mathbf{A}, \mathbf{AR}, \mathbf{R}^\top\mathbf{w})$ *is statistically close to* $(\mathbf{A}, \mathbf{B}, \mathbf{R}^\top\mathbf{w})$*.*

4 Framework for Constructing AAEs

We first present the syntax and security model for asymmetric anamorphic extensions in Section 4.1. In Section 4.2, we introduce *witness PRF for PKE*, a new primitive at the core of our framework. Section 4.3 gives a compiler that transforms a witness PRF for PKE into an asymmetric anamorphic extension. Finally, Section 4.4 discusses the necessity of this primitive.

4.1 Collusion-Resistant Asymmetric Anamorphic Extension

We consider asymmetric anamorphic extensions for our framework. In this section, we present the syntax and formalize the security of collusion-resistance. Discussion of the original definition of AAE and issues with existing security models is deferred to Appendix A.

Definition 13 (Asymmetric Anamorphic Extension). *For a CPA-secure PKE scheme* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$*, we define an asymmetric anamorphic extension* $\Sigma = (\mathsf{aKGen}, \mathsf{aEnc}, \mathsf{aDec})$ *for* Π *by a triplet of efficient algorithms as:*

- $\mathsf{aKGen}(\mathsf{pk}) \to (\mathsf{sdk}, \mathsf{rdk})$*. The double key generation algorithm takes as input a public key* pk*, then outputs a pair of sender double key and receiver double key* $(\mathsf{sdk}, \mathsf{rdk})$*.*
- $\mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am}) \to \mathsf{ct}$*. The anamorphic encryption algorithm takes as input a public key* pk*, a sender double key* sdk*, a normal message* m*, and an anamorphic (covert) message* am*. It outputs an anamorphic ciphertext* ct*.*
- $\mathsf{aDec}(\mathsf{sk}, \mathsf{rdk}, \mathsf{ct}) \to \mathsf{am}$*. The anamorphic decryption algorithm takes as input a secret key* sk*, a receiver double key* rdk *and a ciphertext* ct*, then outputs the anamorphic message* am*.*

Definition 14 (Correctness). *Let* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be some PKE scheme equipped with an anamorphic extension* $\Sigma = (\mathsf{aKGen}, \mathsf{aEnc}, \mathsf{aDec})$*. We say that* Σ *is correct if for any normal message* m *and anamorphic message* am*, there is*

$$\Pr\left[\mathsf{aDec}(\mathsf{sk}, \mathsf{rdk}, \mathsf{ct}) \neq \mathsf{am}\right] \leq \mathsf{negl}(\lambda),$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$*,* $(\mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(\mathsf{pk})$*,* $\mathsf{ct} \leftarrow \mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am})$*.*

Security against collusion (formerly CIND-xCPA1 [14]) in the setting of anamorphic extension is refined as follows, which we call *indistinguishability against collusion*. We emphasize that this single model subsumes the known notions of CPA-security for AAE and is strictly stronger, capturing security requirements which are not addressed by previous models. Further details can be found in Appendix A.

Definition 15 (Indistinguishability Against Collusion). *An anamorphic extension* $\Sigma = (\mathsf{aKGen}, \mathsf{aEnc}, \mathsf{aDec})$ *for a PKE scheme* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be indistinguishable against collusion, if for any PPT dictator (adversary)* D*, its advantage is negligible in the security parameter. In particular,*

$$\mathsf{Adv}^{\mathsf{IND\text{-}Collu}}_{\mathsf{D},\Sigma}(\lambda) = \left|\Pr\left[\mathsf{ExpReal}_{\mathsf{D}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ExpAnam}_{\mathsf{D}}(1^\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda),$$

where the experiments $\mathsf{ExpReal}$ *and* $\mathsf{ExpAnam}$ *are defined in Fig. 7.*

4.2 Witness PRF for PKE

We introduce *witness PRF for PKE*, the primitive that isolates an intermediate layer from the goal of designing asymmetric anamorphic extension.

The witness PRF for PKE is analogous to the notion of general witness PRF for arbitrary NP-relations [35], but is tailored to some relation associated with a given PKE scheme. Intuitively speaking, a witness PRF for some PKE scheme $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is constructed to provide two ways to compute the pseudorandom function value on some statement $\mathsf{x} = (\mathsf{m}, \mathsf{ct})$ consisting of a

$\mathsf{ExpReal}_{\mathsf{D}}(1^\lambda)$:

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(\mathsf{pk})$
3 **return** $\mathsf{D}^{\mathcal{O}(\cdot,\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$
4 **where** $\mathcal{O}(\cdot,\cdot)$ on input m, am:
5 **return** $\mathsf{Enc}(\mathsf{pk}, \mathsf{m})$

$\mathsf{ExpAnam}_{\mathsf{D}}(1^\lambda)$:

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(\mathsf{pk})$
3 **return** $\mathsf{D}^{\mathcal{O}(\cdot,\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$
4 **where** $\mathcal{O}(\cdot,\cdot)$ on input m, am:
5 **return** $\mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am})$

Fig. 7: Indistinguishability Against Collusion

message m and a ciphertext ct. On the one hand, given the secret function key fk, one can efficiently compute the function value on any pair $\mathsf{x} = (\mathsf{m}, \mathsf{ct})$. On the other hand, given only the public evaluation key ek, one can still efficiently compute the function value provided that the evaluator also knows encryption randomness s (namely witness) such that ct is an encryption of m (induced by another encryption algorithm wEnc).

The formal definition of a witness PRF for PKE is given below.

Definition 16 (Witness PRF for PKE). *Let* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a PKE scheme. A witness PRF* $\mathsf{wPRF} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wF}, \mathsf{wEval})$ *for* Π *consists of four PPT algorithms defined as follows.*

- $\mathsf{wGen}(\mathsf{pk}) \to (\mathsf{ek}, \mathsf{fk})$. *The key generation algorithm takes as input a public key* pk *under* Π*, then outputs an evaluation key* ek *and a function key* fk.
- $\mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mathsf{m}) \to (\mathsf{s}, \mathsf{ct})$. *The encryption algorithm (disregarding its additional output* s*, it may still differ from the original one* Enc*) takes as input a public key* pk*, an evaluation key* ek *and a message* m*, then outputs a ciphertext* ct *along with the related encryption randomness* s.
- $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, \mathsf{x} = (\mathsf{m}, \mathsf{ct})) \to \mathsf{y} \in \{0,1\}^\ell$. *The algorithm takes as input a secret key* sk*, a function key* fk*, and a statement* x *which contains a message* m *and a ciphertext* ct*, then calculates the function value* $\mathsf{y} \in \{0,1\}^\ell$ *on* x*, where* ℓ *is at most logarithmic in the security parameter.*
- $\mathsf{wEval}(\mathsf{ek}, \mathsf{s}, \mathsf{x} = (\mathsf{m}, \mathsf{ct})) \to \mathsf{y} \in \{0,1\}^\ell \cup \{\perp\}$. *The algorithm takes as input an evaluation key* ek*, an encryption randomness* s *(that is, witness) and a statement* $\mathsf{x} = (\mathsf{m}, \mathsf{ct})$*. It returns a value* y*, which is either the function value on* x *lying in* $\{0,1\}^\ell$*, or a special symbol* $\perp \notin \{0,1\}^\ell$.
  The purpose of this special symbol is to indicate that wEval *(if forced to output an evaluation result) may output a value inconsistent with the evaluation of* wF *on the current statement* x*. Consequently, we do not require* wEval *and* wF *to always agree (in the instantiation for Regev encryption), as long as such an event can be explicitly signaled.*

Naturally, we impose the correctness requirement only in the case where the output of wEval is *not* the special symbol $\perp$. The formal definition of correctness is given as follows.

Definition 17 (Correctness). *For* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{wGen}(\mathsf{pk})$, *for any* $\mathsf{m} \in \mathcal{M}_{\mathsf{pk}}$, *for* $(\mathsf{s}, \mathsf{ct}) \leftarrow \mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mathsf{m})$, *and for any* $\mathsf{y} \in \{0,1\}^\ell$, *there is*

$$\Pr\left[\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (\mathsf{m}, \mathsf{ct})) \neq \mathsf{y} \mid \mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct})) = \mathsf{y}\right] \leq \mathsf{negl}(\lambda),$$

where the probability is defined over the randomness of KGen, wGen, *and* wEnc.

Informally, the indistinguishability of a witness PRF for PKE requires that ciphertexts generated by wEnc and by Enc are indistinguishable, even in the presence of the secret key.

Definition 18 (Indistinguishability). *For any PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, *its advantage is negligible in the security parameter, specifically,*

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

where the experiment ExpIND *is defined in Fig. 8.*

$\mathsf{ExpIND}_{\mathcal{A},b}(1^\lambda)$:

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{wGen}(\mathsf{pk})$
3 $(\mathsf{m}, \mathsf{state}) \leftarrow \mathcal{A}_1(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
4 $\mathsf{ct}_0 \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
5 $(\mathsf{s}, \mathsf{ct}_1) \leftarrow \mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mathsf{m})$
6 **return** $\mathcal{A}_2(\mathsf{state}, \mathsf{ct}_b)$

Fig. 8: The Experiment ExpIND

Informally, the weak pseudorandomness of a witness PRF for PKE requires that the evaluations via wEval on statements $\mathsf{x} = (\mathsf{m}, \mathsf{ct})$, where for each pair ct is a random ciphertext of m generated by wEnc, are pseudorandom. Considering the output space of wEval, we define a distribution, termed as *uniform with failure*, as follows.

Definition 19 (Uniform with Failure). *The distribution called uniform with failure is defined over* $\{0,1\}^\ell \cup \{\perp\}$, *where* $\perp \notin \{0,1\}^\ell$. *We write it as* UF_p *for some probability* p. *In particular,* UF_p *outputs* $\perp$ *with probability* $1-p$, *and each* $a \in \{0,1\}^\ell$ *with probability* $p/2^\ell$ *(namely, with probability* p *proceeding to sample uniformly from* $\{0,1\}^\ell$*).*

Continuing the above discussion, the pseudorandomness of the evaluations on statements means that they are indistinguishable from samples drawn from UF_p. Moreover, to guarantee indistinguishability against collusion of our framework, we require that the probability that all attempts (within $\mathsf{poly}(\lambda)$ trials) to find a

suitable ciphertext for a covert message fail be negligible. Accordingly, p cannot be too small: it must be non-negligible.

The formal definition of weak pseudorandomness of a witness PRF for PKE is presented as follows.

Definition 20 (Weak Pseudorandomness). *We have two requirements for this property.*

1. *For any PPT adversary $\mathcal{A}$, its advantage is negligible in the security parameter as follows:*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{WPR}}_{\mathcal{A},\mathsf{wPRF}}(\lambda) &= \left|\Pr\left[\mathsf{ExpWPR}_{\mathcal{A},0}(1^\lambda)=1\right]-\Pr\left[\mathsf{ExpWPR}_{\mathcal{A},1}(1^\lambda)=1\right]\right| \\ &\leq \mathsf{negl}(\lambda),\end{aligned}$$

   where the experiment ExpWPR *is defined in Fig. 9.*
2. *The probability p in the distribution UF_p should satisfy $p \geq 1/\mathsf{poly}(\lambda)$.*

$\mathsf{ExpWPR}_{\mathcal{A},b}(1^\lambda)$:

1 $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{ek},\mathsf{fk}) \leftarrow \mathsf{wGen}(\mathsf{pk})$
3 **return** $\mathcal{A}^{\mathcal{O}_b(\cdot)}(\mathsf{pk},\mathsf{sk},\mathsf{ek})$
4 **where** $\mathcal{O}_b(\cdot)$ on input m:
5 $(\mathsf{s},\mathsf{ct}) \leftarrow \mathsf{wEnc}(\mathsf{pk},\mathsf{ek},\mathsf{m})$
6 $\mathsf{y}_0 \leftarrow \mathsf{wEval}(\mathsf{ek},\mathsf{s},(\mathsf{m},\mathsf{ct}))$, $\mathsf{y}_1 \hookleftarrow \mathsf{UF}_p$
7 **return** $(\mathsf{ct},\mathsf{y}_b)$

Fig. 9: The Experiment ExpWPR

4.3 Compiler

We show how to compile a witness PRF for any PKE scheme Π into an asymmetric anamorphic extension for Π. In particular, given some PKE scheme $\Pi = (\mathsf{KGen},\mathsf{Enc},\mathsf{Dec})$, if there exists a witness PRF $\mathsf{wPRF} = (\mathsf{wGen},\mathsf{wEnc},\mathsf{wF},\mathsf{wEval})$ for Π, then Π is asymmetric anamorphic. Moreover, an asymmetric anamorphic extension $\Sigma = (\mathsf{aKGen},\mathsf{aEnc},\mathsf{aDec})$ for Π can be constructed as follows.

- $\mathsf{aKGen}(\mathsf{pk}) \rightarrow (\mathsf{sdk},\mathsf{rdk})$. On input a public key pk, the anamorphic key generation algorithm runs $(\mathsf{ek},\mathsf{fk}) \leftarrow \mathsf{wGen}(\mathsf{pk})$, then returns the sender double key sdk as ek, and the receiver double key rdk as fk.
- $\mathsf{aEnc}(\mathsf{pk},\mathsf{sdk},\mathsf{m},\mathsf{am}) \rightarrow \mathsf{ct}$. Given a public key pk, a sender double key $\mathsf{sdk} = \mathsf{ek}$, a normal message m, and a covert message $\mathsf{am} \in \{0,1\}^\ell$, the anamorphic

encryption algorithm tries $v = \mathsf{poly}(\lambda)$ times: $(\mathsf{s}, \mathsf{ct}) \leftarrow \mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mathsf{m})$ and return ct when $\mathsf{am} = \mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct}))$.
If the number of attempts reaches the prescribed limit, we artificially set the output of aEnc to be the covert message am.
Such design is introduced to simplify the proof of standard correctness for the anamorphic mode. Besides, since the probability of reaching the limit is set to be negligible, this design does not compromise security, namely the indistinguishability against collusion. Accordingly, in the following aDec algorithm, whenever the ciphertext belongs to the covert message space $\{0,1\}^\ell$, it directly outputs that ciphertext as the covert message. Note that here we assume without loss of generality that ciphertexts generated by the wEnc algorithm do not belong to $\{0,1\}^\ell$, so in aDec the event that the input ciphertext ct is in $\{0,1\}^\ell$ indicates that ct itself is the covert message am.

- $\mathsf{aDec}(\mathsf{sk}, \mathsf{rdk}, \mathsf{ct}) \rightarrow \mathsf{am}$. On input a secret key sk, a receiver double key $\mathsf{rdk} = \mathsf{fk}$, and a ciphertext ct, the anamorphic decryption algorithm directly outputs $\mathsf{am} = \mathsf{ct}$ if $\mathsf{ct} \in \{0,1\}^\ell$, or otherwise first performs normal decryption as $\mathsf{m} = \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$, and then computes the anamorphic message as $\mathsf{am} = \mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$.

To simultaneously ensure correctness, that aEnc runs in PPT, and the indistinguishability against collusion, ℓ should be at most logarithmic in the security parameter. Moreover, we can set $v = \lambda \cdot 2^\ell / p$ which is a polynomial in λ.

Theorem 2. *Σ is an asymmetric anamorphic extension of Π.*

The full proof of this theorem is given in Appendix B.

4.4 On the Necessity of Witness PRF for PKE

In the converse direction, we show that a witness PRF for PKE is essential for an asymmetric anamorphic extension in our setting. Concretely, suppose that $\Sigma = (\mathsf{aKGen}, \mathsf{aEnc}, \mathsf{aDec})$ is an asymmetric anamorphic extension with covert message space $\{0,1\}^\ell$ for $\ell = O(\log \lambda)$, of a PKE scheme $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$. A witness PRF $\mathsf{wPRF} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wF}, \mathsf{wEval})$ for Π is shown as follows.

- $\mathsf{wGen}(\mathsf{pk}) \rightarrow (\mathsf{ek}, \mathsf{fk})$. On input a public key pk, the key generation algorithm invokes $(\mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(\mathsf{pk})$, and then returns the pair of evaluation and function keys as $(\mathsf{ek}, \mathsf{fk}) = (\mathsf{sdk}, \mathsf{rdk})$.
- $\mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mathsf{m}) \rightarrow (\mathsf{s}, \mathsf{ct})$. Given a public key pk, an evaluation key $\mathsf{ek} = \mathsf{sdk}$ and a message m, the (modified) encryption algorithm samples $\mathsf{am} \xleftarrow{\$} \{0,1\}^\ell$, then invokes $\mathsf{ct} \leftarrow \mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am}; \mathsf{r})$, where r denotes the internal coin tosses. The algorithm returns $\mathsf{s} = (\mathsf{am}, \mathsf{r})$ and ct.
- $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (\mathsf{m}, \mathsf{ct})) \rightarrow \mathsf{y}$. On input a secret key sk, a function key $\mathsf{fk} = \mathsf{rdk}$, a message m and a ciphertext ct, the algorithm runs $\mathsf{am} \leftarrow \mathsf{aDec}(\mathsf{sk}, \mathsf{rdk}, \mathsf{ct})$, then returns $\mathsf{y} = \mathsf{am}$.
- $\mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct})) \rightarrow \mathsf{y}$. For $\mathsf{s} = (\mathsf{am}, \mathsf{r})$, the algorithm returns $\mathsf{y} = \mathsf{am}$.

Theorem 3. *wPRF is a witness PRF for Π.*

The concrete proof of this theorem is deferred to Appendix C.

5 Instantiations

In this section, we present several constructions of witness PRF for PKE, each of which yields a corresponding asymmetric anamorphic extension through our compiler. Our first instantiation is a generic construction for any PKE scheme with high min-entropy ciphertexts. We also give practical constructions for specific PKE schemes, including ElGamal, Regev, and Linear encryption, without relying on heavy cryptographic mechanisms.

5.1 Generic Construction from Any PKE with High Min-Entropy Ciphertexts

Given any IND-CPA secure PKE scheme $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ with high min-entropy ciphertexts, we define $\kappa = \mathsf{H}_\infty(\mathsf{Enc}) = \Omega(\lambda)$. Let $\mathsf{G} : \{0,1\}^\sigma \to \{0,1\}^\tau$ be a PRG whose seed length is $\sigma = \kappa/2$ and whose output space $\{0,1\}^\tau$ is the random coin space of Enc. Suppose further that $\mathsf{WPRF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Eval})$ is a general witness PRF [35] with output space $\mathcal{Y} = \{0,1\}^\ell$, where $\ell = O(\log \lambda)$. Our witness PRF $\mathsf{wPRF}_{\mathsf{pke}} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wF}, \mathsf{wEval})$ for Π is as follows.

- $\mathsf{wGen}(\mathsf{pk}) \to (\mathsf{ek}, \mathsf{fk})$. Define a relation circuit $R_{\mathsf{pk}} : \{0,1\}^\sigma \times (\mathcal{M}_{\mathsf{pk}} \times \mathcal{C}_{\mathsf{pk}}) \to \{0,1\}$ such that $R_{\mathsf{pk}}(\mathsf{s}, (\mathsf{m}, \mathsf{ct})) = 1$ if and only if $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$, where $\mathcal{M}_{\mathsf{pk}}$ and $\mathcal{C}_{\mathsf{pk}}$ denote the message space and ciphertext space (related to pk) respectively. Then run $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$ and return $(\mathsf{ek}, \mathsf{fk})$.
- $\mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mathsf{m}) \to (\mathsf{s}, \mathsf{ct})$. Sample $\mathsf{s} \xleftarrow{\$} \{0,1\}^\sigma$, then return s and $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$.
- $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (\mathsf{m}, \mathsf{ct})) \to \mathsf{y}$. Return $\mathsf{y} = \mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$.
- $\mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct})) \to \mathsf{y}$. Return $\mathsf{y} = \mathsf{Eval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct}))$.

Theorem 4. $\mathsf{wPRF}_{\mathsf{pke}}$ *is a witness PRF for the PKE scheme* Π.

The proof of this theorem is deferred to Appendix D.

5.2 From ElGamal Encryption

We briefly recall ElGamal encryption scheme [19] defined over a cyclic group $\mathbb{G}$ of prime order q with generator $g \in \mathbb{G}$ as follows:

- $\mathsf{KGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$. Sample $x \xleftarrow{\$} \mathbb{Z}_q$, and compute $h = g^x$. Return the public key pk and secret key sk as

$$\mathsf{pk} = h, \quad \mathsf{sk} = x.$$

- $\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$. To encrypt a message $m \in \mathbb{G}$, sample an encryption randomness $s \xleftarrow{\$} \mathbb{Z}_q$, and compute the ciphertext ct as

$$\mathsf{ct} = (g^s, h^s \cdot m)\,.$$

- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to m$. On input a secret key $\mathsf{sk} = x$ and a ciphertext $\mathsf{ct} = (c_0, c_1)$, recover the message as $m = c_1/c_0^x$.

We use the function $\mathsf{bin}(a)$ to denote the binary representation of an element $a \in \mathbb{G}$. Moreover, we assume that the binary encodings of all elements in $\mathbb{G}$ have a common length $n = \mathsf{poly}(\lambda)$; that is, $\mathsf{bin}(a) \in \{0,1\}^n$ for every $a \in \mathbb{G}$. Our witness PRF $\mathsf{wPRF}_{\mathsf{elg}} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wF}, \mathsf{wEval})$ for ElGamal encryption is defined as follows.

- $\mathsf{wGen}(\mathsf{pk}) \to (\mathsf{ek}, \mathsf{fk})$. Sample $\widehat{x} \xleftarrow{\$} \mathbb{Z}_q$ and compute $\widehat{h} = g^{\widehat{x}}$. Sample a matrix $\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$. Return the evaluation key ek and function key fk as

$$\mathsf{ek} = \left(\widehat{h}, \mathbf{A}\right), \quad \mathsf{fk} = (\widehat{x}, \mathbf{A}).$$

- $\mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, m) \to (s, \mathsf{ct})$. Sample a randomness $s \xleftarrow{\$} \mathbb{Z}_q$, then return s and $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, m; s)$.
- $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (m, \mathsf{ct})) \to \mathsf{y} \in \{0,1\}^\ell$. Given $\mathsf{fk} = (\widehat{x}, \mathbf{A})$ and $\mathsf{ct} = (c_0, c_1)$, calculate $r = c_0^{\widehat{x}}$, then return y as

$$\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(r) \bmod 2.$$

- $\mathsf{wEval}(\mathsf{ek}, s, (m, \mathsf{ct})) \to \mathsf{y} \in \{0,1\}^\ell$. For $\mathsf{ek} = \left(\widehat{h}, \mathbf{A}\right)$, compute $r = \widehat{h}^s$, and return y as

$$\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(r) \bmod 2.$$

Theorem 5. $\mathsf{wPRF}_{\mathsf{elg}}$ *is a witness PRF for ElGamal encryption.*

The concrete proof of this theorem is deferred to Appendix E.

5.3 From Regev Encryption

We briefly recall Regev encryption scheme [28] for a prime modulus q as follows.

- $\mathsf{KGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$, and $\mathbf{e} \hookleftarrow \chi^m$. Compute $\mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e} \in \mathbb{Z}_q^m$. Return the public key pk and secret key sk as

$$\mathsf{pk} = \begin{bmatrix} \mathbf{A} \\ \mathbf{b}^t \end{bmatrix}, \quad \mathsf{sk} = \mathbf{x}.$$

- $\mathsf{Enc}(\mathsf{pk}, \mu) \to \mathsf{ct}$. To encrypt a message $\mu \in \{0,1\}$ by a public key $\mathsf{pk} = \begin{bmatrix} \mathbf{A} \\ \mathbf{b}^t \end{bmatrix}$, sample $\mathbf{s} \xleftarrow{\$} \{0,1\}^m$, and compute the ciphertext ct as

$$\mathsf{ct} = \begin{bmatrix} \mathbf{A} \\ \mathbf{b}^t \end{bmatrix} \mathbf{s} + \begin{bmatrix} \mathbf{0} \\ \mu \cdot \lfloor \frac{q}{2} \rceil \end{bmatrix} = \begin{bmatrix} \mathbf{As} \\ \mathbf{b}^t\mathbf{s} + \mu \cdot \lfloor \frac{q}{2} \rceil \end{bmatrix} \in \mathbb{Z}_q^n \times \mathbb{Z}_q.$$

- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to \mu$. To decrypt a ciphertext $\mathsf{ct} = \begin{bmatrix} \mathbf{c} \\ c \end{bmatrix}$ by the secret key $\mathsf{sk} = \mathbf{x}$, compute $\mu' = c - \mathbf{x}^t\mathbf{c} \in \mathbb{Z}_q$. Output 0 if μ' is closer to 0 than $\lfloor \frac{q}{2} \rceil$, or output 1 otherwise.

For $\mathbb{Z}/q\mathbb{Z}$, we use the representative set $\mathbb{Z}_q = \left[-\frac{q-1}{2}, \frac{q-1}{2}\right]$. We write $\mathsf{sign}(a)$ for the sign of an element $a \in \mathbb{Z}_q$: $\mathsf{sign}(a) = 0$ if $a \leq 0$, and $\mathsf{sign}(a) = 1$ if $a > 0$. Our witness PRF $\mathsf{wPRF}_{\mathsf{reg}}$ for Regev encryption is defined as follows.

- $\mathsf{wGen}(\mathsf{pk}) \to (\mathsf{ek}, \mathsf{fk})$. Given public key $\mathsf{pk} = \begin{bmatrix} \mathbf{A} \\ \mathbf{b}^t \end{bmatrix} \in \mathbb{Z}_q^{(n+1)\times m}$, sample $\widehat{\mathbf{x}} \xleftarrow{\$} \mathbb{Z}_q^n$, and $\widehat{\mathbf{e}} \hookleftarrow \chi^m$, then compute $\widehat{\mathbf{b}} = \mathbf{A}^t\widehat{\mathbf{x}} + \widehat{\mathbf{e}} \in \mathbb{Z}_q^m$. Return the evaluation key ek and function key fk as
$$\mathsf{ek} = \widehat{\mathbf{b}}, \quad \mathsf{fk} = \widehat{\mathbf{x}}.$$
- $\mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mu) \to (\mathbf{s}, \mathsf{ct})$. Sample a randomness $\mathbf{s} \xleftarrow{\$} \{0,1\}^m$, and then return $\mathbf{s}$ along with $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mu; \mathbf{s})$.
- $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (\mu, \mathsf{ct})) \to y$. For function key $\mathsf{fk} = \widehat{\mathbf{x}}$ and ciphertext $\mathsf{ct} = \begin{bmatrix} \mathbf{c} \\ c \end{bmatrix} \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, compute $r' = \widehat{\mathbf{x}}^t\mathbf{c} \in \mathbb{Z}_q$, and return the function value y on ct as
$$y = \mathsf{sign}(r') \in \{0,1\}.$$
- $\mathsf{wEval}(\mathsf{ek}, \mathbf{s}, (\mu, \mathsf{ct})) \to y$. For $\mathsf{ek} = \widehat{\mathbf{b}}$, compute $r = \widehat{\mathbf{b}}^t\mathbf{s} \in \mathbb{Z}_q$. If $\lfloor \frac{q}{8} \rfloor \leq |r| \leq 3\lfloor \frac{q}{8} \rfloor + 1$, return $y = \mathsf{sign}(r) \in \{0,1\}$. Else, return $\perp$.

We choose the parameters as follows.

- To guarantee that Regev's reduction applies: $\alpha \cdot q > 2\sqrt{n}$.
- To satisfy correctness, we set $q \geq 10m$ and $\alpha \leq \frac{1}{\sqrt{m}\omega(\sqrt{\log m})}$.
- To ensure that the generalized leftover hash lemma (Lemma 4) applies: $m > (n+2)\log_2 q + \omega(\log(n+1))$. The lemma is used in the proof of Theorem 6.

Theorem 6. $\mathsf{wPRF}_{\mathsf{reg}}$ *is a witness PRF for Regev encryption.*

The proof of this theorem is deferred to Appendix F.

5.4 From Linear Encryption

We briefly recall the Linear encryption scheme [7], defined over a cyclic group $\mathbb{G}$ of prime order q as follows:

- $\mathsf{KGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$. Sample $h \xleftarrow{\$} \mathbb{G}$ and exponents $x, y \xleftarrow{\$} \mathbb{Z}_q^*$. Set $u = h^{x^{-1}}$ and $v = h^{y^{-1}}$, where the inverses are taken modulo q, so that $u^x = v^y = h$. Return the public key pk and secret key sk as
$$\mathsf{pk} = (u, v, h), \quad \mathsf{sk} = (x, y).$$

- $\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$. To encrypt a message $m \in \mathbb{G}$, sample encryption randomness $r, s \xleftarrow{\$} \mathbb{Z}_q$, and compute the ciphertext ct as

$$\mathsf{ct} = (c_0, c_1, c_2) = \left(u^r, v^s, h^{r+s} \cdot m\right).$$

- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to m$. On input a secret key $\mathsf{sk} = (x, y)$ and a ciphertext $\mathsf{ct} = (c_0, c_1, c_2)$, recover the message as $m = c_2/(c_0^x c_1^y)$.

As in our construction for the ElGamal encryption, we write $\mathsf{bin}(u) \in \{0,1\}^n$ for the binary representation of any group element $u \in \mathbb{G}$, where $n = \mathsf{poly}(\lambda)$. Formally, our witness PRF $\mathsf{wPRF}_{\mathsf{le}}$ for Linear encryption is defined as follows.

- $\mathsf{wGen}(\mathsf{pk}) \to (\mathsf{ek}, \mathsf{fk})$. Given $\mathsf{pk} = (u, v, h)$, sample $\widehat{a}, \widehat{b} \xleftarrow{\$} \mathbb{Z}_q$ and compute

$$\widehat{v} = v^{\widehat{a}}, \quad \widehat{h} = h^{\widehat{a}}, \quad \widehat{H} = h^{\widehat{a}\widehat{b}}.$$

  Sample a matrix $\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$. Return the evaluation key ek and function key fk as

$$\mathsf{ek} = (\widehat{v}, \widehat{h}, \widehat{H}, \mathbf{A}), \quad \mathsf{fk} = (\widehat{a}, \widehat{b}, \mathbf{A}).$$

- $\mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, m) \to ((r, s), \mathsf{ct})$. Given $\mathsf{pk} = (u, v, h)$ and $\mathsf{ek} = (\widehat{v}, \widehat{h}, \widehat{H}, \mathbf{A})$, sample $r, s \xleftarrow{\$} \mathbb{Z}_q$ and compute

$$\mathsf{ct} = (c_0, c_1, c_2) = \left(u^r, \widehat{v}^s, h^r \cdot \widehat{h}^s \cdot m\right).$$

  Return (r, s) and ct.
- $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (m, \mathsf{ct})) \to \mathsf{y} \in \{0,1\}^\ell$. Given $\mathsf{sk} = (x, y)$, $\mathsf{fk} = (\widehat{a}, \widehat{b}, \mathbf{A})$, and $\mathsf{ct} = (c_0, c_1, c_2)$, compute the group element R as

$$R = \left((c_0^x)^{\widehat{a}} \cdot c_1^y\right)^{\widehat{b}},$$

  and return

$$\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(R) \bmod 2.$$

- $\mathsf{wEval}(\mathsf{ek}, (r, s), (m, \mathsf{ct})) \to \mathsf{y} \in \{0,1\}^\ell$. Given $\mathsf{ek} = (\widehat{v}, \widehat{h}, \widehat{H}, \mathbf{A})$, compute the group element as

$$R = \widehat{H}^{r+s},$$

  and return

$$\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(R) \bmod 2.$$

Theorem 7. $\mathsf{wPRF}_{\mathsf{le}}$ *is a witness PRF for the Linear encryption.*

The concrete proof of this theorem is deferred to Appendix G.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H) IBE in the standard model. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 553–572. Springer (2010)
2. Avitabile, G., Botta, V., Giunta, E., Mielniczuk, M., Migliaro, F.: The malice of ELFs: practical anamorphic-resistant encryption without random oracles. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 303–333. Springer (2026)
3. Banerjee, S., Pal, T., Rupp, A., Slamanig, D.: Simple asymmetric anamorphic encryption and signature using multi-message extensions. In: Annual International Cryptology Conference. pp. 432–464. Springer (2026)
4. Banfi, F., Gegier, K., Hirt, M., Maurer, U., Rito, G.: Anamorphic encryption, revisited. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 3–32. Springer (2024)
5. Beullens, W., Dobson, S., Katsumata, S., Lai, Y.F., Pintore, F.: Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 95–126. Springer (2022)
6. Blaze, M.: Protocol failure in the escrowed encryption standard. In: Proceedings of the 2nd ACM Conference on Computer and Communications Security. pp. 59–67 (1994)
7. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Annual international cryptology conference. pp. 41–55. Springer (2004)
8. Boneh, D., Shoup, V.: A graduate course in applied cryptography (Jan 2023), version 0.6
9. Canetti, R., Dwork, C., Naor, M., Ostrovsky, R.: Deniable encryption. In: Annual International Cryptology Conference. pp. 90–104. Springer (1997)
10. Carnemolla, D., Catalano, D., Giunta, E., Migliaro, F.: Anamorphic resistant encryption: the good, the bad and the ugly. In: Annual International Cryptology Conference. pp. 472–503. Springer (2025)
11. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: International conference on the theory and application of cryptology and information security. pp. 395–427. Springer (2018)
12. Catalano, D., Giunta, E., Migliaro, F.: Anamorphic encryption: New constructions and homomorphic realizations. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 33–62. Springer (2024)
13. Catalano, D., Giunta, E., Migliaro, F.: Generic anamorphic encryption, revisited: New limitations and constructions. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 275–303. Springer (2025)
14. Choi, W., Collins, D., Liu, X., Stracovsky, R., Zikas, V.: A unified treatment of anamorphic encryption. In: Annual International Cryptology Conference. pp. 465–496. Springer (2026)
15. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Annual international cryptology conference. pp. 13–25. Springer (1998)
16. Deo, A., Libert, B.: Fully asymmetric anamorphic homomorphic encryption from LWE. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 609–639. Springer (2026)

17. Do, X.T., Persiano, G., Phan, D.H., Yung, M.: Randomness-recovery trapdoors: a new methodology for enhancing anamorphic encryption. Cryptology ePrint Archive (2026)
18. Dodis, Y., Goldin, E.: Anamorphic-resistant encryption; or why the encryption debate is still alive. In: Annual International Cryptology Conference. pp. 440–471. Springer (2025)
19. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE transactions on information theory **31**(4), 469–472 (1985)
20. Frankel, Y., Yung, M.: Escrow encryption systems visited: attacks, analysis and designs. In: Annual International Cryptology Conference. pp. 222–235. Springer (1995)
21. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the fortieth annual ACM symposium on Theory of computing. pp. 197–206 (2008)
22. Horel, T., Park, S., Richelson, S., Vaikuntanathan, V.: How to subvert backdoored encryption: Security against adversaries that decrypt all ciphertexts. 10th Innovations in Theoretical Computer Science (2019)
23. Kutylowski, M., Persiano, G., Phan, D.H., Yung, M., Zawada, M.: The self-anticensorship nature of encryption: On the prevalence of anamorphic cryptography. Proceedings on Privacy Enhancing Technologies **2023**(4), 170–183 (2023)
24. Micali, S.: Fair public-key cryptosystems. In: Annual International Cryptology Conference. pp. 113–138. Springer (1992)
25. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of the twenty-second annual ACM symposium on Theory of computing. pp. 427–437 (1990)
26. Persiano, G., Phan, D.H., Yung, M.: Anamorphic encryption: private communication against a dictator. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 34–63. Springer (2022)
27. Persiano, G., Phan, D.H., Yung, M.: Public-key anamorphism in (CCA-secure) public-key encryption and beyond. In: Annual International Cryptology Conference. pp. 422–455. Springer (2024)
28. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM) **56**(6), 1–40 (2009)
29. Rivest, R.L., et al.: Chaffing and winnowing: Confidentiality without encryption. CryptoBytes (RSA laboratories) **4**(1), 12–17 (1998)
30. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Proceedings of the forty-sixth annual ACM symposium on Theory of computing. pp. 475–484 (2014)
31. Von Ahn, L., Hopper, N.J.: Public-key steganography. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 323–341. Springer (2004)
32. Young, A., Yung, M.: The dark side of "black-box" cryptography or: Should we trust capstone? In: Annual International Cryptology Conference. pp. 89–103. Springer (1996)
33. Young, A., Yung, M.: The prevalence of kleptographic attacks on discrete-log based cryptosystems. In: Annual International Cryptology Conference. pp. 264–276. Springer (1997)
34. Young, A., Yung, M.: Auto-recoverable auto-certifiable cryptosystems. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 17–31. Springer (1998)

35. Zhandry, M.: How to avoid obfuscation using witness PRFs. In: Theory of Cryptography Conference. pp. 421–448. Springer (2015)

A Asymmetric Anamorphic Encryption, Revisited

In this section, we review the original definition of AAE, explain the incompleteness of existing security models, and motivate our use of anamorphic extensions to avoid it. Moreover, we show the conciseness of our approach: apart from the basic CPA-security required to the underlying PKE scheme, we need only one additional security model (namely, indistinguishability against collusion) to capture all known CPA-security requirements for AAE.

We first give the original definition of asymmetric anamorphic triplets.

Definition 21 (Asymmetric Anamorphic Triplets). *Given a CPA-secure PKE scheme* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$*, we define an asymmetric anamorphic triplet* $\Sigma = (\mathsf{aKGen}, \mathsf{aEnc}, \mathsf{aDec})$ *for* Π *by three efficient algorithms as follows.*

- $\mathsf{aKGen}(1^\lambda) \to (\mathsf{apk}, \mathsf{ask}, \mathsf{sdk}, \mathsf{rdk})$*. The anamorphic key generation algorithm takes as input a security parameter* λ*, then outputs* $(\mathsf{apk}, \mathsf{ask}, \mathsf{sdk}, \mathsf{rdk})$ *consisting of an anamorphic public key, an anamorphic secret key, a sender double key and a receiver double key.*
- $\mathsf{aEnc}(\mathsf{apk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am}) \to \mathsf{ct}$*. The anamorphic encryption algorithm takes as input an anamorphic public key* apk*, a sender double key* sdk*, a normal message* m*, and an anamorphic (covert) message* am*. It outputs an anamorphic ciphertext* ct*.*
- $\mathsf{aDec}(\mathsf{ask}, \mathsf{rdk}, \mathsf{ct}) \to \mathsf{am}$*. The anamorphic decryption algorithm takes as input an anamorphic secret key* ask*, a receiver double key* rdk *and a ciphertext* ct*, then outputs the anamorphic message* am*.*

Definition 22 (Correctness). *Let* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be some PKE scheme equipped with an anamorphic triplet* $\Sigma = (\mathsf{aKGen}, \mathsf{aEnc}, \mathsf{aDec})$*. We say that* Σ *is correct if for any normal message* m *and anamorphic message* am*, there is*

$$\Pr\left[\mathsf{aDec}(\mathsf{ask}, \mathsf{rdk}, \mathsf{ct}) \neq \mathsf{am}\right] \leq \mathsf{negl}(\lambda),$$

where $(\mathsf{apk}, \mathsf{ask}, \mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(1^\lambda)$*,* $\mathsf{ct} \leftarrow \mathsf{aEnc}(\mathsf{apk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am})$*, and the probability is taken over the randomness of* aKGen *and* aEnc*.*

At present, the strongest approach to defining CPA-security for AAE is due to [14], which consists of three security models informally described as below.

- STE-xCPA1: This is the most basic security requirement, which states that a dictator D with access to the secret key sk cannot distinguish the anamorphic mode from the normal mode.
- CIND-xCPA1: This model ensures that, a dictator D who possesses the secret key sk and additionally obtains the sender double key sdk via collusion with an anamorphic sender cannot distinguish ciphertexts normally generated by Enc from those generated via aEnc.

- SIND-nCPA: This model captures the privacy of normal messages in normal ciphertexts (generated by Enc), against an anamorphic sender aS who knows the sender double key sdk.

A limitation of the above models is that, in the collusion setting, they do not exclude the possibility that, once D is given sdk, it can infer that the receiver has indeed deployed the anamorphic mode, rather than suspecting that sdk was merely forged by the corrupted anamorphic sender. This can be illustrated by a simple example. Specifically, consider an asymmetric anamorphic triplet for some PKE which satisfies the three security models discussed above. We then artificially construct another anamorphic triplet that is essentially identical to the original one, except that now the sender double key sdk' consists of the original sender double key together with a hash value of the secret key sk, namely, $\mathsf{sdk}' = (\mathsf{sdk}, H(\mathsf{sk}))$. It is easy to verify that, when H is modeled as a random oracle, this artificially constructed anamorphic triplet still satisfies the above three security models. Nevertheless, once D is given sdk', it can check whether the second component is indeed the hash of sk. This enables it to judge whether the anamorphic mode has been deployed, since this hash value cannot be forged by the anamorphic sender.

This incompleteness does not arise in the notion of anamorphic extensions introduced by [4]. For the setting of AAE, the anamorphic extension indicates that a pair of sender/receiver double keys can be generated from a normal public key, which serves directly as the anamorphic public key for covert communications. This property can provide some degree of deniability. Returning to the issue raised earlier, deniability intuitively means that if an anamorphic sender presents a sender double key to the dictator, the receiver may claim that the key was generated by the sender itself (through the property of extension), a situation the dictator may be unable to exclude. This motivates our use of anamorphic extensions in this work.

To conclude this section, we explain how our approach subsumes all the security requirements captured by the three security models discussed above. Firstly, indistinguishability against collusion directly implies both STE-xCPA1 and CIND-xCPA1. Next, SIND-nCPA follows from the CPA-security of the underlying PKE scheme, where essentially, since we adopt the notion of anamorphic extension, the sender double key derivable from the public key does not compromise the semantic security of normal ciphertexts (against an anamorphic sender who holds the sender double key but not the secret key).

B Proof of Theorem 2

The proof is divided into two parts. We first establish correctness and then prove indistinguishability against collusion.

Correctness. From aEnc and aDec, the event that $\mathsf{aDec}(\mathsf{sk}, \mathsf{rdk}, \mathsf{ct}) \neq \mathsf{am}$ happens only when there is some trial in aEnc indexed by $i \in [1, v]$ where $(\mathsf{s}, \mathsf{ct}) \leftarrow \mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mathsf{m})$ and $\mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct})) = \mathsf{am} \neq \perp$, such that $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \neq \mathsf{m}$

or $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (\mathsf{m}, \mathsf{ct})) \neq \mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct}))$. Thus, from the correctness of Π and wPRF, we have

$$\Pr[\mathsf{aDec}(\mathsf{sk}, \mathsf{rdk}, \mathsf{ct}) \neq \mathsf{am}] \leq v \cdot \big(\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \neq \mathsf{m}] + \Pr[\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (\mathsf{m}, \mathsf{ct})) \neq \mathsf{am} \mid \mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct})) = \mathsf{am}]\big) \leq \mathsf{negl}(\lambda).$$

Indistinguishability Against Collusion. We define the following sequence of games between a challenger $\mathcal{C}$ and an adversary (the dictator) D. We assume that D is allowed to make at most $Q = \mathsf{poly}(\lambda)$ oracle queries. For convenience, the game-hopping argument is illustrated in Fig. 10, where the modifications introduced in each game relative to the previous one are highlighted.

- **Game 0:** This is the original anamorphic game $\mathsf{ExpAnam}_{\mathsf{D}}$, defined on the right-hand side of Fig. 7.
- **Game 1:** This game is identical to Game 0, except that, in Step 13, each invocation of wEval is replaced by an independent fresh sample from UF_p.
- **Game 2:** This game is identical to Game 1, except that, in Step 12, the ciphertext ct is generated by the original encryption algorithm Enc.
- **Game 3:** This game is identical to Game 2, except that, in Step 11, the oracle returns a fresh ciphertext ct generated by Enc.
- **Game 4:** This game is identical to Game 3, except that we remove the condition for returning a ciphertext. Consequently, in this game, the oracle invokes $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$ once and directly returns ct as its response. Note that this game is exactly the real game $\mathsf{ExpReal}_{\mathsf{D}}$, defined on the left-hand side of Fig. 7.

For each $\xi \in [0, 4]$, let W_ξ denote the event that D outputs 1 in Game ξ. We now state the following claims.

Claim 1 $|\Pr[W_0] - \Pr[W_1]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim follows from the weak pseudorandomness of wPRF. In particular, we construct a simulator $\mathcal{B}$ against this property from the adversary D. The simulator $\mathcal{B}$ proceeds exactly as the challenger $\mathcal{C}$ in Game 0, except for the following modifications: (1) the keys $(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$ are generated by $\mathcal{B}$'s challenger for weak pseudorandomness, denoted by $\mathcal{C}_{\mathsf{WPR}}$; and (2) each invocation of wEnc in Step 12 and of wEval in Step 13 is replaced by a corresponding query to the oracle $\mathcal{O}(\cdot)$ provided by $\mathcal{C}_{\mathsf{WPR}}$. For clarity, we describe the behavior of $\mathcal{B}$ in Fig. 11, where the changes relative to Game 0 are highlighted.

Note that if y is computed as $\mathsf{y} \leftarrow \mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct}))$, then $\mathcal{B}$ simulates Game 0, whereas if y is sampled as $\mathsf{y} \hookleftarrow \mathsf{UF}_p$, then $\mathcal{B}$ simulates Game 1. Therefore, by the weak pseudorandomness of wPRF, we have

$$\begin{aligned} |\Pr[W_0] - \Pr[W_1]| &= \left|\Pr\left[\mathsf{ExpWPR}_{\mathcal{B},0}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ExpWPR}_{\mathcal{B},1}(1^\lambda) = 1\right]\right| \\ &= \mathsf{Adv}^{\mathsf{WPR}}_{\mathcal{B},\mathsf{wPRF}}(\lambda) \leq \mathsf{negl}(\lambda), \end{aligned}$$

and thus the claim follows. □

Game 0

```
1  (pk, sk) ← KGen(1^λ)
2  (ek, fk) ← wGen(pk)
3  i' ← 1
4  return D^{O(·,·)}(pk, sk, ek)
5    where O(·,·) on input m, am:
6      if i' > Q
7        return ⊥
8      j' ← 1
9      if j' > v
10       i' ← i' + 1
11       return am
12     (s, ct) ← wEnc(pk, ek, m)
13     if am = wEval(ek, s, (m, ct))
14       i' ← i' + 1
15       return ct
16     else
17       j' ← j' + 1
18       go to Step 9
```

Game 1

```
1  (pk, sk) ← KGen(1^λ)
2  (ek, fk) ← wGen(pk)
3  i' ← 1
4  return D^{O(·,·)}(pk, sk, ek)
5    where O(·,·) on input m, am:
6      if i' > Q
7        return ⊥
8      j' ← 1
9      if j' > v
10       i' ← i' + 1
11       return am
12     (s, ct) ← wEnc(pk, ek, m)
13     if am = r ↩ UF_p
14       i' ← i' + 1
15       return ct
16     else
17       j' ← j' + 1
18       go to Step 9
```

Game 2

```
1  (pk, sk) ← KGen(1^λ)
2  (ek, fk) ← wGen(pk)
3  i' ← 1
4  return D^{O(·,·)}(pk, sk, ek)
5    where O(·,·) on input m, am:
6      if i' > Q
7        return ⊥
8      j' ← 1
9      if j' > v
10       i' ← i' + 1
11       return am
12     ct ← Enc(pk, m)
13     if am = r ↩ UF_p
14       i' ← i' + 1
15       return ct
16     else
17       j' ← j' + 1
18       go to Step 9
```

Game 3

```
1  (pk, sk) ← KGen(1^λ)
2  (ek, fk) ← wGen(pk)
3  i' ← 1
4  return D^{O(·,·)}(pk, sk, ek)
5    where O(·,·) on input m, am:
6      if i' > Q
7        return ⊥
8      j' ← 1
9      if j' > v
10       i' ← i' + 1
11       return ct ← Enc(pk, m)
12     ct ← Enc(pk, m)
13     if am = r ↩ UF_p
14       i' ← i' + 1
15       return ct
16     else
17       j' ← j' + 1
18       go to Step 9
```

Game 4

```
1  (pk, sk) ← KGen(1^λ)
2  (ek, fk) ← wGen(pk)
3  i' ← 1
4  return D^{O(·,·)}(pk, sk, ek)
5    where O(·,·) on input m, am:
6      if i' > Q
7        return ⊥
8      i' ← i' + 1
9      return ct ← Enc(pk, m)
10
11
12
13
14
15
16
17
18
```

Fig. 10: Indistinguishability Against Collusion of Framework

Simulator $\mathcal{B}$:

```
1   (pk, sk, ek) ← C_WPR(1^λ)
2
3   i' ← 1
4   return D^{O(·,·)}(pk, sk, ek)
5     where O(·,·) on input m, am:
6       if i' > Q
7         return ⊥
8       j' ← 1
9       if j' > v
10        i' ← i' + 1
11        return am
12      (ct, y) ← C_WPR.O(m)
13      if am = y
14        i' ← i' + 1
15        return ct
16      else
17        j' ← j' + 1
18        go to Step 9
```

Fig. 11: The Simulator for Proving Game 0 $\approx$ Game 1

Claim 2 $|\Pr[W_1] - \Pr[W_2]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim is based on the indistinguishability of wPRF. To prove it, we define a sequence of games interpolating between Game 1 and Game 2, denoted by $\mathsf{G}_{(i,j)}$ for $(i,j) \in ([1,Q]\times[1,v])\cup\{(Q+1,1)\}$. In each game $\mathsf{G}_{(i,j)}$, we assign an index $(i',j') \in [1,Q]\times[1,v]$ to every attempt to find a valid ciphertext (see Steps 12 – 15 below, or the corresponding steps in Game 1 and Game 2), where i' denotes the i'-th oracle query and j' denotes the j'-th attempt within that query. We further equip these indices with the natural lexicographic order, namely, $(i',j') < (i,j)$ if and only if either $i' < i$, or $i' = i$ and $j' < j$. For convenience, we define the successor of an index by

$$(i',j')+1 := \begin{cases} (i',j'+1), & \text{if } j' < v, \\ (i'+1,1), & \text{if } j' = v. \end{cases}$$

- $\mathsf{G}_{(i,j)}$: This game is identical to Game 1, except that we modify the way in which $\mathcal{C}$ generates ciphertexts in Step 12. Specifically, for each attempt indexed by (i',j') with $(i',j') < (i,j)$, $\mathcal{C}$ uses Enc, whereas for every remaining attempt with index $(i',j') \geq (i,j)$, it uses wEnc. A detailed description of $\mathsf{G}_{(i,j)}$ is given in Fig. 12, where the changes relative to Game 1 are highlighted.

$\mathsf{G}_{(i,j)}$:

```
1   (pk, sk) ← KGen(1^λ)
2   (ek, fk) ← wGen(pk)
3   i' ← 1
4   return D^{O(·,·)}(pk, sk, ek)
5     where O(·,·) on input m, am:
6       if i' > Q
7         return ⊥
8       j' ← 1
9       if j' > v
10        i' ← i' + 1
11        return am
12      if (i', j') < (i, j)
          ct ← Enc(pk, m)
        else if (i', j') ≥ (i, j)
          (s, ct) ← wEnc(pk, ek, m)
13      if am = r ↩ UF_p
14        i' ← i' + 1
15        return ct
16      else
17        j' ← j' + 1
18        go to Step 9
```

Fig. 12: Game Hopping Between Game 1 and Game 2

Note that $\mathsf{G}_{(1,1)}$ is the same as Game 1, and that $\mathsf{G}_{(Q+1,1)} = \mathsf{G}_{(Q,v)+1}$ is identical to Game 2. Let $X_{(i,j)}$ denote the event that D outputs 1 in $\mathsf{G}_{(i,j)}$. We next show that, for every $(i,j) \in [1,Q] \times [1,v]$,

$$\left|\Pr\left[X_{(i,j)}\right] - \Pr\left[X_{(i,j)+1}\right]\right| \leq \mathsf{negl}(\lambda).$$

To this end, we use D to construct a simulator $\mathcal{B}$ against the indistinguishability property of wPRF. The behavior of $\mathcal{B}$ is identical to that of the challenger $\mathcal{C}$ in $\mathsf{G}_{(i,j)}$, except that: (1) the keys $(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$ are generated by $\mathcal{B}$'s challenger for indistinguishability, denoted by $\mathcal{C}_{\mathsf{IND}}$; and (2) the ciphertext in the attempt indexed by (i,j) is formed using the challenge returned by the one-time challenge oracle, denoted by $\mathcal{O}_{\mathsf{Chall}}$. The detailed description of $\mathcal{B}$ is given in Fig. 13, where the changes relative to $\mathsf{G}_{(i,j)}$ are highlighted.

Notice that if ct^* is generated by wEnc, then $\mathcal{B}$ simulates $\mathsf{G}_{(i,j)}$, whereas if ct^* is generated by Enc, then $\mathcal{B}$ simulates $\mathsf{G}_{(i,j)+1}$. Thus, by the indistinguishability of wPRF, it follows that

$$\left|\Pr\left[X_{(i,j)}\right] - \Pr\left[X_{(i,j)+1}\right]\right| = \left|\Pr\left[\mathsf{ExpIND}_{\mathcal{B},0}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ExpIND}_{\mathcal{B},1}(1^\lambda) = 1\right]\right|$$

Simulator $\mathcal{B}$:

1 $(\mathsf{pk}, \mathsf{sk}, \mathsf{ek}) \leftarrow \mathcal{C}_{\mathsf{IND}}(1^\lambda)$
2
3 $i' \leftarrow 1$
4 **return** $\mathsf{D}^{\mathcal{O}(\cdot,\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
5 **where** $\mathcal{O}(\cdot,\cdot)$ on input m, am:
6 **if** $i' > Q$
7 **return** $\perp$
8 $j' \leftarrow 1$
9 **if** $j' > v$
10 $i' \leftarrow i' + 1$
11 **return** am
12 **if** $(i', j') < (i, j)$
$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
else if $(i', j') > (i, j)$
$(\mathsf{s}, \mathsf{ct}) \leftarrow \mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, \mathsf{m})$
else if $(i', j') = (i, j)$
$\mathsf{ct} \leftarrow \mathsf{ct}^* \leftarrow \mathcal{C}_{\mathsf{IND}}.\mathcal{O}_{\mathsf{Chall}}(\mathsf{m})$
13 **if** $\mathsf{am} = \mathsf{r} \hookleftarrow \mathsf{UF}_p$
14 $i' \leftarrow i' + 1$
15 **return** ct
16 **else**
17 $j' \leftarrow j' + 1$
18 **go to** Step 9

Fig. 13: The Simulator for Proving $\mathsf{G}_{(i,j)} \approx \mathsf{G}_{(i,j)+1}$

$$= \mathsf{Adv}^{\mathsf{IND}}_{\mathcal{B},\mathsf{wPRF}}(\lambda) \leq \mathsf{negl}(\lambda).$$

Finally, by the triangle inequality, we obtain

$$\begin{aligned} |\Pr[W_1] - \Pr[W_2]| &= \left|\Pr\left[X_{(1,1)}\right] - \Pr\left[X_{(Q+1,1)}\right]\right| \\ &\leq Qv \cdot \mathsf{negl}(\lambda) \leq \mathsf{negl}(\lambda), \end{aligned}$$

thereby proving the claim. □

Claim 3 $|\Pr[W_2] - \Pr[W_3]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. Consider the event V in Game 2 and Game 3 that there exists an oracle query for which Steps 10 and 11 are executed. If V does not occur, then Game 2 and Game 3 are identical. Hence,

$$\Pr[W_2 \wedge \neg V] = \Pr[W_3 \wedge \neg V].$$

By the difference lemma, it follows that

$$|\Pr[W_2] - \Pr[W_3]| \leq \Pr[V]\,,$$

and moreover,

$$\begin{aligned}\Pr[V] &\leq Q \cdot (1 - p/2^{\ell})^{v} = Q \cdot (1 - p/2^{\ell})^{\lambda \cdot 2^{\ell}/p} \\ &\leq Q \cdot 2^{-\lambda} \leq \mathsf{negl}(\lambda).\end{aligned}$$

This establishes the claim. □

Claim 4 $|\Pr[W_3] - \Pr[W_4]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. The claim follows directly from Lemma 1. □

Combining the above claims and applying the triangle inequality, we obtain that, for every PPT dictator D,

$$\mathsf{Adv}^{\mathsf{IND\text{-}Collu}}_{\mathsf{D},\Sigma}(\lambda) = |\Pr[W_0] - \Pr[W_4]| \leq \mathsf{negl}(\lambda),$$

thereby proving indistinguishability against collusion of the asymmetric anamorphic extension Σ for the PKE scheme Π.

C Proof of Theorem 3

This proof is divided into three parts: correctness, indistinguishability, and weak pseudorandomness.

Correctness. This property comes directly from the correctness of Σ.

Indistinguishability. This property is due to the indistinguishability against collusion of Σ. Assume a PPT adversary $\mathcal{A}$ against the indistinguishability of wPRF, we construct a PPT simulator $\mathcal{B}$ to break the indistinguishability against collusion of Σ, which simulates either $\mathsf{ExpIND}_{\mathcal{A},0}$ or $\mathsf{ExpIND}_{\mathcal{A},1}$, as defined in Fig. 8. Let $\mathcal{C}$ denote the challenger of $\mathcal{B}$, then $\mathcal{B}$ works as follows. $\mathcal{B}$ obtains the keys $(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$ from $\mathcal{C}$, and directly forwards them to $\mathcal{A}$. Next, $\mathcal{A}$ sends a message m to $\mathcal{B}$, $\mathcal{B}$ then picks $\mathsf{am} \xleftarrow{\$} \{0,1\}^{\ell}$, and makes an encryption query on $(\mathsf{m}, \mathsf{am})$ to $\mathcal{C}$. When $\mathcal{B}$ gets a ciphertext ct from $\mathcal{C}$, it directly sends ct to $\mathcal{A}$ and outputs whatever $\mathcal{A}$ outputs. If ct is generated as $\mathsf{Enc}(\mathsf{pk}, \mathsf{m})$, then $\mathcal{B}$ simulates $\mathsf{ExpIND}_{\mathcal{A},0}$. If ct is answered as $\mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am})$, then $\mathcal{B}$ simulates $\mathsf{ExpIND}_{\mathcal{A},1}$ according to the definition of wEnc.

Weak Pseudorandomness. To illustrate this property, we define the following games between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. For clarity, the game hopping process is shown algorithmically in Fig. 14.

- **Game 0:** This is the experiment $\mathsf{ExpWPR}_{\mathcal{A},0}$ defined as Fig. 9.
- **Game 1:** Same as Game 0, except that in Step 6, ct is output by the original encryption algorithm Enc.

Game 0

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(\mathsf{pk})$
3 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$
4 **where** $\mathcal{O}(\cdot)$ on input m:
5 $\mathsf{am} \xleftarrow{\$} \{0,1\}^\ell$
6 $\mathsf{ct} \leftarrow \mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am})$
7 $\mathsf{y} \leftarrow \mathsf{am}$
8 **return** $(\mathsf{ct}, \mathsf{y})$

Game 1

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(\mathsf{pk})$
3 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$
4 **where** $\mathcal{O}(\cdot)$ on input m:
5 $\mathsf{am} \xleftarrow{\$} \{0,1\}^\ell$
6 $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
7 $\mathsf{y} \leftarrow \mathsf{am}$
8 **return** $(\mathsf{ct}, \mathsf{y})$

Game 2

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(\mathsf{pk})$
3 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$
4 **where** $\mathcal{O}(\cdot)$ on input m:
5 $\mathsf{am} \xleftarrow{\$} \{0,1\}^\ell$
6 $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
7 $\mathsf{y} \xleftarrow{\$} \{0,1\}^\ell$
8 **return** $(\mathsf{ct}, \mathsf{y})$

Game 3

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{sdk}, \mathsf{rdk}) \leftarrow \mathsf{aKGen}(\mathsf{pk})$
3 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$
4 **where** $\mathcal{O}(\cdot)$ on input m:
5 $\mathsf{am} \xleftarrow{\$} \{0,1\}^\ell$
6 $\mathsf{ct} \leftarrow \mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am})$
7 $\mathsf{y} \xleftarrow{\$} \{0,1\}^\ell$
8 **return** $(\mathsf{ct}, \mathsf{y})$

Fig. 14: Weak Pseudorandomness of wPRF

- **Game 2:** Same as Game 1, except that in Step 7, y is now sampled uniformly from $\{0,1\}^\ell$ (specifically, independent from am).
- **Game 3:** Same as Game 2, except that in Step 6, ct is generated via the anamorphic encryption algorithm aEnc.

Let W_ξ denote the event that $\mathcal{A}$ outputs 1 in Game ξ for $\xi \in [0,3]$, then we have the following claims.

Claim 5 $|\Pr[W_0] - \Pr[W_1]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim stems from the indistinguishability against collusion of Σ. Specifically, we construct from $\mathcal{A}$ a simulator $\mathcal{B}$ against this property of Σ. $\mathcal{B}$ simulates either Game 0 or Game 1. Let $\mathcal{C}_{\mathsf{IND\text{-}Collu}}$ be the challenger of $\mathcal{B}$, who provides $\mathcal{B}$ with an oracle $\mathcal{O}_{\mathsf{E}}$ that on input $(\mathsf{m}, \mathsf{am})$ outputs either $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$ or $\mathsf{ct} \leftarrow \mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am})$. The simulator $\mathcal{B}$ proceeds the same as the challenger $\mathcal{C}$ in Game 0, except that: (1) the keys $(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$ are generated by $\mathcal{C}_{\mathsf{IND\text{-}Collu}}$; and (2) in Step 6, the ciphertexts ct are generated using the responses returned by $\mathcal{O}_{\mathsf{E}}$. An algorithmic description of $\mathcal{B}$ is provided in Fig. 15, with the differences relative to Game 0 highlighted.

Notice that if ct is computed as $\mathsf{ct} \leftarrow \mathsf{aEnc}(\mathsf{pk}, \mathsf{sdk}, \mathsf{m}, \mathsf{am})$, then $\mathcal{B}$ simulates Game 0, whereas if ct is generated as $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$, then $\mathcal{B}$ simulates Game 1. Thus, by the indistinguishability against collusion of Σ, we have

$$\begin{aligned} |\Pr[W_0] - \Pr[W_1]| &= \left|\Pr\left[\mathsf{ExpAnam}_{\mathcal{B}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ExpReal}_{\mathcal{B}}(1^\lambda) = 1\right]\right| \\ &= \mathsf{Adv}^{\mathsf{IND\text{-}Collu}}_{\mathcal{B},\Sigma}(\lambda) \leq \mathsf{negl}(\lambda), \end{aligned}$$

which concludes the claim. □

Simulator $\mathcal{B}$:

1 $(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk}) \leftarrow \mathcal{C}_{\mathsf{IND\text{-}Collu}}(1^\lambda)$
2
3 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{sdk})$
4 **where** $\mathcal{O}(\cdot)$ on input m:
5 $\mathsf{am} \xleftarrow{\$} \{0,1\}^\ell$
6 $\mathsf{ct} \leftarrow \mathcal{C}_{\mathsf{IND\text{-}Collu}}.\mathcal{O}_{\mathsf{E}}(\mathsf{m}, \mathsf{am})$
7 $\mathsf{y} \leftarrow \mathsf{am}$
8 **return** $(\mathsf{ct}, \mathsf{y})$

Fig. 15: The Simulator for Proving Game 0 ≈ Game 1

Claim 6 $|\Pr[W_1] - \Pr[W_2]| = 0$.

Proof of Claim. The change is conceptual. □

Claim 7 $|\Pr[W_2] - \Pr[W_3]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim is due to the indistinguishability against collusion of Σ and the proof is similar to that of Claim 5. Thus, we omit it here. □

D Proof of Theorem 4

The proof is divided into three parts: we first establish correctness, then prove indistinguishability, and finally show weak pseudorandomness.

Correctness. From the definition of wEnc, we have $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$, i.e., $R_{\mathsf{pk}}(\mathsf{s}, (\mathsf{m}, \mathsf{ct})) = 1$. Therefore, by the correctness of WPRF, $\mathsf{wEval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct})) = \mathsf{Eval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct})) = \mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct})) = \mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$.

Indistinguishability. This property follows from the pseudorandomness of G. Given a PPT adversary $\mathcal{A}$ against the indistinguishability of $\mathsf{wPRF}_{\mathsf{pke}}$, we construct a PPT simulator $\mathcal{B}$ against the pseudorandomness of G that simulates either $\mathsf{ExpIND}_{\mathcal{A},0}$ or $\mathsf{ExpIND}_{\mathcal{A},1}$, as defined in Fig. 8. The simulator $\mathcal{B}$

first runs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, constructs the circuit R_{pk}, and runs $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$. It then sends $(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$ to $\mathcal{A}$. After receiving a message m from $\mathcal{A}$, $\mathcal{B}$ obtains a challenge r from its pseudorandomness challenger, where r is either sampled uniformly from $\{0,1\}^\tau$ or set to $\mathsf{G}(\mathsf{s})$ for $\mathsf{s} \xleftarrow{\$} \{0,1\}^\sigma$. The simulator then computes $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{r})$ and sends ct to $\mathcal{A}$. Finally, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs. Notice that if r is sampled uniformly from $\{0,1\}^\tau$, then ct is distributed exactly as output of Enc, and hence $\mathcal{B}$ simulates $\mathsf{ExpIND}_{\mathcal{A},0}$; whereas if $\mathsf{r} = \mathsf{G}(\mathsf{s})$ for $\mathsf{s} \xleftarrow{\$} \{0,1\}^\sigma$, then ct is distributed exactly as output of wEnc, and hence $\mathcal{B}$ simulates $\mathsf{ExpIND}_{\mathcal{A},1}$.

Weak Pseudorandomness. To prove this property, we define the following games between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Assume that $\mathcal{A}$ is allowed to make at most $Q = \mathsf{poly}(\lambda)$ oracle queries. For clarity, we describe the game-hopping process algorithmically in Fig. 16.

- **Game 0:** This is the experiment $\mathsf{ExpWPR}_{\mathcal{A},0}$ defined in Fig. 9.
- **Game 1:** Same as Game 0, except that in Step 10, y is now computed by F instead of Eval.
- **Game 2:** Same as Game 1, except that in Steps 8 and 9, the ciphertext ct is computed using the original encryption algorithm, without using the PRG output as the encryption randomness.
- **Game 3:** Same as Game 2, except that in Step 10, y is sampled uniformly from $\{0,1\}^\ell$.
- **Game 4:** Same as Game 3, except that in Steps 8 and 9, the way that ct is generated is changed back to that in Game 0 and Game 1, namely the encryption randomness is set to the output of the PRG. Note that Game 4 is exactly the experiment $\mathsf{ExpWPR}_{\mathcal{A},1}$ defined in Fig. 9.

Let W_ξ denote the event that $\mathcal{A}$ outputs 1 in Game ξ for $\xi \in [0,4]$. We then have the following claims.

Claim 8 $|\Pr[W_0] - \Pr[W_1]| = 0$.

Proof of Claim. This claim follows directly from the correctness of WPRF. □

Claim 9 $|\Pr[W_1] - \Pr[W_2]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim follows from the pseudorandomness of G. To prove it, we define a sequence of games G_i for $i \in [1, Q+1]$ that interpolate between Game 1 and Game 2. For clarity, we present an algorithmic description of G_i in Fig. 17, with the modifications relative to Game 1 highlighted.

- G_i: Same as Game 1, except that for any oracle query before the i-th query (that is, whose index $i' < i$), ct is generated using Enc with fresh uniform randomness. For every query on or after the i-th one (namely, $i' \geq i$), the encryption randomness is still generated by G.

Game 0

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $\mathsf{s} \xleftarrow{\$} \{0,1\}^\sigma$
    $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$
    $\mathsf{y} \leftarrow \mathsf{Eval}(\mathsf{ek}, \mathsf{s}, (\mathsf{m}, \mathsf{ct}))$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Game 1

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $\mathsf{s} \xleftarrow{\$} \{0,1\}^\sigma$
    $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$
    $\mathsf{y} \leftarrow \mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Game 2

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$

    $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
    $\mathsf{y} \leftarrow \mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Game 3

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$

    $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
    $\mathsf{y} \xleftarrow{\$} \{0,1\}^\ell$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Game 4

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $\mathsf{s} \xleftarrow{\$} \{0,1\}^\sigma$
    $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$
    $\mathsf{y} \xleftarrow{\$} \{0,1\}^\ell$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Fig. 16: Weak Pseudorandomness of $\mathsf{wPRF}_{\mathsf{pke}}$

Note that G_1 is the same as Game 1, and G_{Q+1} is the same as Game 2. Thus, it suffices to show that $\mathsf{G}_i \approx \mathsf{G}_{i+1}$ for every $i \in [1, Q]$. Let X_i denote the event that $\mathcal{A}$ outputs 1 in G_i.

To this end, we construct from $\mathcal{A}$ a simulator $\mathcal{B}$ against the pseudorandomness of G that simulates either G_i or G_{i+1}. Its challenger provides a string r^* that is either $\mathsf{G}(\mathsf{s}^*)$ for $\mathsf{s}^* \xleftarrow{\$} \{0,1\}^\sigma$, or sampled uniformly from $\{0,1\}^\tau$. The simulator behaves identically to the challenger in G_i, except that the ciphertext in the i-th oracle query is computed as $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{r}^*)$. A formal description of $\mathcal{B}$ is given in Fig. 18, where the differences relative to G_i are highlighted.

G_i :

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$
3 $i' \leftarrow 1$
4 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
5 **where** $\mathcal{O}(\cdot)$ on input m:
6 **if** $i' > Q$
7 **return** $\perp$
8 $\mathsf{s} \xleftarrow{\$} \{0,1\}^\sigma$
9 **if** $i' < i$
$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
else if $i' \geq i$
$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$
10 $\mathsf{y} \leftarrow \mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$
11 $i' \leftarrow i' + 1$
12 **return** $(\mathsf{ct}, \mathsf{y})$

Fig. 17: Game Hopping Between Game 1 and Game 2

If $\mathsf{r}^* = \mathsf{G}(\mathsf{s}^*)$, then $\mathcal{B}$ simulates G_i; if r^* is uniform over $\{0,1\}^\tau$, then it simulates G_{i+1}. Therefore, by the pseudorandomness of G, we obtain

$$|\Pr[X_i] - \Pr[X_{i+1}]| \leq \mathsf{negl}(\lambda).$$

The claim then follows. □

Claim 10 $|\Pr[W_2] - \Pr[W_3]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim follows from the pseudorandomness of WPRF. To prove it, we define a sequence of games H_i for $i \in [1, Q+1]$ that interpolate between Game 2 and Game 3. For clarity, we present an algorithmic description of H_i in Fig. 19, with the modifications relative to Game 2 highlighted.

- H_i: Same as Game 2, except that we modify the way y is generated. Specifically, for any oracle query before the i-th query (that is, whose index $i' < i$), y is sampled uniformly from $\{0,1\}^\ell$, while for any oracle query on or after the i-th query (namely, $i' \geq i$), y is computed by F.

Note that H_1 is the same as Game 2, and H_{Q+1} is the same as Game 3. Thus, it suffices to show that $\mathsf{H}_i \approx \mathsf{H}_{i+1}$ for every $i \in [1, Q]$. In other words, let X_i be the event that $\mathcal{A}$ outputs 1 in H_i, it is enough to prove that for every PPT adversary $\mathcal{A}$, there is $|\Pr[X_i] - \Pr[X_{i+1}]| \leq \mathsf{negl}(\lambda)$.

To this end, we construct from $\mathcal{A}$ a simulator $\mathcal{B}$ against the pseudorandomness of WPRF that simulates either H_i or H_{i+1}. Let $\mathcal{C}_{\mathsf{PR}}$ denote the challenger of $\mathcal{B}$.

Simulator $\mathcal{B}(\mathsf{r}^*)$:

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$
3 $i' \leftarrow 1$
4 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
5 **where** $\mathcal{O}(\cdot)$ on input m:
6 **if** $i' > Q$
7 **return** $\perp$
8 $\mathsf{s} \xleftarrow{\$} \{0,1\}^\sigma$
9 **if** $i' < i$
$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
else if $i' = i$
$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{r}^*)$
else if $i' > i$
$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))$
10 $\mathsf{y} \leftarrow \mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$
11 $i' \leftarrow i' + 1$
12 **return** $(\mathsf{ct}, \mathsf{y})$

Fig. 18: The Simulator for Proving $\mathsf{G}_i \approx \mathsf{G}_{i+1}$

The challenger $\mathcal{C}_{\mathsf{PR}}$ provides $\mathcal{B}$ with an oracle $\mathcal{O}_{\mathsf{F}}$ that, on input $(\mathsf{m}, \mathsf{ct})$, outputs $\mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$, together with a one-time challenge oracle $\mathcal{O}_{\mathsf{Chall}}$ that, on input $(\mathsf{m}, \mathsf{ct})$, outputs either $\mathsf{y} \xleftarrow{\$} \{0,1\}^\ell$ or $\mathsf{y} \leftarrow \mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$. The simulator $\mathcal{B}$ behaves identically to the challenger $\mathcal{C}$ in H_i, except that: (1) the key ek is generated by $\mathcal{C}_{\mathsf{PR}}$; (2) the value y in the i-th oracle query is generated using the challenge returned by $\mathcal{O}_{\mathsf{Chall}}$; and (3) the value y in every oracle query after the i-th one is generated using the response returned by $\mathcal{O}_{\mathsf{F}}$. A formal description of $\mathcal{B}$ is given in Fig. 20, where the differences relative to H_i are highlighted.

In H_i, H_{i+1}, and the simulation of $\mathcal{B}$, we make the following observations.

- Define $\mathcal{L}_{\mathsf{pk}} = \{(\mathsf{m}, \mathsf{ct}) : \exists\ \mathsf{s} \in \{0,1\}^\sigma \text{ s.t. } \mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}; \mathsf{G}(\mathsf{s}))\}$, that is, the language related to R_{pk}. Let $(\mathsf{m}^*, \mathsf{ct}^*)$ be the statement, a pair of message and ciphertext, shown in Step 9 of the i-th oracle query, then by the choice of $\sigma = \kappa/2$, we claim that $\Pr[(\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{L}_{\mathsf{pk}}] \leq \mathsf{negl}(\lambda)$.
  Fix the message m^*, there are at most 2^σ ciphertexts ct such that $(\mathsf{m}^*, \mathsf{ct}) \in \mathcal{L}_{\mathsf{pk}}$. Then from the high min-entropy property of Enc, we have that the probability of ct^* being such a ciphertext, namely $\Pr[(\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{L}_{\mathsf{pk}}]$, is at most $2^\sigma \cdot 2^{-\mathsf{H}_\infty(\mathsf{Enc})} = 2^{\kappa/2} \cdot 2^{-\kappa} = 2^{-\kappa/2} \leq \mathsf{negl}(\lambda)$.
- Let $\mathcal{T}$ represent the set of statements $(\mathsf{m}, \mathsf{ct})$ shown in Step 9 of all subsequent queries after the i-th one, then we have that $\Pr[(\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{T}] \leq Q \cdot 2^{-\mathsf{H}_\infty(\mathsf{Enc})} = Q \cdot 2^{-\kappa} \leq \mathsf{negl}(\lambda)$.

H_i :

1 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 $(\mathsf{ek}, \mathsf{fk}) \leftarrow \mathsf{Gen}(1^\lambda, R_{\mathsf{pk}})$
3 $i' \leftarrow 1$
4 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
5 **where** $\mathcal{O}(\cdot)$ on input m:
6 **if** $i' > Q$
7 **return** $\perp$
8
9 $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
10 **if** $i' < i$
$\mathsf{y} \xleftarrow{\$} \{0,1\}^\ell$
else if $i' \geq i$
$\mathsf{y} \leftarrow \mathsf{F}(\mathsf{fk}, (\mathsf{m}, \mathsf{ct}))$
11 $i' \leftarrow i' + 1$
12 **return** $(\mathsf{ct}, \mathsf{y})$

Fig. 19: Game Hopping Between Game 2 and Game 3

We now turn to the analysis of $\mathcal{B}$'s simulation.

– If $\mathsf{y}^* = \mathsf{F}(\mathsf{fk}, (\mathsf{m}^*, \mathsf{ct}^*))$, then $\mathcal{B}$ simulates H_i. From the experiment output of ExpPR defined in Fig. 6, we have that

$$\begin{aligned}\Pr\left[\mathsf{ExpPR}^{R_{\mathsf{pk}}}_{\mathcal{B},0}(1^\lambda) = 1\right] &= \Pr\left[X_i \wedge (\mathsf{m}^*, \mathsf{ct}^*) \notin \mathcal{L}_{\mathsf{pk}} \wedge (\mathsf{m}^*, \mathsf{ct}^*) \notin \mathcal{T}\right] \\ &= \Pr\left[X_i \wedge \neg\left((\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{L}_{\mathsf{pk}} \vee (\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{T}\right)\right].\end{aligned}$$

Since $\Pr[A] - \Pr[B] = \Pr[A \wedge \neg B] + \Pr[A \wedge B] - \Pr[B] \leq \Pr[A \wedge \neg B]$, we have that (by setting $A = X_i$ and $B = (\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{L}_{\mathsf{pk}} \vee (\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{T}$):

$$\Pr[X_i] - \mathsf{negl}(\lambda) \leq \Pr\left[\mathsf{ExpPR}^{R_{\mathsf{pk}}}_{\mathcal{B},0}(1^\lambda) = 1\right] \leq \Pr[X_i]. \tag{1}$$

– If y^* is sampled uniformly from $\{0,1\}^\ell$, then the simulation by $\mathcal{B}$ is identical to H_{i+1}. From the experiment output of ExpPR defined in Fig. 6, there is

$$\begin{aligned}\Pr\left[\mathsf{ExpPR}^{R_{\mathsf{pk}}}_{\mathcal{B},1}(1^\lambda) = 1\right] &= \Pr\left[X_{i+1} \wedge (\mathsf{m}^*, \mathsf{ct}^*) \notin \mathcal{L}_{\mathsf{pk}} \wedge (\mathsf{m}^*, \mathsf{ct}^*) \notin \mathcal{T}\right] \\ &= \Pr\left[X_{i+1} \wedge \neg\left((\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{L}_{\mathsf{pk}} \vee (\mathsf{m}^*, \mathsf{ct}^*) \in \mathcal{T}\right)\right].\end{aligned}$$

From a similar argument, we obtain

$$\Pr[X_{i+1}] - \mathsf{negl}(\lambda) \leq \Pr\left[\mathsf{ExpPR}^{R_{\mathsf{pk}}}_{\mathcal{B},1}(1^\lambda) = 1\right] \leq \Pr[X_{i+1}]. \tag{2}$$

Simulator $\mathcal{B}$:

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$\mathsf{ek} \leftarrow \mathcal{C}_{\mathsf{PR}}(1^\lambda, R_{\mathsf{pk}})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$

    $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$
    if $i' < i$
      $\mathsf{y} \xleftarrow{\$} \{0,1\}^\ell$
    else if $i' = i$
      $\mathsf{y} \leftarrow \mathsf{y}^* \leftarrow \mathcal{C}_{\mathsf{PR}}.\mathcal{O}_{\mathsf{Chall}}(\mathsf{m}, \mathsf{ct})$
    else if $i' > i$
      $\mathsf{y} \leftarrow \mathcal{C}_{\mathsf{PR}}.\mathcal{O}_{\mathsf{F}}(\mathsf{m}, \mathsf{ct})$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Fig. 20: The Simulator for Proving $\mathsf{H}_i \approx \mathsf{H}_{i+1}$

Combining Formulas (1), (2), triangle inequality, and the pseudorandomness of WPRF, it follows that

$$\begin{aligned}
|\Pr[X_i] - \Pr[X_{i+1}]| &\leq \left|\Pr[X_i] - \Pr\left[\mathsf{ExpPR}^{R_{\mathsf{pk}}}_{\mathcal{B},0}(1^\lambda) = 1\right]\right| \\
&\quad + \left|\Pr\left[\mathsf{ExpPR}^{R_{\mathsf{pk}}}_{\mathcal{B},0}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ExpPR}^{R_{\mathsf{pk}}}_{\mathcal{B},1}(1^\lambda) = 1\right]\right| \\
&\quad + \left|\Pr\left[\mathsf{ExpPR}^{R_{\mathsf{pk}}}_{\mathcal{B},1}(1^\lambda) = 1\right] - \Pr[X_{i+1}]\right| \\
&\leq \mathsf{Adv}^{\mathsf{PR}}_{\mathcal{B},\mathsf{WPRF}}(\lambda) + \mathsf{negl}(\lambda) \leq \mathsf{negl}(\lambda),
\end{aligned}$$

which shows that $\mathsf{H}_i \approx \mathsf{H}_{i+1}$ and thus our claim follows. □

Claim 11 $|\Pr[W_3] - \Pr[W_4]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. Again, this claim follows from the pseudorandomness of G, and its proof is similar to that of Claim 9. □

E Proof of Theorem 5

Since wEnc generates ciphertexts in exactly the same way as Enc, indistinguishability is immediate. It therefore remains only to demonstrate correctness and weak pseudorandomness.

Correctness. Consider a ciphertext $\mathsf{ct} = (c_0, c_1) = (g^s, h^s \cdot m) = \mathsf{Enc}(\mathsf{pk}, m; s)$, we have that $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (m, \mathsf{ct})) = \mathbf{A} \cdot \mathsf{bin}(c_0^{\widehat{x}}) = \mathbf{A} \cdot \mathsf{bin}(g^{s \cdot \widehat{x}}) = \mathbf{A} \cdot \mathsf{bin}(\widehat{h}^s) = \mathsf{wEval}(\mathsf{ek}, s, (m, \mathsf{ct}))$.

Game 0

$x \xleftarrow{\$} \mathbb{Z}_q$, $h = g^x$
$\widehat{x} \xleftarrow{\$} \mathbb{Z}_q$, $\widehat{h} = g^{\widehat{x}}$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = h$, $\mathsf{sk} = x$, $\mathsf{ek} = \left(\widehat{h}, \mathbf{A}\right)$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $s \xleftarrow{\$} \mathbb{Z}_q$
    $\mathsf{ct} = (g^s, h^s \cdot m)$
    $r = \widehat{h}^s$
    $\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(r) \bmod 2$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Game 1

$x \xleftarrow{\$} \mathbb{Z}_q$, $h = g^x$
$\widehat{x} \xleftarrow{\$} \mathbb{Z}_q$, $\widehat{h} = g^{\widehat{x}}$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = h$, $\mathsf{sk} = x$, $\mathsf{ek} = \left(\widehat{h}, \mathbf{A}\right)$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $s \xleftarrow{\$} \mathbb{Z}_q$
    $\mathsf{ct} = (g^s, h^s \cdot m)$
    $r \xleftarrow{\$} \mathbb{G}$
    $\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(r) \bmod 2$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Game 2

$x \xleftarrow{\$} \mathbb{Z}_q$, $h = g^x$
$\widehat{x} \xleftarrow{\$} \mathbb{Z}_q$, $\widehat{h} = g^{\widehat{x}}$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = h$, $\mathsf{sk} = x$, $\mathsf{ek} = \left(\widehat{h}, \mathbf{A}\right)$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $s \xleftarrow{\$} \mathbb{Z}_q$
    $\mathsf{ct} = (g^s, h^s \cdot m)$
    $\mathsf{y} \xleftarrow{\$} \{0,1\}^{\ell}$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Fig. 21: Weak Pseudorandomness of $\mathsf{wPRF}_{\mathsf{elg}}$

Weak Pseudorandomness. We define the following games run between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Suppose that $\mathcal{A}$ is allowed to make at most $Q = \mathsf{poly}(\lambda)$ oracle queries. An overview of the game-hopping sequence is given in Fig. 21, where the changes in each game relative to the previous one are highlighted.

- **Game 0.** This is the experiment $\mathsf{ExpWPR}_{\mathcal{A},0}$ defined in Fig. 9.
- **Game 1.** Same as Game 0, except that in Step 12, r is sampled uniformly from $\mathbb{G}$.
- **Game 2.** Same as Game 1, except that in Step 13, y is sampled uniformly from $\{0,1\}^{\ell}$. This is the experiment $\mathsf{ExpWPR}_{\mathcal{A},1}$ defined in Fig. 9.

Let W_ξ denote the event that $\mathcal{A}$ outputs 1 in Game ξ for $\xi \in [0, 2]$. We then have the following claims.

Claim 12 $|\Pr[W_0] - \Pr[W_1]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim follows from the DDH assumption. Formally, we define a sequence of games H_i for $i \in [1, Q+1]$ that interpolate between Game 0 and Game 1. A concrete description of H_i is given in Fig. 22.

H_i :

1 $x \xleftarrow{\$} \mathbb{Z}_q$, $h = g^x$
2 $\widehat{x} \xleftarrow{\$} \mathbb{Z}_q$, $\widehat{h} = g^{\widehat{x}}$
3 $\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
4 $\mathsf{pk} = h$, $\mathsf{sk} = x$, $\mathsf{ek} = \left(\widehat{h}, \mathbf{A}\right)$
5 $i' \leftarrow 1$
6 **return** $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
7 **where** $\mathcal{O}(\cdot)$ on input m:
8 **if** $i' > Q$
9 **return** $\perp$
10 $s \xleftarrow{\$} \mathbb{Z}_q$
11 $\mathsf{ct} = (g^s, h^s \cdot m)$
12 **if** $i' < i$
$r \xleftarrow{\$} \mathbb{G}$
else if $i' \geq i$
$r = \widehat{h}^s$
13 $\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(r) \bmod 2$
14 $i' \leftarrow i' + 1$
15 **return** $(\mathsf{ct}, \mathsf{y})$

Fig. 22: Game Hopping Between Game 0 and Game 1

- H_i: Same as Game 0, except that we modify the way in which $\mathcal{C}$ generates r in Step 12. Specifically, for the i'-th oracle query where $i' < i$, r is picked uniformly from $\mathbb{G}$, while for the i'-th oracle query where $i' \geq i$, r is computed as Game 0 by $r \leftarrow \widehat{h}^s$.

Let X_i denote the event that $\mathcal{A}$ outputs 1 in H_i. Observe that H_1 and H_{Q+1} are identical to Game 0 and Game 1, respectively. Thus, it suffices to prove that for every $i \in [1, Q]$, $|\Pr[X_i] - \Pr[X_{i+1}]| \leq \mathsf{negl}(\lambda)$.

To this end, we construct from $\mathcal{A}$ a simulator $\mathcal{B}$ against the DDH assumption that simulates either H_i or H_{i+1}. The behavior of $\mathcal{B}$ is identical to that of the challenger $\mathcal{C}$ in H_i, except that the following values are derived from the given DDH instance (g, g^a, g^b, g^c), where c is either equal to ab or sampled uniformly from $\mathbb{Z}_q$: (1) the element $\widehat{h}$; and (2) the ciphertext ct and the element r in the i-th oracle query. A concrete description of $\mathcal{B}$ is given in Fig. 23.

Note that if $c = ab$, then $\mathcal{B}$ simulates H_i; whereas if $c \xleftarrow{\$} \mathbb{Z}_q$, then $r \leftarrow g^c \xleftarrow{\$} \mathbb{G}$, and hence $\mathcal{B}$ simulates H_{i+1}. Therefore, by the DDH assumption, we obtain $|\Pr[X_i] - \Pr[X_{i+1}]| \leq \mathsf{negl}(\lambda)$, from which the claim follows. □

Claim 13 $|\Pr[W_1] - \Pr[W_2]| \leq \mathsf{negl}(\lambda)$.

Simulator $\mathcal{B}$ on Input (g, g^a, g^b, g^c):

$x \xleftarrow{\$} \mathbb{Z}_q$, $h = g^x$
$\widehat{h} \leftarrow g^a$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = h$, $\mathsf{sk} = x$, $\mathsf{ek} = \left(\widehat{h}, \mathbf{A}\right)$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $s \xleftarrow{\$} \mathbb{Z}_q$
    if $i' \neq i$
      $\mathsf{ct} = (g^s, h^s \cdot m)$
    else if $i' = i$
      $\mathsf{ct} \leftarrow (g^b, (g^b)^x \cdot m)$
    if $i' < i$
      $r \xleftarrow{\$} \mathbb{G}$
    else if $i' > i$
      $r = \widehat{h}^s$
    else if $i' = i$
      $r \leftarrow g^c$
    $\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(r) \bmod 2$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Fig. 23: The Simulator for Proving $\mathsf{H}_i \approx \mathsf{H}_{i+1}$

Proof of Claim. The claim follows directly from Lemma 3. □

F Proof of Theorem 6

Since wEnc generates ciphertexts exactly as Enc does, the indistinguishability follows immediately. Thus, it only remains to prove correctness and weak pseudorandomness.

Correctness. We show that wEval is identical to wF with overwhelming probability when wEval returns 0 or 1, over the randomness of KGen, wGen and wEnc. Consider a ciphertext $\mathsf{ct} = \begin{bmatrix} \mathbf{c} \\ c \end{bmatrix} = \begin{bmatrix} \mathbf{As} \\ c \end{bmatrix}$, a corresponding function key $\mathsf{fk} = \widehat{\mathbf{x}}$,

and evaluation key $\mathsf{ek} = \widehat{\mathbf{b}} = \mathbf{A}^t\widehat{\mathbf{x}}+\widehat{\mathbf{e}}$. The difference between r and r' (computed by wEval and wF, respectively) is $\Delta = \widehat{\mathbf{b}}^t\mathbf{s} - \widehat{\mathbf{x}}^t\mathbf{c} = (\widehat{\mathbf{x}}^t\mathbf{A} + \widehat{\mathbf{e}}^t)\mathbf{s} - \widehat{\mathbf{x}}^t(\mathbf{A}\mathbf{s}) = \widehat{\mathbf{e}}^t\mathbf{s}$.

For correctness, it is sufficient to ensure that with overwhelming probability, the quantity $|\Delta|$, where Δ is viewed as an element in $\mathbb{Z}$, satisfies $|\Delta| \leq q/10$. From Lemma 2 and our choice of $q \geq 10m$, $\alpha \leq \frac{1}{\sqrt{m}\omega(\sqrt{\log m})}$, we have that with overwhelming probability,

$$\begin{aligned}|\Delta| = |\widehat{\mathbf{e}}^t\mathbf{s}| &\leq ||\mathbf{s}||q\alpha \cdot \omega\left(\sqrt{\log m}\right) + ||\mathbf{s}||\sqrt{m}/2 \\ &\leq \sqrt{m}q\alpha \cdot \omega\left(\sqrt{\log m}\right) + m/2 \\ &\leq q \cdot \frac{\omega\left(\sqrt{\log m}\right)}{\omega\left(\sqrt{\log m}\right)} + \frac{q}{20} \leq \frac{q}{20} + \frac{q}{20} = \frac{q}{10}.\end{aligned}$$

Weak Pseudorandomness. We define the following games played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Assume that $\mathcal{A}$ is allowed to make at most $Q = \mathsf{poly}(\lambda)$ oracle queries. An overview of the game-hopping sequence is given in Fig. 24, where the changes in each game relative to the previous one are highlighted.

- **Game 0.** This is the experiment $\mathsf{ExpWPR}_{\mathcal{A},0}$ defined in Fig. 9.
- **Game 1.** Same as Game 0, except that in Step 3, the vector $\widehat{\mathbf{b}}$ is sampled uniformly from $\mathbb{Z}_q^m$.
- **Game 2.** Same as Game 1, except that in Steps 11 and 12, the vector

  $$\begin{bmatrix}\mathbf{A} \\ \widehat{\mathbf{b}}^t\end{bmatrix}\mathbf{s} \in \mathbb{Z}_q^{n+1}$$

  is replaced by a uniformly sampled vector

  $$\begin{bmatrix}\mathbf{z} \\ r\end{bmatrix} \in \mathbb{Z}_q^{n+1}.$$

  Accordingly, $\mathbf{b}^t\mathbf{s} = \mathbf{x}^t(\mathbf{A}\mathbf{s}) + \mathbf{e}^t\mathbf{s}$ is replaced by $\mathbf{x}^t\mathbf{z} + \mathbf{s}^t\mathbf{e}$.
- **Game 3.** Same as Game 2, except that in Step 11, $\mathbf{z}$ is changed back to $\mathbf{A}\mathbf{s}$. Accordingly, $\mathbf{x}^t\mathbf{z} + \mathbf{s}^t\mathbf{e}$ is changed back to $\mathbf{b}^t\mathbf{s}$.
- **Game 4.** Same as Game 3, except that in Step 13, y is sampled from UF_p. Note that here we set

  $$p = \frac{3\lfloor\frac{q}{8}\rfloor + 1 - \lfloor\frac{q}{8}\rfloor + 1}{q} \times 2 \geq \frac{2\left(\frac{q}{8} - 1\right) + 2}{q} \times 2 = \frac{1}{2},$$

  which is acceptable, since the probability that a random sample from this distribution is not the special symbol $\perp$ is a non-negligible value p which is at least $1/2$.
- **Game 5.** Same as Game 4, except that in Step 3, the generation of the vector $\widehat{\mathbf{b}}$ is changed back to $\widehat{\mathbf{b}} = \mathbf{A}^t\widehat{\mathbf{x}} + \widehat{\mathbf{e}}$. Note that Game 5 is exactly the experiment $\mathsf{ExpWPR}_{\mathcal{A},1}$ defined in Fig. 9.

Game 0

$\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$
$\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \hookleftarrow \chi^m$, $\mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e}$
$\widehat{\mathbf{x}} \xleftarrow{\$} \mathbb{Z}_q^n$, $\widehat{\mathbf{e}} \hookleftarrow \chi^m$, $\widehat{\mathbf{b}} = \mathbf{A}^t\widehat{\mathbf{x}} + \widehat{\mathbf{e}}$
$\mathsf{pk} \leftarrow \begin{bmatrix}\mathbf{A}\\ \mathbf{b}^t\end{bmatrix}$, $\mathsf{sk} \leftarrow \mathbf{x}$, $\mathsf{ek} \leftarrow \widehat{\mathbf{b}}$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input μ:
    if $i' > Q$
      return $\perp$
    $\mathbf{s} \xleftarrow{\$} \{0,1\}^m$
    $\mathsf{ct} = \begin{bmatrix}\mathbf{As}\\ \mathbf{b}^t\mathbf{s} + \mu \cdot \lfloor \frac{q}{2} \rceil\end{bmatrix}$
    $r = \widehat{\mathbf{b}}^t\mathbf{s}$
    if $\lfloor \frac{q}{8} \rfloor \le |r| \le 3\lfloor \frac{q}{8} \rfloor + 1$
      $y = \mathsf{sign}(r)$
    else
      $y = \perp$
    $i' \leftarrow i' + 1$
    return (ct, y)

Game 1

$\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$
$\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \hookleftarrow \chi^m$, $\mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e}$
$\widehat{\mathbf{b}} \xleftarrow{\$} \mathbb{Z}_q^m$
$\mathsf{pk} \leftarrow \begin{bmatrix}\mathbf{A}\\ \mathbf{b}^t\end{bmatrix}$, $\mathsf{sk} \leftarrow \mathbf{x}$, $\mathsf{ek} \leftarrow \widehat{\mathbf{b}}$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input μ:
    if $i' > Q$
      return $\perp$
    $\mathbf{s} \xleftarrow{\$} \{0,1\}^m$
    $\mathsf{ct} = \begin{bmatrix}\mathbf{As}\\ \mathbf{b}^t\mathbf{s} + \mu \cdot \lfloor \frac{q}{2} \rceil\end{bmatrix}$
    $r = \widehat{\mathbf{b}}^t\mathbf{s}$
    if $\lfloor \frac{q}{8} \rfloor \le |r| \le 3\lfloor \frac{q}{8} \rfloor + 1$
      $y = \mathsf{sign}(r)$
    else
      $y = \perp$
    $i' \leftarrow i' + 1$
    return (ct, y)

Game 2

$\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$
$\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \hookleftarrow \chi^m$, $\mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e}$
$\widehat{\mathbf{b}} \xleftarrow{\$} \mathbb{Z}_q^m$
$\mathsf{pk} \leftarrow \begin{bmatrix}\mathbf{A}\\ \mathbf{b}^t\end{bmatrix}$, $\mathsf{sk} \leftarrow \mathbf{x}$, $\mathsf{ek} \leftarrow \widehat{\mathbf{b}}$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input μ:
    if $i' > Q$
      return $\perp$
    $\mathbf{s} \xleftarrow{\$} \{0,1\}^m$
    $\mathsf{ct} = \begin{bmatrix}\mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^n\\ \mathbf{x}^t\mathbf{z} + \mathbf{s}^t\mathbf{e} + \mu \cdot \lfloor \frac{q}{2} \rceil\end{bmatrix}$
    $r \xleftarrow{\$} \mathbb{Z}_q$
    if $\lfloor \frac{q}{8} \rfloor \le |r| \le 3\lfloor \frac{q}{8} \rfloor + 1$
      $y = \mathsf{sign}(r)$
    else
      $y = \perp$
    $i' \leftarrow i' + 1$
    return (ct, y)

Game 3

$\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$
$\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \hookleftarrow \chi^m$, $\mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e}$
$\widehat{\mathbf{b}} \xleftarrow{\$} \mathbb{Z}_q^m$
$\mathsf{pk} \leftarrow \begin{bmatrix}\mathbf{A}\\ \mathbf{b}^t\end{bmatrix}$, $\mathsf{sk} \leftarrow \mathbf{x}$, $\mathsf{ek} \leftarrow \widehat{\mathbf{b}}$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input μ:
    if $i' > Q$
      return $\perp$
    $\mathbf{s} \xleftarrow{\$} \{0,1\}^m$
    $\mathsf{ct} = \begin{bmatrix}\mathbf{As}\\ \mathbf{b}^t\mathbf{s} + \mu \cdot \lfloor \frac{q}{2} \rceil\end{bmatrix}$
    $r \xleftarrow{\$} \mathbb{Z}_q$
    if $\lfloor \frac{q}{8} \rfloor \le |r| \le 3\lfloor \frac{q}{8} \rfloor + 1$
      $y = \mathsf{sign}(r)$
    else
      $y = \perp$
    $i' \leftarrow i' + 1$
    return (ct, y)

Game 4

$\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$
$\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \hookleftarrow \chi^m$, $\mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e}$
$\widehat{\mathbf{b}} \xleftarrow{\$} \mathbb{Z}_q^m$
$\mathsf{pk} \leftarrow \begin{bmatrix}\mathbf{A}\\ \mathbf{b}^t\end{bmatrix}$, $\mathsf{sk} \leftarrow \mathbf{x}$, $\mathsf{ek} \leftarrow \widehat{\mathbf{b}}$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input μ:
    if $i' > Q$
      return $\perp$
    $\mathbf{s} \xleftarrow{\$} \{0,1\}^m$
    $\mathsf{ct} = \begin{bmatrix}\mathbf{As}\\ \mathbf{b}^t\mathbf{s} + \mu \cdot \lfloor \frac{q}{2} \rceil\end{bmatrix}$
    $y \xleftarrow{\$} \mathsf{UF}_p$
    $i' \leftarrow i' + 1$
    return (ct, y)

Game 5

$\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$
$\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \hookleftarrow \chi^m$, $\mathbf{b} = \mathbf{A}^t\mathbf{x} + \mathbf{e}$
$\widehat{\mathbf{x}} \xleftarrow{\$} \mathbb{Z}_q^n$, $\widehat{\mathbf{e}} \hookleftarrow \chi^m$, $\widehat{\mathbf{b}} = \mathbf{A}^t\widehat{\mathbf{x}} + \widehat{\mathbf{e}}$
$\mathsf{pk} \leftarrow \begin{bmatrix}\mathbf{A}\\ \mathbf{b}^t\end{bmatrix}$, $\mathsf{sk} \leftarrow \mathbf{x}$, $\mathsf{ek} \leftarrow \widehat{\mathbf{b}}$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input μ:
    if $i' > Q$
      return $\perp$
    $\mathbf{s} \xleftarrow{\$} \{0,1\}^m$
    $\mathsf{ct} = \begin{bmatrix}\mathbf{As}\\ \mathbf{b}^t\mathbf{s} + \mu \cdot \lfloor \frac{q}{2} \rceil\end{bmatrix}$
    $y \xleftarrow{\$} \mathsf{UF}_p$
    $i' \leftarrow i' + 1$
    return (ct, y)

Fig. 24: Weak Pseudorandomness of $\mathsf{wPRF}_{\mathsf{reg}}$

Let W_ξ denote the event that $\mathcal{A}$ outputs 1 in Game ξ, for $\xi \in [0,5]$. We then have the following claims.

Claim 14 $|\Pr[W_0] - \Pr[W_1]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim follows from the decisional LWE assumption. □

Claim 15 $|\Pr[W_1] - \Pr[W_2]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim is implied by Lemma 4. □

Claim 16 $|\Pr[W_2] - \Pr[W_3]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim can be derived from Lemma 4. □

Claim 17 $|\Pr[W_3] - \Pr[W_4]| = 0$.

Proof of Claim. The change is conceptual. □

Claim 18 $|\Pr[W_4] - \Pr[W_5]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim is based on the decisional LWE assumption. □

G Proof of Theorem 7

We prove the theorem in three parts: correctness, indistinguishability, and weak pseudorandomness.

Correctness. Consider $((r,s),\mathsf{ct}) \leftarrow \mathsf{wEnc}(\mathsf{pk},\mathsf{ek},m)$, where

$$\mathsf{ct} = (c_0, c_1, c_2) = \left(u^r, \widehat{v}^s, h^r \cdot \widehat{h}^s \cdot m\right).$$

Since $u^x = v^y = h$, $\widehat{v} = v^{\widehat{a}}$, and $\widehat{H} = h^{\widehat{a}\widehat{b}}$, the group element computed by wF satisfies

$$\begin{aligned} R &= \left((c_0^x)^{\widehat{a}} \cdot c_1^y\right)^{\widehat{b}} = \left(((u^r)^x)^{\widehat{a}} \cdot ((v^{\widehat{a}})^s)^y\right)^{\widehat{b}} \\ &= \left((h^r)^{\widehat{a}} \cdot (h^{\widehat{a}})^s\right)^{\widehat{b}} = h^{\widehat{a}\widehat{b}(r+s)} = \widehat{H}^{r+s}, \end{aligned}$$

which is exactly the group element computed by wEval. As a result, both algorithms return $\mathbf{A} \cdot \mathsf{bin}(\widehat{H}^{r+s}) \bmod 2$, and hence

$$\mathsf{wF}(\mathsf{sk},\mathsf{fk},(m,\mathsf{ct})) = \mathsf{wEval}(\mathsf{ek},(r,s),(m,\mathsf{ct})).$$

Indistinguishability. A ciphertext generated by wEnc can be rewritten as

$$\mathsf{ct} = \left(u^r, \widehat{v}^s, h^r \cdot \widehat{h}^s \cdot m\right) = \left(u^r, (v^{\widehat{a}})^s, h^r \cdot (h^{\widehat{a}})^s \cdot m\right)$$

$$= \left(u^r, v^{\widehat{a}s}, h^{r+\widehat{a}s} \cdot m\right) = \mathsf{Enc}(\mathsf{pk}, m; (r, \widehat{a}s)).$$

Since $\widehat{a}, s \xleftarrow{\$} \mathbb{Z}_q$ independently, the distribution of their product $\widehat{a}s$ is statistically close to the uniform distribution over $\mathbb{Z}_q$. Therefore, the ciphertext distribution induced by wEnc is statistically close to that induced by Enc, and indistinguishability follows.

Weak Pseudorandomness. We define the following games played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Suppose that $\mathcal{A}$ is allowed to make at most $Q = \mathsf{poly}(\lambda)$ oracle queries. An overview of the game-hopping sequence is given in Fig. 25, where the changes in each game relative to the previous one are highlighted.

- **Game 0.** This is the experiment $\mathsf{ExpWPR}_{\mathcal{A},0}$ defined in Fig. 9.
- **Game 1.** Same as Game 0, except that in Step 16, the group element R is sampled uniformly from $\mathbb{G}$ instead of being computed as $\widehat{H}^{r+s}$.
- **Game 2.** Same as Game 1, except that in Step 17, y is sampled uniformly from $\{0,1\}^\ell$ independently of R. Notice that the view of $\mathcal{A}$ in Game 2 is identically distributed to its view in the experiment $\mathsf{ExpWPR}_{\mathcal{A},1}$ defined in Fig. 9.

Let W_ξ denote the event that $\mathcal{A}$ outputs 1 in Game ξ for $\xi \in [0,2]$. Then we have the following claims.

Claim 19 $|\Pr[W_0] - \Pr[W_1]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. This claim follows from the DLIN assumption. Since $\mathcal{A}$ may make multiple oracle queries, we define a sequence of games H_i for $i \in [1, Q+1]$ that interpolate between Game 0 and Game 1. A concrete description of H_i is given in Fig. 26.

- H_i: Same as Game 0, except that we modify the way in which $\mathcal{C}$ generates R in Step 16. Specifically, for the i'-th oracle query where $i' < i$, R is picked uniformly from $\mathbb{G}$, while for the i'-th oracle query where $i' \geq i$, R is computed as in Game 0 by $R \leftarrow \widehat{H}^{r+s}$.

Let X_i denote the event that $\mathcal{A}$ outputs 1 in H_i. Observe that H_1 and H_{Q+1} are identical to Game 0 and Game 1, respectively. Thus, it suffices to prove that for every $i \in [1, Q]$, $|\Pr[X_i] - \Pr[X_{i+1}]| \leq \mathsf{negl}(\lambda)$.

To this end, we construct from $\mathcal{A}$ a simulator $\mathcal{B}$ against the DLIN assumption that simulates either H_i or H_{i+1}. The behavior of $\mathcal{B}$ is identical to that of the challenger $\mathcal{C}$ in H_i, except that the following values are derived from the given DLIN instance $(g_1, g_2, g_3, g_1^\sigma, g_2^\tau, T)$, where $g_1, g_2, g_3 \xleftarrow{\$} \mathbb{G}$, $\sigma, \tau \xleftarrow{\$} \mathbb{Z}_q$, and T is either $g_3^{\sigma+\tau}$ or sampled uniformly from $\mathbb{G}$: (1) the elements h, $\widehat{v}$, $\widehat{h}$, and $\widehat{H}$ in the setup; and (2) the ciphertext ct and the group element R in the i-th oracle query. A concrete description of $\mathcal{B}$ is given in Fig. 27, with the differences relative to the challenger $\mathcal{C}$ of H_i highlighted.

Game 0

$h \xleftarrow{\$} \mathbb{G}$
$x, y \xleftarrow{\$} \mathbb{Z}_q^*$
$u = h^{x^{-1}}$, $v = h^{y^{-1}}$
$\widehat{a}, \widehat{b} \xleftarrow{\$} \mathbb{Z}_q$
$\widehat{v} = v^{\widehat{a}}$, $\widehat{h} = h^{\widehat{a}}$, $\widehat{H} = h^{\widehat{a}\widehat{b}}$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = (u, v, h)$, $\mathsf{sk} = (x, y)$
$\mathsf{ek} = (\widehat{v}, \widehat{h}, \widehat{H}, \mathbf{A})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $r, s \xleftarrow{\$} \mathbb{Z}_q$
    $\mathsf{ct} = (u^r, \widehat{v}^s, h^r \cdot \widehat{h}^s \cdot m)$
    $R = \widehat{H}^{r+s}$
    $\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(R) \bmod 2$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Game 1

$h \xleftarrow{\$} \mathbb{G}$
$x, y \xleftarrow{\$} \mathbb{Z}_q^*$
$u = h^{x^{-1}}$, $v = h^{y^{-1}}$
$\widehat{a}, \widehat{b} \xleftarrow{\$} \mathbb{Z}_q$
$\widehat{v} = v^{\widehat{a}}$, $\widehat{h} = h^{\widehat{a}}$, $\widehat{H} = h^{\widehat{a}\widehat{b}}$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = (u, v, h)$, $\mathsf{sk} = (x, y)$
$\mathsf{ek} = (\widehat{v}, \widehat{h}, \widehat{H}, \mathbf{A})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $r, s \xleftarrow{\$} \mathbb{Z}_q$
    $\mathsf{ct} = (u^r, \widehat{v}^s, h^r \cdot \widehat{h}^s \cdot m)$
    $R \xleftarrow{\$} \mathbb{G}$
    $\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(R) \bmod 2$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Game 2

$h \xleftarrow{\$} \mathbb{G}$
$x, y \xleftarrow{\$} \mathbb{Z}_q^*$
$u = h^{x^{-1}}$, $v = h^{y^{-1}}$
$\widehat{a}, \widehat{b} \xleftarrow{\$} \mathbb{Z}_q$
$\widehat{v} = v^{\widehat{a}}$, $\widehat{h} = h^{\widehat{a}}$, $\widehat{H} = h^{\widehat{a}\widehat{b}}$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = (u, v, h)$, $\mathsf{sk} = (x, y)$
$\mathsf{ek} = (\widehat{v}, \widehat{h}, \widehat{H}, \mathbf{A})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $r, s \xleftarrow{\$} \mathbb{Z}_q$
    $\mathsf{ct} = (u^r, \widehat{v}^s, h^r \cdot \widehat{h}^s \cdot m)$
    $R \xleftarrow{\$} \mathbb{G}$
    $\mathsf{y} \xleftarrow{\$} \{0,1\}^{\ell}$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Fig. 25: Weak Pseudorandomness of $\mathsf{wPRF}_{\mathsf{le}}$

Let E denote the setup event that $h \neq 1_{\mathbb{G}}$ and $\widehat{h} \neq 1_{\mathbb{G}}$. Conditioned on E, it is easy to verify that $(h, \widehat{h}, \widehat{H})$ is uniformly distributed over $(\mathbb{G} \setminus \{1_{\mathbb{G}}\})^2 \times \mathbb{G}$ in H_i, H_{i+1}, and the simulation of $\mathcal{B}$. Besides, in all three probability spaces,

$$\Pr[\neg E] = 1 - \left(1 - \frac{1}{q}\right)^2 \leq \frac{2}{q}.$$

Moreover, the ciphertext in the i-th oracle query satisfies

$$\mathsf{ct} = \left((g_1^{\sigma})^{x^{-1}}, (g_2^{\tau})^{y^{-1}}, g_1^{\sigma} \cdot g_2^{\tau} \cdot m\right) = \left(u^{\sigma}, \widehat{v}^{\tau}, h^{\sigma} \cdot \widehat{h}^{\tau} \cdot m\right),$$

which has the required form due to wEnc, with encryption randomness (σ, τ).

Let Y_0 and Y_1 denote the events that $\mathcal{B}$ outputs 1 when $T = g_3^{\sigma+\tau}$ and when $T \xleftarrow{\$} \mathbb{G}$, respectively. If $T = g_3^{\sigma+\tau}$, then $R = \widehat{H}^{\sigma+\tau}$ in the i-th query; if $T \xleftarrow{\$} \mathbb{G}$, then R in the i-th query is uniform over $\mathbb{G}$. Therefore,

$$\Pr[X_i \mid E] = \Pr[Y_0 \mid E], \quad \Pr[X_{i+1} \mid E] = \Pr[Y_1 \mid E].$$

H_i :

$h \xleftarrow{\$} \mathbb{G}$
$x, y \xleftarrow{\$} \mathbb{Z}_q^*$
$u = h^{x^{-1}}$, $v = h^{y^{-1}}$
$\widehat{a}, \widehat{b} \xleftarrow{\$} \mathbb{Z}_q$
$\widehat{v} = v^{\widehat{a}}$, $\widehat{h} = h^{\widehat{a}}$, $\widehat{H} = h^{\widehat{a}\widehat{b}}$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = (u, v, h)$, $\mathsf{sk} = (x, y)$
$\mathsf{ek} = (\widehat{v}, \widehat{h}, \widehat{H}, \mathbf{A})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $r, s \xleftarrow{\$} \mathbb{Z}_q$
    $\mathsf{ct} = (u^r, \widehat{v}^s, h^r \cdot \widehat{h}^s \cdot m)$
    if $i' < i$
      $R \xleftarrow{\$} \mathbb{G}$
    else if $i' \geq i$
      $R = \widehat{H}^{r+s}$
    $\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(R) \bmod 2$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Fig. 26: Game Hopping Between Game 0 and Game 1

By difference lemma, it follows that

$$|\Pr[X_i] - \Pr[Y_0]| \leq \Pr[\neg E] \leq \frac{2}{q}, \quad |\Pr[X_{i+1}] - \Pr[Y_1]| \leq \Pr[\neg E] \leq \frac{2}{q}.$$

Combining the above inequalities using the triangle inequality, we obtain

$$\begin{aligned} |\Pr[X_i] - \Pr[X_{i+1}]| &\leq |\Pr[X_i] - \Pr[Y_0]| + |\Pr[Y_0] - \Pr[Y_1]| \\ &\quad + |\Pr[Y_1] - \Pr[X_{i+1}]| \\ &\leq \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{B}}(\lambda) + \frac{4}{q} \leq \mathsf{negl}(\lambda), \end{aligned}$$

from which the claim follows. □

Claim 20 $|\Pr[W_1] - \Pr[W_2]| \leq \mathsf{negl}(\lambda)$.

Proof of Claim. The claim follows directly from Lemma 3. □

Simulator $\mathcal{B}$ on Input $(g_1, g_2, g_3, g_1^{\sigma}, g_2^{\tau}, T)$:

$h \leftarrow g_1$
$x, y \xleftarrow{\$} \mathbb{Z}_q^*$
$u = h^{x^{-1}}$, $v = h^{y^{-1}}$

$\widehat{v} \leftarrow g_2^{y^{-1}}$, $\widehat{h} \leftarrow g_2$, $\widehat{H} \leftarrow g_3$
$\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$
$\mathsf{pk} = (u, v, h)$, $\mathsf{sk} = (x, y)$
$\mathsf{ek} = (\widehat{v}, \widehat{h}, \widehat{H}, \mathbf{A})$
$i' \leftarrow 1$
return $\mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{pk}, \mathsf{sk}, \mathsf{ek})$
  where $\mathcal{O}(\cdot)$ on input m:
    if $i' > Q$
      return $\perp$
    $r, s \xleftarrow{\$} \mathbb{Z}_q$
    if $i' \neq i$
      $\mathsf{ct} = (u^r, \widehat{v}^s, h^r \cdot \widehat{h}^s \cdot m)$
    else if $i' = i$
      $\mathsf{ct} = ((g_1^{\sigma})^{x^{-1}}, (g_2^{\tau})^{y^{-1}}, g_1^{\sigma} \cdot g_2^{\tau} \cdot m)$
    if $i' < i$
      $R \xleftarrow{\$} \mathbb{G}$
    else if $i' > i$
      $R = \widehat{H}^{r+s}$
    else if $i' = i$
      $R \leftarrow T$
    $\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(R) \bmod 2$
    $i' \leftarrow i' + 1$
    return $(\mathsf{ct}, \mathsf{y})$

Fig. 27: The Simulator for Proving $\mathsf{H}_i \approx \mathsf{H}_{i+1}$

H Witness PRF for CSIDH-Based ElGamal Encryption

H.1 Group Action and Assumption

We use the following abstraction of the CSIDH group action [11, 5]. Let a non-trivial finite abelian group $\mathcal{G}$ of odd order act freely and transitively on a finite set $\mathcal{X}$, with a public base element $E_0 \in \mathcal{X}$. Thus, $a \mapsto a \star E_0$ is a bijection, and for all $a, b \in \mathcal{G}$ and $E \in \mathcal{X}$,

$$a \star (b \star E) = (ab) \star E = b \star (a \star E).$$

As in [5], we assume polynomial-size representations supporting efficient uniform sampling from $\mathcal{G}$, action evaluation, and equality testing in $\mathcal{X}$. All parameters depend on the security parameter λ.

Definition 23 (Decisional CSIDH Assumption). *The decisional CSIDH assumption states that $(a \star E_0, b \star E_0, (ab) \star E_0)$ and $(a \star E_0, b \star E_0, U)$ are computationally indistinguishable, where $a, b \xleftarrow{\$} \mathcal{G}$ and $U \xleftarrow{\$} \mathcal{X}$ are sampled independently. The public action parameters and E_0 are available to the adversary.*

H.2 Construction

For simplicity, we recall the ElGamal-like encryption scheme of [5]. Its decryption enumerates a publicly fixed, efficiently enumerable message space $\mathcal{M}_{\mathsf{pk}} \subseteq \mathcal{G}$ of polynomial size.

- $\mathsf{KGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$. Sample $x \xleftarrow{\$} \mathcal{G}$ and compute $E_x = x \star E_0$. Return the public key pk and secret key sk as

$$\mathsf{pk} = E_x, \quad \mathsf{sk} = x.$$

- $\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$. To encrypt a message $m \in \mathcal{M}_{\mathsf{pk}}$, sample encryption randomness $s \xleftarrow{\$} \mathcal{G}$, and compute the ciphertext as

$$\mathsf{ct} = (c_0, c_1) = (s \star E_0,\ m \star (s \star E_x)).$$

- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to m$. Given $\mathsf{sk} = x$ and $\mathsf{ct} = (c_0, c_1)$, try each $m \in \mathcal{M}_{\mathsf{pk}}$ and check whether

$$c_1 = m \star (x \star c_0).$$

  Return the unique matching message, or $\perp$ if none exists.

For a larger message space, one may instead use hashed ElGamal by applying a randomness extractor with a public seed (e.g., via leftover hashing as in Lemma 3) to the shared element $s \star E_x = x \star c_0$ and XORing the output with a bit-string message of the same length. This changes only the second ciphertext component and leaves the witness PRF below unchanged, so we use the simpler form as ElGamal for exposition.

We use $\mathsf{bin}(E)$ to denote an injective binary representation of $E \in \mathcal{X}$, and assume that all such representations have a common length $n = \mathsf{poly}(\lambda)$. Our witness PRF $\mathsf{wPRF}_{\mathsf{csidh}} = (\mathsf{wGen}, \mathsf{wEnc}, \mathsf{wF}, \mathsf{wEval})$ for the above encryption scheme is defined as follows.

- $\mathsf{wGen}(\mathsf{pk}) \to (\mathsf{ek}, \mathsf{fk})$. Sample $\widehat{x} \xleftarrow{\$} \mathcal{G}$ and compute $E_{\widehat{x}} = \widehat{x} \star E_0$. Sample a matrix $\mathbf{A} \xleftarrow{\$} \{0,1\}^{\ell \times n}$. Return the evaluation key ek and function key fk as

$$\mathsf{ek} = (E_{\widehat{x}}, \mathbf{A}), \quad \mathsf{fk} = (\widehat{x}, \mathbf{A}).$$

- $\mathsf{wEnc}(\mathsf{pk}, \mathsf{ek}, m) \to (s, \mathsf{ct})$. Sample randomness $s \xleftarrow{\$} \mathcal{G}$, then return s and $\mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, m; s)$.
- $\mathsf{wF}(\mathsf{sk}, \mathsf{fk}, (m, \mathsf{ct})) \to \mathsf{y} \in \{0,1\}^\ell$. Given $\mathsf{fk} = (\widehat{x}, \mathbf{A})$ and $\mathsf{ct} = (c_0, c_1)$, compute $R = \widehat{x} \star c_0$, and return

$$\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(R) \bmod 2.$$

- $\mathsf{wEval}(\mathsf{ek}, s, (m, \mathsf{ct})) \to \mathsf{y} \in \{0,1\}^\ell$. Given $\mathsf{ek} = (E_{\widehat{x}}, \mathbf{A})$, compute $R = s \star E_{\widehat{x}}$, and return

$$\mathsf{y} = \mathbf{A} \cdot \mathsf{bin}(R) \bmod 2.$$

Under the decisional CSIDH assumption, $\mathsf{wPRF}_{\mathsf{csidh}}$ is a witness PRF for the above encryption scheme. The proof follows the same argument as for ElGamal in Appendix E, with the group action replacing exponentiation and the decisional CSIDH assumption replacing DDH, and is therefore omitted.