Simple Public Key Anamorphic Encryption and Signature using Multi-Message Extensions

Shalini Banerjee[1], Tapas Pal[1], Andy Rupp[1,2], and Daniel Slamanig[3]

[1] Karlsruhe Institute of Technology, KASTEL Security Research Labs
[2] University of Luxembourg
[3] Research Institute CODE, Universität der Bundeswehr München

Abstract. Anamorphic encryption (AE) considers secure communication in the presence of a powerful surveillant (typically called a "dictator") who only allows certain cryptographic primitives and knows all the secret keys in a system. The basic idea is that there is a second (anamorphic) mode of encryption that allows to transmit an anamorphic message using a double key to a receiver that can decrypt this message using a double key. From the point of view of the dictator the encryption keys as well as the ciphertexts in the regular and anamorphic mode are indistinguishable. The most recent works in this field consider public key anamorphic encryption (PKAE), i.e., the sender of an anamorphic message requires an encryption double key (or no key at all) and the receiver requires a decryption double key. Known constructions, however, either work only for schemes that are mostly of theoretical interest or come with conceptual limitations.

In this paper we ask whether we can design such PKAE schemes without such limitations and being closer to PKE schemes used in practice. In fact, such schemes are more likely to be allowed by a cognizant dictator. Moreover, we initiate the study of identity-based anamorphic encryption (IBAE), as the IBE setting seems to be a natural choice for a dictator. For both PKAE and IBAE, we show how well-known IND-CPA and IND-CCA secure primitives can be extended by an anamorphic encryption channel. In contrast to previous work, we additionally consider CCA (rather than just CPA) security notions for the anamorphic channel and also build upon CPA (rather than just CCA) secure PKE.

Finally, we ask whether it is possible to port the recent concept of anamorphic signatures, which considers constructing symmetric anamorphic channels in case only signature schemes are allowed by the dictator, to the asymmetric setting, which we denote by public-key anamorphic signatures (PKAS). Also here we consider security beyond IND-CPA for the anamorphic channel.

Contents

1 Introduction

Anamorphic encryption, introduced in [30] and subsequently explored in [26, 3, 11, 13, 12, 31], is a very recent concept that allows individuals to confidentially exchange messages in a very restricted setting where a powerful surveillant (typically called a "dictator") has full access to the secret keys in the system. Such a capability can foster a strong foundation for enabling privacy in the face of a powerful authoritarian government where every citizen is constantly surveilled. The basic idea behind a PKE scheme to be anamorphic, is that there is a second mode of encryption (the anamorphic mode) that allows to encrypt an anamorphic message that can be recovered by the legitimate holder of the secret key. But from the perspective of the dictator, these ciphertexts look indistinguishable from regular ciphertexts and so the dictator is oblivious of these anamorphic messages. Moreover, keys used for the anamorphic mode must also be indistinguishable from the ones used for the regular mode. This is achieved by having a second key (the so-called double key) that can be used by the sender and the receiver (and has to be exchanged covertly in advance). The general idea is to incorporate a covert channel using a symmetric or asymmetric encryption scheme on top of an existing PKE scheme. While the asymmetric setting will be the focus of this paper, we want to mention that for a symmetric setting one usually relies either on some randomness recoverability property of the PKE [3, 27], on hybrid encryption or an IBE-to-CCA transform [11]. Interestingly, there are well-known CCA-secure public-key encryption schemes (or generic approaches to construct such schemes) such as the Naor-Yung double encryption paradigm [29], (RSA)-OAEP [5] or Cramer-Shoup [15] that are inherently anamorphic.

The most recent works in this fast-pacing field focus on schemes with asymmetric anamorphic channels [11, 31]. Additionally, they investigate the limitations of constructing anamorphic schemes [13,12]. Arguably, asymmetric anamorphic channels, so-called public key anamorphic encryption (PKAE), seem to be superior to symmetric ones. In a PKAE the keys for the anamorphic mode are asymmetric, i.e., there is an encryption double key dk and a decryption double key tk. This is beneficial as in the case of multiple senders, no complex key management is required in order to prevent senders to reading anamorphic messages sent to the receiver by other senders.

Public-key anamorphism. Persiano, Phan and Yung in [31] propose a variant of the PKAE notion where $\mathsf{dk} = \epsilon$, which interestingly eliminates the need of communicating the encryption double key dk privately to the receiver. They complement this by providing a proof of concept of the notion using the CCA-secure Koppula-Waters (KW) scheme [25]. Additionally, they initiate a theoretical investigation on the link between CPA-to-CCA compilers and public key anamorphism to extract a set of generic "reduction-based" properties required in the underlying PKE, satisfied by the KW scheme. Worth mentioning is also that both the PKAE constructions based on KW proposed in [31] and Naor-Yung (NY) proposed in [11] rely on a common feature of the underlying PKE scheme: the setup procedure generates multiple key pairs $(\mathsf{sk}_i, \mathsf{pk}_i)_{i\in[n]}$, while

retaining only a subset of secret keys sk_i, which makes them a natural choice for constructing PKAE by embedding anamorphic message amsg within ciphertext ct_i and allowing the receiver to retain the corresponding sk_i (which ideally should have been forgotten). Nevertheless, we observe that most of the well-known PKE schemes and, in particular, practically used ones do not fulfil this property.

In another recent work, Catalano, Giunta and Migliaro [11] design PKAE schemes, where $\mathsf{dk} \neq \epsilon$, which they call fully-asymmetric AE schemes. Such schemes are asymmetric but require to share the encryption key to potential senders. In particular, the authors focus on Homomorphic Anamorphic Encryption (HAE) and demonstrate that NY instantiated with any IND-CPA secure fully-homomorphic encryption scheme and fully-homomorphic NIZK [1], the Gentry-Sahai-Waters scheme [23] and Cramer-Shoup Lite [15] are fully-asymmetric. Moreover, in a follow-up work Catalano, Giunta and Migliaro [12] show that such a type of scheme can be obtained from any IND-CPA secure PKE with sufficiently high min-entropy ciphertexts using $i\mathcal{O}$. Also here it is worth noting that this does not cover PKE schemes that (or variants thereof) are used in practice. Importantly, all these constructions in [11] are anamorphic triplets rather than anamorphic extensions (introduced in [3]), i.e., the double keys have to be generated at the time the normal keys are generated. However, it is desirable that schemes support on-the-fly anamorphic key generation, i.e., double keys can be generated based on existing keys of the normal PKE.

Moreover, interestingly all these existing anamorphic schemes build on top of *CCA-secure* PKE schemes, but with respect to the anamorphic message (i.e., the indistinguishability of anamorphic mode from normal mode) they just achieve protection in the *CPA-sense*.

This makes us think whether these limiting restrictions, i.e., reduction-based properties, being limited to anamorphic triplets rather than extensions as well as relying on CCA-secure PKEs, are inherently linked to the design of asymmetric anamorphic channels. In particular, we ask

Can we design simple public key anamorphic encryption schemes without such limitations and ideally based on practically used PKE schemes?

In fact being able to focus on much simpler PKE schemes used in practice (or variants thereof), are more likely to be allowed by a cognizant dictator. We believe this is a promising direction to explore in the sense that a positive result would give us hope towards establishing PKAE as a practical cryptographic primitive, while a negative result would imply relying on the existence of specialized PKE schemes, that might be outlawed by the dictator.

Towards an affirmative answer, we first give constructions for PKAE (with $\mathsf{dk} \neq \epsilon$) based on the ElGamal and Dual-Regev PKE schemes where we embed an anamorphic message in $\mu > 1$ ciphertexts of the respective PKE (which we call multi-message extension). Note that ElGamal is used as a KEM in the popular Elliptic Curve Integrated Encryption Scheme (ECIES) and Dual-Regev is the basic scheme underlying many modern lattice-based schemes such as Kyber. Second, many practically used schemes (such as Kyber) apply the Fujisaki-Okamoto transform [19, 20] to a weakly (i.e., OW-CPA or IND-CPA) secure scheme to

obtain an IND-CCA secure one. Here, we show that when the ciphertext of the underlying PKE is pseudo-random, then together with a suitable encoding, anamorphic messages can be embedded into the FO-transformed PKE such that the anamorphic encryption channel achieves replayable CCA (RCCA) [10] security. To the best of our knowledge, this is the first work that considers security stronger than IND-CPA for the anamorphic message (though many works in the past relied on CCA secure schemes for the normal message).

Identity-based anamorphic encryption. When considering a setting where a dictator wants to control the entire communication, it seems natural to consider identity-based encryption [8]. In this setting, there is an entity which generates the private keys for all the users (identities) from a compact msk and thus there is an inherent backdoor (this msk) already built into the system. Thus, it seems likely that a dictator would allow such encryption mechanisms, which in addition would save the dictator from the effort of collecting all the private keys. Consequently, it seems natural to ask

Can we port the concept of anamorphic encryption to the identity-based encryption setting?

We demonstrate how to realize anamorphic channels in such a world and construct identity-based anamorphic encryption (IBAE) based on the Boneh-Franklin IBE [8]. Moreover, we extend our results on the FO-transform from the PKAE to the IBAE setting. Interestingly, this allows us to obtain IBAE with strong security for the anamorphic channel for *standardized* IBE.

Anamorphic signatures. Besides the dedicated study on encryption, recently Kutylowski et al. [26] investigated whether such a confidential message transmission can still work in case the dictator forbids the use of encryption, and only authentication primitives such as signatures are available. They considered symmetric anamorphic channels and provided an affirmative answer to that question. After considering simpler constructions for PKAE is seems natural to ask

Are there public-key anamorphic signatures (PKAS), and can we design such schemes as a simple combination of existing signature and PKE schemes?

Towards this goal, we introduce a general framework for PKAS and showcase constructions based on two known signature schemes, one being *standardized*.

1.1 Our Contributions

In the following we briefly summarize our contributions:

- We introduce the concept of μ-message anamorphic extensions to construct public-key anamorphic encryption schemes from simple CPA-secure PKE schemes that (or variants thereof) are used in practice. In particular, both the encryption of regular messages and anamorphic messages as well as their decryption should essentially resemble that of a *single* PKE scheme. This concept allows to embed an anamorphic message in $\mu > 1$ ciphertexts of a PKE scheme and we demonstrate such extensions for the well-known ElGamal and the Dual-Regev schemes. We note that our Dual-Regev construction

can be considered as the first PKAE based on lattices, given the existing constructions as either based on groups or SKE. We follow the same approach for the Fujisaki-Okamoto (FO) transform often found in practical IND-CCA secure PKE schemes (such as Kyber). Interestingly, although many previous works used IND-CCA secure PKEs to create anamorphic channels, these works did restrict to IND-CPA security for the anamorphic message. We also consider security beyond IND-CPA and show that with our FO-approach, we can achieve strong RCCA security where strong refers to the model that provides the secret key of the PKE scheme to the adversary.
- We initiate the study of identity-based anamorphic encryption (IBAE). We show that the Boneh-Franklin (BF) IBE admits a 2-message IBAE. Moreover, we show that the FO-approach to construct an anamorphic channel can be equivalently applied to the IBE setting. This covers CCA secure variants of the BF and the Sakai-Kasahara IBEs which are standardized [18].
- We introduce the concept of public-key anamorphic signature extensions (PKASE). Our goal here is to combine well-known signature schemes and PKE schemes in a way that the embedding is simple. We introduce a generic framework that covers signature schemes which are canonical candidates for anamorphic extensions and in particular have at least one uniformly random component. The random components of a few signatures suffice to embed a pseudo-random ciphertext. We showcase our framework for Waters signatures and linear encryption as well as Boneh-Boyen-Sacham (BBS) signatures, being in the process of *standardization*, and FO-ElGamal encryption and achieve security beyond IND-CPA for the latter.

We depict our anamorphic encryption and signature schemes in Table 1.

1.2 Technical Overview

Subsequently, we provide a technical overview of our contributions and how our approaches work, with details deferred to the respective sections.

μ-message PKAE extensions for simple IND-CPA secure PKE. We have already discussed that constructing PKAE is known to work when making an additional assumption such as when relying on PKE where the key generation algorithm samples various key pairs of the underlying CPA-secure PKE schemes and a *subset* of the secret keys is sufficient for decryption, which gives rise to a public key anamorphic channel. Nevertheless, only theoretically interesting schemes are known to provide such a feature. Moreover, for the schemes in [11], despite being not very close to schemes used in practice, the limitation to anamorphic triplets rather than anamorphic extensions seems inherent.

Alternatively, we could take inspiration from recent works [26, 27] constructing secret key anamorphic encryption schemes (i.e., $\mathsf{dk} = \mathsf{tk}$) using the randomness-recovering properties of PKE schemes. We recall that for such a scheme the secret key can recover the random coins sampled during encryption. Let r be the randomness used to encrypt msg by $\mathsf{PKE.Enc}(\mathsf{pk}, \mathsf{msg}; r) = \mathsf{ct}$ then there is a randomness recovery algorithm that extracts r along with msg using sk. To create an

PKAEE				
Our Work	PKE	$\mu = \#\mathsf{ct}$	$\widehat{\mathcal{M}}$	Security
Sec. 3.1	ElGamal	2	restricted	IND-CPA
Sec. 3.2	DualRegev	2	restricted	IND-CPA
Sec. 3.3	FO-HashedElGamal	4	unrestricted	sIND-RCCA

IBAEE				
Our Work	IBE	$\mu = \#\mathsf{ct}$	$\widehat{\mathcal{M}}$	Security
Sec. 4.1	BF	2	restricted	IND-CPA
Sec. F	FO-BF	4	unrestricted	sIND-RCCA

PKASE					
Our Work	PKE	Sig	$\mu = \#\sigma$	$\widehat{\mathcal{M}}$	Security
Sec. 5.1	LinEnc	Waters	3	unrestricted	IND-CPA
Sec. 5.2	ElGamal	BBS	4	unrestricted	IND-CPA
Sec. 5.2	FO-ElGamal	BBS	5	unrestricted	sIND-RCCA + IND-CCA

Table 1: Our anamorphic encryption and signature schemes. Here, we denote by $\#\mathsf{ct}$ and $\#\sigma$, the total number of (regular) ciphertexts and signatures of the underlying PKE, IBE and Sig schemes needed to construct our PKAEE, IBAEE and PKASE respectively; by $\widehat{\mathcal{M}}$, the anamorphic message space where "restricted" means polynomial-sized and "unrestricted" means exponential-sized; by RCCA replayable CCA; by FO Fujisaki-Okamoto; by BF Boneh-Franklin; by LinEnc linear encryption; "Security" means the security of the anamorphic message.

anamorphic channel using this property, one needs a symmetric key encryption scheme SKE to encrypt an anamorphic message amsg and use $\mathsf{SKE.Enc}(\widetilde{\mathsf{sk}}, \mathsf{amsg})$ as the random coins r while computing ct. The secret key $\widetilde{\mathsf{sk}}$ plays the role of the double key. If the SKE scheme produces pseudorandom ciphertexts then the dictator remains oblivious about the presence of such an anamorphic message inside ct having access to $(\mathsf{pk}, \mathsf{sk}, \mathsf{ct})$. A naïve way to make this work in the case of PKAE is to replace the underlying SKE having pseudorandom ciphertexts with a PKE having the same property.

However, our approach towards PKAE is not to use any specific property of the PKE but to devise an approach that is capable of covering various well-known PKE schemes that (or variations thereof) are used in practice. Our design goal is to rely on a *single* PKE scheme both for the regular and anamorphic messages. It is however not clear a priori how to embed the anamorphic message amsg into a single ciphertext ct of the underlying PKE scheme such that decryption recovers both msg and amsg without relying on aforementioned assumptions or properties. To get around this apparent constraint, we embed an anamorphic message amsg within *multiple* ciphertexts of the underlying PKE. We formalize this as public-key anamorphic μ-message extension and consequently an anamorphic ciphertext consists of μ (not necessarily consecutively sent) ciphertexts of

the underlying PKE. This can be seen as distributing a ciphertext of the PKE carrying amsg over μ ciphertexts containing regular messages $(m_1, \ldots, m_\mu)$ generated with the *same* PKE. To formally capture this, we additionally introduce the notion of PKAE-compatibility, which requires to generate double-keys on the fly, and importantly *ciphertext extractibility*, demanding that the existence of an efficiently computable function F such that the anamorphic message amsg can be computed by the decryption algorithm of the PKE on ciphertext $F(\mathsf{act})$ computed from the anamorphic ciphertext act, i.e., $\mathsf{PKE.Dec}(\mathsf{tk}, F(\mathsf{act})) = \mathsf{amsg}$. Latter ensures that decryption of an anamorphic ciphertext can be performed using the decryption algorithm of the PKE. We then demonstrate a public-key anamorphic 2-message extension for the ElGamal and the Dual-Regev encryption schemes. To give an idea for ElGamal with public key $\mathsf{pk} = g^{\mathsf{sk}}$ where a ciphertext to message M is $C = (g^\kappa, pk^\kappa \cdot M)$ for random κ. We can set the double keys as $\mathsf{dk} = (g^\alpha, g^{\alpha \cdot \mathsf{sk}})$ and $\mathsf{tk} = \alpha$. Embedding an anamorphic message amsg into two ciphertexts for messages M_1 and M_2 will result in $(r_1, c_1) = (g^\kappa, \mathsf{pk}^\kappa \cdot M_1)$ and $(r_2, c_2) = (\mathsf{dk}_1^\kappa \cdot g^{\mathsf{amsg}}, \mathsf{dk}_2^\kappa \cdot \mathsf{pk}^{\mathsf{amsg}} \cdot M_2)$. Decrypting the normal messages works as usual and the anamorphic message can be recovered by calling ElGamal decryption on (r_1, r_2) and finding $\log g^{\mathsf{amsg}}$.

Why μ-message extension? One could wonder whether it is hard to embed an ElGamal ciphertext $(g^\kappa, g^{\kappa \cdot \mathsf{tk} + \mathsf{amsg}})$ into a *single* ElGamal ciphertext $(r_1 = g^\alpha, c_1 = g^{\alpha \cdot \mathsf{sk} + \mathsf{msg}})$. Since α is uniformly random over $\mathbb{Z}_q$, it is not possible to extract something meaningful if we try to embed $g^{\kappa \cdot \mathsf{tk} + \mathsf{amsg}}$ into r_1. On the other hand, both g^κ and r_1 are statistically close. Hence, we can directly set $\alpha = \kappa$ for recovering the correct anamorphic message during decryption. Consequently, the only possible option left is to somehow encode $g^{\kappa \cdot \mathsf{tk} + \mathsf{amsg}}$ into c_1 where $\alpha = \kappa$. Thus, we have the following equation $\kappa \cdot \mathsf{tk} + \mathsf{amsg} = \kappa \cdot \mathsf{sk} + \mathsf{msg}$. Given a pair of messages $\mathsf{msg}, \mathsf{amsg}$, this equality cannot hold with high probability when tk, sk are independently and uniformly sampled from $\mathbb{Z}_q$.

Anamorphic Clans. While the entire revolutionary theme of anamorphic cryptography is built on top of a totalitarian government regime, it is quite natural to imagine a setting where secret rebellious communities exist within. A prominent use-case would be an underground community liaising secretly to advocate freedom of citizens from the dictatorship rule. Additionally, the members of such communities must be subject to background checks to avoid infiltration by the government, and hence managed by a leader; we call such communities *anamorphic clans*.

Having introduced the concept of anamorphic clans, consider a scenario where the encryption double key dk of an anamorphic clan member is revealed to the dictator. A dictator could simply procure dk when getting distributed within the clan, or by corrupting its member(s). Intuitively, one might expect the dictator to forcefully demand the corresponding decryption double key tk. However, the dictator may have a stronger motive of silently surveilling the communication patterns to learn the communicating peers of the alleged clan member, and hence not demand tk immediately. It could also be the case that the dictator is unable to link the procured dk to a clan member and hence wishes to investigate

who is anamorphically communicating with whom. This demands us to study a *stronger* security notion which is beyond the usual IND-CPA notion, where the adversary compromises both ask and dk. Additionally, the dictator could also learn a list of anamorphic messages and corresponding anamorphic ciphertexts from a corrupted clan member. This motivates us to consider an *even stronger* security notion where we allow the adversary to access an oracle $\mathsf{aDec}(\mathsf{tk}, \mathsf{act})$ that decrypts anamorphic ciphertexts. In other words, we consider IND-CCA security for anamorphic messages.

PKAE extensions with security beyond IND-CPA. Another goal of this paper is to build PKAE for popular IND-CCA secure schemes using the same idea of a multi-message extension. Existing results for PKAE based on IND-CCA secure PKEs are based on theoretical constructions such as Koppula-Waters. Moreover, and interestingly, previous work though looking at anamorphisms for CCA secure schemes only guarantee IND-CPA security for the anamorpic messages and consequently no integrity protection at all. We study the use of the popular FO transformation which is particular relevant in the construction of post-quantum PKEs such as Kyber. We recall that FO encrypts a random message r with a weakly secure (e.g., IND-CPA secure PKE) using randomness derived from a hash function H on $H(r, m)$ with m being the message that one wants to encrypt. The random message r is then used as a basis to derive a key $G(r)$, with G being another hash function, for a symmetric encryption (SKE) scheme which then encrypts m. The overall scheme then achieves IND-CCA security in the random oracle model. The basic idea of our construction is to use two FO-layers, an inner and outer layer of FO encryption. Basically, the inner-layer is used to encrypt the anamorphic message amsg where the PKE scheme and the SKE scheme are required to have pseudorandom ciphertexts. Then we use a pseudo-random encoding (e.g,. Elligator Squared in case of elliptic curves) to encode the inner-layer ciphertext into the message space of the PKE of the outer-layer FO encryption. Due to the property of the FO-transformation, i.e., randomness recoverability, we can thus use the randomness obtained from decrypting the outer-layer to decode this into a ciphertext of the inner-layer and decrypt. Interestingly, for such schemes we achieve strong RCCA security for the anamorphic message.

Identity-based anamorphic encryption (IBAE). Deploying anamorphic clans using PKAE would require the anamorphic clan lead to generate the double key-pairs $(\mathsf{dk}, \mathsf{tk})$, sending the double decryption keys to the members while registering double encryption keys for all the clan members. Clearly, such a solution would not be desirable in light of the overall complexity in generating and storing the double keys, and would additionally lead to scalability concerns. To overcome this, we could think of using identity-based encryption to establish the anamorphic channel, where the clan lead generates master key-pairs $(\mathsf{mpk}, \mathsf{msk})$, derives and distributes double private keys $\mathsf{sk}_{\mathsf{id}}$ to clan members. Nevertheless we are unsure how to realize such an anamorphic IBE channel within the dictator approved PKE scheme. Thus we ask : *could we reformulate anamorphic encryp-*

tion through the lens of identity-based encryption? More specifically, could we design a covert anamorphic IBE channel within a regular IBE channel? This would require the dictatorship to authorize identity-based encryption but why would that be convincing? However, as argued within, and demonstrated by our following discussion, identity-based encryption could be the most natural way to formulate anamorphic cryptography. To start, in an identity-based encryption settings, a dictator would generate private keys associated with the public identity of its citizens at any stage during the existence of the system. This would also capture the powerful concept of silent surveillance within a dictatorship regime as any message encrypted with respect to any id can be decrypted by the dictator without forcing citizens to submit private keys on demand. Finally, a dictator can derive private keys as convenient, and hence there is no need to store the keys, contrary to the realization of an anamorphic clan using PKAE settings.

Therefore, IBE seems to be a premier candidate for an encryption scheme allowed by a dictator. We initiate the study of identity-based anamorphic encryption (IBAE) which allows to embed an IBE anamorphic channel inside an IBE. We present a concrete instantiation of anamorphic IBE based on the seminal Boneh-Franklin scheme [8] using our multi-message extension concept. A 2-IBAE associated with the Boneh-Franklin IBE [8], works roughly as follows. Unlike the anamorphic extension of the ElGamal encryption scheme where for two ciphertexts the first components (r_1, r_2) represent an ElGamal encryption of the anamorphic message, embedding amsg within two regular Boneh-Franklin ciphertexts is not that straightforward. We recall that, in Boneh-Franklin ciphertext $C = (r_2 = P^{\kappa_2}, c_2 = \mathsf{msg} \oplus \mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2}))$, the group element $g_{\mathsf{id}}^{\kappa_2}$ is not explicitly available. Rather, it only appears as an input to the hash function H_2. Therefore, if we set κ_2 following the ElGamal encryption extension, then it seems impossible to recover the anamorphic message amsg due to the one-wayness of H_2. At this point, the only solution appears to be encoding amsg outside the hash function H_2. Consequently, we use rejection sampling such that amsg could be extracted from c_2. More specifically, we sample κ_2 subject to $\mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2}) = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1})$; call it $\widehat{c}$. We then treat $(r_1 = P^{\kappa_1}, \widehat{c} = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1}))$ as a BF ciphertext and apply the regular decryption algorithm of BF to recover amsg. Using this technique we can convey anamorphic messages of at most $\mathrm{O}(\log \lambda)$ bits to ensure that encryption remains polynomial. Similar to PKAE, we then explore the use of the FO-transformation to create an anamorphic channel which achieves strong IND-RCCA security and allows for λ-bit anamorphic messages. Standardized Boneh-Franklin and Sakai-Kasahara are covered by our results.

Public-key anamorphic signatures extensions. Anamorphic signatures are a recent concept due to [26] but, so far, they are not considered in the context of asymmetric anamorphic channels. Our aim is to investigate such public-key anamorphic signatures (PKAS) based on simple combinations of well-known signature and PKE schemes. Our main observation is that there are well-known schemes where signatures contain some uniformly random components and the remaining components can be computed from the former in a "black-box way"

(e.g., without knowing their DLs). This opens up the possibility to embed pseudo-random ciphertexts into these random signature components. The number of random signature components and the number of ciphertext components, then controls how many signatures are required to transmit one anamorphic message. We put this intuition into a generic formal framework and show how to instantiate this framework based on Water's signatures [36] together with linear encryption [7] as well as BBS signatures [7, 34], currently being standardized, together with (FO-)ElGamal encryption, latter with the help of the Elligator Squared encoding [35]. We note that for signatures we are able to achieve IND-CCA security for anamorphic messages.

To provide an intuition, for Water's signatures the hash is computed as $H(m) = h_0 \prod_{i=1}^{n} h_i^{m_i}$, where the m_i is the i'th bit of message m. A signature is $(g^r, g^{\alpha\beta} H(m)^r)$ for random r, secret key $g^{\alpha\beta}$ and public key $(g^\alpha, g^\beta, h_0, \ldots, h_n)$. Also recall that in linear encryption the public key is $\mathsf{pk} = (u, v, h) \in \mathbb{G}^3$ and the secret key $(x, y) \in \mathbb{Z}_q$ such that $u^x = v^y = h$ holds. Encrypting a message $\mathsf{msg} \in \mathbb{G}$ amounts to choosing random κ_1, κ_2 and computing ciphertext $(c_1, c_2, c_3) = (u^{\kappa_1}, v^{\kappa_2}, h^{\kappa_1+\kappa_2} \cdot \mathsf{msg})$ which looks pseudorandom under the decision linear (DLIN) assumption and decryption computes $\mathsf{msg} = c_3 \cdot (c_1^x \cdot c_2^y)^{-1}$. When making the assumption that every user generates its own parameters for the Water's hash and thus knows a_i such that $h_i = g^{a_i}$ for $i \in \{0, \ldots, n\}$ and this is tolerated by the dictator[4], we can construct a simple PKAS. Using the knowledge of these discrete logarithms allows us to transmit an anamorphic message amsg by embedding one ciphertext component of a ciphertext (c_1, c_2, c_3) encrypting amsg into the first components of three Water's signatures.

2 Public Key Anamorphic Encryption

We now present the relevant notions for PKAE from the literature. We follow the current line of works on *receiver* anamorphic encryption introduced in [30], and subsequently generalized in [3,11,26]. Given our core motivation is to realize covert channels using an asymmetric variant, we follow the foundational work on PKAE in [31, 3].

At a high-level, a PKAE is a triplet $\Pi_{\mathsf{PKAE}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ associated with a regular PKE $\Pi_{\mathsf{PKE}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ which operates in either of the two modes: *regular* and *anamorphic*. The regular mode is basically the ordinary encryption scheme Π_{PKE} where the receiver generates a key-pair $(\mathsf{pk}, \mathsf{sk})$ using Gen, and hands over the secret key sk to the dictator, who can then decrypt any ciphertext ct meant for the receiver. In the anamorphic mode, the receiver runs aGen and generates anamorphic key-pair $(\mathsf{apk}, \mathsf{ask})$ along with encryption and decryption *double* keys $(\mathsf{dk}, \mathsf{tk})$. When a sender wants to send an anamorphic message amsg, she chooses an innocuous-looking message msg and runs aEnc, such that the output ciphertext act decrypts to amsg using tk on aDec, and msg using ask on Dec. Additionally, aDec might require to identify the existence of non-anamorphic messages. This is captured by a notion called *robustness* (cf.

[4] For the second construction based on BBS we do not require such an assumption

Appendix I) proposed in [3], which essentially requires aDec to return $\perp$ when decrypting a regular message msg encrypted with Enc, with overwhelming probability. *Anamorphic security* essentially requires that a PPT dictator should not be able to tell apart the output distributions of regular and anamorphic ciphertexts, along with the corresponding key pairs. $\mathsf{IND\text{-}CPA}$ *security* for PKAE is a security property with respect to the group of peers using the anamorphic channel. Here, an adversary given $(\mathsf{apk}, \mathsf{dk})$ must not be able to derive information about amsg from a given ciphertext. Below, we also define several (obvious) security notions beyond $\mathsf{IND\text{-}CPA}$.

Definition 1 (Public Key Anamorphic Encryption [11] [31]) *A public key anamorphic encryption (PKAE) scheme with anamorphic message space $\widehat{\mathcal{M}}$ associated with a public key encryption scheme $\Pi_{\mathsf{PKE}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with a regular message space $\mathcal{M}$ is a triplet $\Pi_{\mathsf{PKAE}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ of PPT algorithms defined as follows:*

$(\mathsf{apk}, \mathsf{ask}, \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGen}(1^\lambda)$: *The anamorphic key generation algorithm* aGen *takes as input the security parameter 1^λ, and outputs an anamorphic public key* apk*, an anamorphic secret key* ask*, an encryption double key* dk *and a decryption double key* tk*.*

$\mathsf{act} \leftarrow \mathsf{aEnc}(\mathsf{apk}, \mathsf{dk}, \mathsf{msg}, \mathsf{amsg})$: *The anamorphic encryption algorithm* aEnc *takes as input the anamorphic public key* apk*, the encryption double key* dk*, a regular message $\mathsf{msg} \in \mathcal{M}$ and an anamorphic message $\mathsf{amsg} \in \widehat{\mathcal{M}}$, and outputs an anamorphic ciphertext* act*.*

$\mathsf{amsg} \leftarrow \mathsf{aDec}(\mathsf{tk}, \mathsf{act})$: *The anamorphic decryption algorithm* aDec *takes as input the secret decryption double key* tk *and the anamorphic ciphertext* act*, and outputs an anamorphic message* amsg *or the symbol $\perp \notin \widehat{\mathcal{M}}$ indicating absence of anamorphic message.*

Correctness: We call Π_{PKAE} correct if for every pair of messages $(\mathsf{msg}, \mathsf{amsg}) \in \mathcal{M} \times \widehat{\mathcal{M}}$, it holds that

$$\Pr\left[\mathsf{amsg} \neq \mathsf{aDec}(\mathsf{tk}, \mathsf{act}) : \begin{array}{c}(\mathsf{apk}, \mathsf{ask}, \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGen}(1^\lambda) \\ \mathsf{act} \leftarrow \mathsf{aEnc}(\mathsf{apk}, \mathsf{dk}, \mathsf{msg}, \mathsf{amsg})\end{array}\right] \leq \mathsf{negl}(\lambda)$$

where the probability is taken over the random coins tosses of aGen and aEnc.

Anamorphic security: Π_{PKAE} satisfies anamorphic security if for every PPT dictator $\mathcal{D}$ it holds that

$$\left|\Pr[\mathsf{GReal}^{\mathcal{D}}_{\mathsf{PKE}}(\lambda) = 1] - \Pr[\mathsf{GAnam}^{\mathcal{D}}_{\mathsf{PKAE}}(\lambda) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security games are defined as follows:

$\underline{\mathsf{GReal}^{\mathcal{D}}_{\mathsf{PKE}}(\lambda):}$

1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$
2. **return** $\mathcal{D}^{\mathsf{OEnc}(\cdot,\cdot)}(\mathsf{pk}, \mathsf{sk})$ where $\mathsf{OEnc}(\mathsf{msg}, \mathsf{amsg}) = \mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$

$\underline{\mathsf{GAnam}^{\mathcal{D}}_{\mathsf{PKAE}}(\lambda):}$

1. $(\mathsf{apk}, \mathsf{ask}, \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGen}(1^\lambda)$
2. **return** $\mathcal{D}^{\mathsf{OaEnc}(\cdot,\cdot)}(\mathsf{apk}, \mathsf{ask})$ where $\mathsf{OaEnc}(\mathsf{msg}, \mathsf{amsg}) = \mathsf{aEnc}(\mathsf{apk}, \mathsf{dk}, \mathsf{msg}, \mathsf{amsg})$

IND-CPA security and beyond: Π_{PKAE} satisfies IND-CPA security if for every PPT adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, the following holds:

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

where the security game is defined as follows:

$\underline{\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\mathsf{PKAE}}(\lambda, \beta):}$

1. $(\mathsf{apk}, \mathsf{ask}, \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGen}(1^\lambda)$
2. $(\mathsf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1) \leftarrow \mathcal{A}(\mathsf{apk}, \mathsf{dk})$
3. $\mathsf{act}_\beta \leftarrow \mathsf{aEnc}(\mathsf{apk}, \mathsf{dk}, \mathsf{msg}, \mathsf{amsg}_\beta)$
4. **return** $\mathcal{A}(\mathsf{act}_\beta)$

Let us also consider several new but obvious security notions for Π_{PKAE}: We call Π_{PKAE} IND-CCA secure if the advantage of $\mathcal{A}$ in $\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\mathsf{PKAE}}$ is negligible when it is also given access to an $\mathsf{aDec}(\mathsf{tk}, \cdot)$-oracle which decrypts every given act except for the challenge act_β. To also guarantee IND-CCA security with respect to the dictator who knows ask, we additionally provide ask to $\mathcal{A}$ as input and call the corresponding security notion strong IND-CCA secure (sIND-CCA). Since, unfortunately, our constructions do not achieve sIND-CCA security, we define a relaxation in the spirit of Replayable CCA (RCCA) security for PKE as introduced in [10]: RCCA captures a security notion which is identical to CCA except for the added ability of the attacker to generate new ciphertexts that decrypt to the same plaintexts as previously seen ciphertexts. More precisely, our sIND-RCCA game coincides with the sIND-CCA game, except that the $\mathsf{aDec}(\mathsf{tk}, \cdot)$-oracle rejects for a given act if $\mathsf{aDec}(\mathsf{tk}, \mathsf{act})$ results in amsg_0 or amsg_1.

sIND-RCCA security: Π_{PKAE} satisfies sIND-RCCA security if for every PPT adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, the following holds:

$$\left|\Pr[\mathsf{GsIND\text{-}RCCA}^{\mathcal{A}}_{\mathsf{PKAE}}(\lambda, 0) = 1] - \Pr[\mathsf{GsIND\text{-}RCCA}^{\mathcal{A}}_{\mathsf{PKAE}}(\lambda, 1) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security game is defined as follows:

$\underline{\mathsf{GsIND\text{-}RCCA}^{\mathcal{A}}_{\mathsf{PKAE}}(\lambda,\beta):}$

1. $(\mathsf{apk},\mathsf{ask},\mathsf{dk},\mathsf{tk}) \leftarrow \mathsf{aGen}(1^\lambda)$
2. $(\mathsf{msg},\mathsf{amsg}_0,\mathsf{amsg}_1) \leftarrow \mathcal{A}^{\mathsf{OaDec}(\mathsf{tk},\cdot)}(\mathsf{apk},\mathsf{ask},\mathsf{dk})$
3. $\mathsf{act}_\beta \leftarrow \mathsf{aEnc}(\mathsf{apk},\mathsf{dk},\mathsf{msg},\mathsf{amsg}_\beta)$
4. **return** $\mathcal{A}^{\mathsf{OaDec}(\mathsf{tk},\cdot)}(\mathsf{act}_\beta)$
   where $\mathsf{OaDec}(\mathsf{tk},\mathsf{act}) = \mathsf{aDec}(\mathsf{tk},\mathsf{act})$ and
   $\mathsf{OaDec}(\mathsf{tk},\mathsf{act})$ outputs $\perp$ if $\mathsf{aDec}(\mathsf{tk},\mathsf{act}) = \mathsf{amsg}_0$ or amsg_1.

3 Public Key Anamorphic Encryption Extension

In this section, we present concrete realizations of PKAE based on the well-known ElGamal [17] and the Dual-Regev [22] cryptosystems as well as the Fujisaki-Okamoto transformation [19, 20].
Recall our previous discussion on embedding amsg within μ ciphertexts of the underlying PKE, which we now formally regard through the following definition.

Definition 2 (μ-Message Public Key Anamorphic Encryption Extension) *For $\mu \in \mathbb{N}$, a public key anamorphic encryption extension (μ-PKAEE) scheme with anamorphic message space $\widehat{\mathcal{M}}$ associated with a public key encryption scheme $\Pi_{\mathsf{PKE}} = (\mathsf{Gen},\mathsf{Enc},\mathsf{Dec})$ with a regular message space $\mathcal{M}$ is a triplet $\Pi_{\mathsf{PKAE}} = (\mathsf{aGen},\mathsf{aEnc},\mathsf{aDec})$ of PPT algorithms defined as follows:*

$\underline{(\mathsf{apk},\mathsf{ask},\mathsf{dk},\mathsf{tk}) \leftarrow \mathsf{aGen}(1^\lambda):}$ *The anamorphic key generation algorithm* aGen *takes as input the security parameter 1^λ, and outputs the anamorphic public key* apk*, anamorphic secret key* ask*, an encryption double key* dk *and a decryption double key* tk*.*

$\underline{\mathbf{act} \leftarrow \mathsf{aEnc}(\mathsf{apk},\mathsf{dk},\mathbf{msg},\mathsf{amsg}):}$ *The anamorphic encryption algorithm* aEnc *takes as input the anamorphic public key* apk*, the encryption double key* dk*, a regular message vector $\mathbf{msg} \in \mathcal{M}^\mu$, and an anamorphic message $\mathsf{amsg} \in \widehat{\mathcal{M}}$, and outputs an anamorphic ciphertext vector* $\mathbf{act}$*.*

$\underline{\mathsf{amsg} \leftarrow \mathsf{aDec}(\mathsf{tk},\mathbf{act}):}$ *The anamorphic decryption algorithm* aDec *takes as input the decryption double key* tk *and the anamorphic ciphertext* $\mathbf{act}$*, and outputs an anamorphic message* amsg *or the symbol $\perp \notin \hat{\mathcal{M}}$ indicating absence of an anamorphic message.*

We now introduce the PKAE-compatible notion for a PKE. At a high-level, the notion encompasses two properties. The first property *on-the-fly double key generation* captures the essence of anamorphic extension since the double keys dk and tk can simply be derived from regular key-pairs already in use. More precisely, for key pairs $(\mathsf{pk},\mathsf{sk})$ and $(\mathsf{pk}',\mathsf{sk}')$ generated by $\mathsf{PKE.Gen}$, we set sk' as tk, while dk is set as a function of the key pairs. The second property indicates that aDec is essentially the same as $\mathsf{PKE.Dec}$ under the existence of a publicly known poly-time function F. In particular, $F(\mathbf{act})$ produces a valid ciphertext that decrypts to amsg using $\mathsf{PKE.Dec}$. We call this property *ciphertext extractibility.*

Definition 3 (PKAE-Compatible) *A public key encryption scheme* $\Pi_{\mathsf{PKE}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be PKAE-compatible for the triplet* $\Pi_{\mathsf{PKAE}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ *if the following holds:*

- *On-the-fly double key generation: A part of output* $(\mathsf{dk}, \mathsf{tk})$ *of* $\mathsf{PKAE.aGen}(1^\lambda)$ *is the output of* $\mathsf{PKE.Gen}(1^\lambda)$*. In particular, if* $\mathsf{PKAE.aGen}(1^\lambda) = (\mathsf{ask}, \mathsf{apk}, \mathsf{dk}, \mathsf{tk})$ *and* $\mathsf{PKE.Gen}(1^\lambda) = (\mathsf{pk}, \mathsf{sk})$*,* $\mathsf{PKE.Gen}(1^\lambda) = (\mathsf{pk}', \mathsf{sk}')$ *then* $\mathsf{apk} = \mathsf{pk}, \mathsf{ask} = \mathsf{sk}, \mathsf{tk} = \mathsf{sk}'$ *and there exists an efficiently computable function* G *such that* $\mathsf{dk} = G(\mathsf{pk}, \mathsf{sk}, \mathsf{pk}', \mathsf{sk}')$*.*
- *Ciphertext extractibility: There exist an efficiently computable function* F *such that* $\mathsf{PKE.Dec}(\mathsf{tk}, F(\mathbf{act})) = \mathsf{amsg}$*, where* $\mathbf{act} = \mathsf{PKAE.aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})$*.*

Throughout the rest of this section, we present PKAE constructions for the ElGamal and Dual-Regev encryption schemes which follow this notion. The recently proposed simple public key anamorphic construction based on CS-Lite [11] for instance, however, does not satisfy the PKAE-compatible notion. We defer the discussion on robustness of our constructions to Appendix I.

3.1 Extension of the ElGamal Encryption Scheme

We now show a concrete PKAE construction based on the well-known ElGamal encryption scheme [17]. The ElGamal PKE scheme is a triplet of PPT algorithms $\Pi_{\mathsf{El}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ defined over a cyclic group $\mathbb{G} = \langle g \rangle$ of order q. The key generation algorithm $\mathsf{El.Gen}$ on input 1^λ, samples a uniformly random x in $\mathbb{Z}_q$, and returns $(\mathsf{sk} \coloneqq x, \mathsf{pk} \coloneqq g^x)$. To encrypt a message $\mathsf{msg} \in \mathcal{M}$, where $\mathcal{M}$ is the group $\mathbb{G}$ itself, the encryption algorithm $\mathsf{El.Enc}$ selects a uniformly random κ in $\mathbb{Z}_q$, and computes the ciphertext ct as $(\mathsf{ct}_1, \mathsf{ct}_2) = (g^\kappa, \mathsf{pk}^\kappa \cdot \mathsf{msg})$. The decryption algorithm $\mathsf{EL.Dec}$ simply recovers msg by computing $\mathsf{ct}_2 \cdot (\mathsf{ct}_1)^{-\mathsf{sk}}$. We can also consider ElGamal for $\mathcal{M}$ being a polynomially-bounded subspace of $\mathbb{Z}_q$, then msg can be recovered by computing the DL of $\mathsf{ct}_2 \cdot (\mathsf{ct}_1)^{-\mathsf{sk}}$. It is well-known that ElGamal is $\mathsf{IND\text{-}CPA}$ secure under the DDH assumption in $\mathbb{G}$.

Recall from our earlier discussions (Section 1.2), that our goal is to embed amsg within μ ciphertexts. With the ElGamal encryption scheme, we achieve this goal using $\mu = 2$ ciphertexts. More specifically, we design an ElGamal 2-message public key anamorphic encryption extension Π_{aEl} such that it is indistinguishable from an ElGamal 2-message encryption.

ElGamal 2-Message Encryption Scheme. The ElGamal 2-message encryption scheme $\Pi_{\mathsf{2El}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, associated with $\Pi_{\mathsf{El}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ does the obvious: Π_{2El} accepts a message vector consisting of two messages encrypts them separately using $\mathsf{El.Enc}$ and decrypts the corresponding ciphertext vector by applying $\mathsf{El.Dec}$ twice. Obviously, Π_{2El} is also $\mathsf{IND\text{-}CPA}$ secure under DDH.

ElGamal 2-Message PKAE. We begin with an interesting observation that the ciphertext $\mathbf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2)$ output by $\mathsf{2El.Enc}$ precisely contains two uniformly random group elements κ_1, κ_2 (first components of ct_1 and ct_2 respectively). If we could bias these random choices in a way that it covertly embeds amsg, then

we would be able to recover amsg by treating κ_1 and κ_2 as regular ElGamal ciphertext components. Additionally, we need to ensure that the biased ciphertext components are not "much" different from a random output of the regular ElGamal encryption scheme, as otherwise the dictator would be able to tell apart regular and anamorphic instances. We now give more details.

The ElGamal 2-message public key anamorphic encryption extension $\Pi_{\mathsf{aEl}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ works as follows: the key generation algorithm $\mathsf{aEl.aGen}$ sets $(\mathsf{apk}, \mathsf{ask}) = (\mathsf{pk}, \mathsf{sk})$ as in $\mathsf{El.Gen}$. In addition, for a uniformly random $\alpha \in \mathbb{Z}_q$, it sets the double keys as $\mathsf{tk} = \alpha$ and $\mathsf{dk} = (g^{\alpha}, g^{\alpha x})$. On input a regular message vector $\mathbf{msg} = (\mathsf{msg}_1, \mathsf{msg}_2) \in \mathcal{M}^2$, and an anamorphic message $\mathsf{amsg} \in \widehat{\mathcal{M}}$, where $\widehat{\mathcal{M}}$ is a polynomially bounded subspace of $\mathbb{Z}_q$, the encryption algorithm $\mathsf{aEl.aEnc}$ computes $\mathsf{ct}_1 = (r_1, c_1)$ using a uniformly random κ on msg_1, identical to $\mathsf{2El.Enc}$. However, for computing the second ciphertext ct_2, instead of choosing a uniformly random element κ_2 in $\mathbb{Z}_q$, the algorithm implicitly sets $\kappa_2 = \alpha \cdot \kappa + \mathsf{amsg}$ and obtains $r_2 := g^{\alpha\kappa + \mathsf{amsg}}$ and $c_2 := g^{x(\alpha\kappa + \mathsf{amsg})} \cdot \mathsf{msg}_2$. The decryption algorithm $\mathsf{aEl.aDec}$ recovers amsg by simply returning the discrete logarithm of $r_2 \cdot (r_1)^{-\mathsf{tk}}$, taking advantage of the polynomial message space $\widehat{\mathcal{M}}$. Recall that the dictator has no information on the key pair $(\mathsf{tk}, \mathsf{dk})$, and hence is oblivious to the presence of amsg. We present our construction in Figure 1.

$\underline{\mathsf{aGen}(1^\lambda):}$

1. $(\mathsf{sk} = x, \mathsf{pk} = y) \leftarrow \mathsf{El.Gen}(1^\lambda)$
2. $\alpha \leftarrow \mathbb{Z}_q$
3. $\mathsf{ask} := \mathsf{sk}, \mathsf{apk} := \mathsf{pk}$
4. $\mathsf{dk} := (g^{\alpha}, g^{\alpha x}), \mathsf{tk} := \alpha$
5. **return** $(\mathsf{ask}, \mathsf{apk}, \mathsf{dk}, \mathsf{tk})$

$\underline{\mathsf{aDec}(\mathsf{tk}, \mathbf{act}):}$

1. $g^{\mathsf{amsg}} := \mathsf{EL.Dec}(\mathsf{tk}, F(\mathbf{act}))$
2. **return** $\log g^{\mathsf{amsg}}$

$\underline{\mathsf{aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg}):}$

1. Parse $\mathsf{dk} = (\mathsf{dk}_1, \mathsf{dk}_2)$
2. Parse $\mathbf{msg} = (\mathsf{msg}_1, \mathsf{msg}_2)$
3. $\kappa \leftarrow \mathbb{Z}_q$
4. $\mathsf{act}_1 := \mathsf{El.Enc}(\mathsf{apk}, \mathsf{msg}_1; \kappa)$
   // $\mathsf{act}_1 = (r_1, c_1)$
   // $r_1 = g^{\kappa}, c_1 := \mathsf{apk}^{\kappa} \cdot \mathsf{msg}_1$
5. $\mathsf{act}_2 := \mathsf{El.Enc}(\mathsf{apk}, \mathsf{msg}_2; \kappa\alpha + \mathsf{amsg})$
   // $\mathsf{act}_2 = (r_2, c_2)$
   // $r_2 = \mathsf{dk}_1^{\kappa} \cdot g^{\mathsf{amsg}}$
   // $c_2 = \mathsf{dk}_2^{\kappa} \cdot \mathsf{apk}^{\mathsf{amsg}} \cdot \mathsf{msg}_2$
6. **return act** $:= (\mathsf{act}_1, \mathsf{act}_2)$

$\underline{F(\mathbf{act}):}$

1. Parse $\mathbf{act} = (\mathsf{act}_1, \mathsf{act}_2)$
2. Parse $\mathsf{act}_1 = (r_1, c_1)$, $\mathsf{act}_2 = (r_2, c_2)$
3. **return** (r_1, r_2)

Fig. 1: 2-PKAEE Π_{aEl}

Correctness, Compatibility and Security Analysis. We discuss correctness and the proofs of the following lemma and theorem related to compatibility, anamorphic security and IND-CPA security of Π_{aEl} to the Appendix B.

Lemma 1 *ElGamal is PKAE-compatible for* Π_{aEl} *described in Figure 1 associated with the ElGamal* 2*-message encryption scheme* Π_{2El}*.*

Theorem 1 *Under the* DDH *assumption for* $\mathbb{G}$*, the ElGamal* 2*-message public key anamorphic encryption extension* Π_{aEl} *described in Figure 1 satisfies anamorphic and* IND-CPA *security.*

3.2 Extension of the Dual-Regev Encryption Scheme

We now present a PKAE construction based on the Dual-Regev cryptosystem by Gentry, Peikert and Vaikuntanathan [22]. We focus on a simple version that encrypts single bits. The Dual-Regev encryption scheme is a triplet of PPT algorithms $\Pi_{\mathsf{DReg}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ defined over a security parameter λ as follows: let $n = n(\lambda)$, $q = q(\lambda)$ and $m = m(\lambda)$ be polynomials in λ. Let $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ denote the group of reals $[0, 1)$ with modulo 1 addition. Then for $\alpha \in \mathbb{R}^+$, let Ψ_α be the distribution on $\mathbb{T}$ of a normal variable with mean 0 and standard deviation $\alpha/\sqrt{2\pi}$. Following the notations of [22], we set $m \geq 2n \log q$, $r = \omega(\sqrt{\log m})$, $q \geq 5r(m+1)$, $\alpha \leq 1/(r\sqrt{m+1} \cdot \omega(\sqrt{\log n}))$ and the LWE noise distribution $\chi = \Psi_\alpha$. The key generation algorithm $\mathsf{DReg.Gen}$ randomly samples vector $\mathbf{k} \in \{0,1\}^m$ and a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. It then returns $\mathsf{sk} \coloneqq \mathbf{k}$ and $\mathsf{pk} \coloneqq (\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{k})$. On an input $\mathsf{msg} \in \mathcal{M}$, where $\mathcal{M} = \{0,1\}$, the encryption algorithm $\mathsf{DReg.Enc}$ selects a uniformly random vector $\mathbf{s} \in \mathbb{Z}_q^n$, a vector $\mathbf{e}^T$ sampled randomly from χ^m, and an integer y sampled from χ. It then computes the ciphertext ct as $(\mathsf{ct}_1, \mathsf{ct}_2) \coloneqq (\mathbf{s}^T \cdot \mathbf{A} + \mathbf{e}^T, \mathbf{s}^T \cdot \mathbf{b} + y + \mathsf{msg} \cdot \lfloor \frac{q}{2} \rfloor \bmod q)$. The decryption algorithm $\mathsf{DReg.Dec}$ is probabilistic, and outputs 0 if $|\mathsf{ct}_2 - \mathsf{ct}_1 \cdot \mathbf{k}| \leq \frac{q}{4}$, and 1 otherwise. Note that, $|\mathsf{ct}_2 - \mathsf{ct}_1 \cdot \mathbf{k}|$ simply reduces to $\mathsf{msg} \cdot \lfloor \frac{q}{2} \rfloor$ plus an error term $(y - \mathbf{e}^T \cdot \mathbf{k})$, whose norm is smaller than $\frac{q}{5}$ with an overwhelming probability in λ. Dual-Regev using parameters as described above is IND-CPA secure under the (n, q, χ)-DLWE assumption of dimension m. As in the case of ElGamal, we can similarly define Dual-Regev 2-message encryption scheme $\Pi_{\mathsf{2DReg}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ where encryption takes two messages as input.

Dual-Regev 2-Message PKAE. To design a PKAE associated with the Π_{2DReg} construction, we again replace the random choices made during encryption with specially chosen values. Intuitively, one might expect to find a solution by taking an approach identical to our ElGamal anamorphic extension (see Figure 1). Nevertheless, such a straightforward adaptation does not work, which we now account in more detail. Recall that in an ElGamal 2-message encryption scheme, the group elements $r_1 = g^{\kappa_1}$ and $r_2 = g^{\kappa_2}$ (first components of two ciphertexts ct_1 and ct_2) are uniformly random. In our ElGamal public key anamorphic 2-message extension, we set $\kappa_2 = \alpha\kappa_1 + \mathsf{amsg}$, where κ_1 is uniformly random in $\mathbb{Z}_q$, such that amsg could be recovered by treating (r_1, r_2) as an ElGamal ciphertext and executing the regular decryption algorithm $\mathsf{EL.Dec}$. This worked fine as an ElGamal ciphertext indeed contains two group elements. However, if we proceed in a similar manner for our Dual-Regev anamorphic construction, i.e., bias the randomness $\mathbf{s}_2$ in $\mathbf{r}_2 = \mathbf{s}_2^T \cdot \mathbf{A} + \mathbf{e}_2^T$ (first component of regular Dual-Regev ciphertext ct_2) such that amsg is encoded within, then we are not able to recover

amsg since $\mathbf{s}_2$ cannot be extracted from $\mathbf{r}_2$. One could aim for a better approach by encoding amsg in $\mathbf{s}_2^T \cdot \mathbf{A} + \mathbf{e}_2^T$ and treating $(\mathbf{r}_1, \mathbf{r}_2)$ as a regular Dual-Regev ciphertext. But this again fails as a Dual-Regev ciphertext comprises of a vector $\mathbf{r}$ and a scalar c. To get around this apparent constraint, we compute $\mathbf{s}_2$ such that the first element of the vector $\mathbf{r}_2$ is $\mathbf{s}_1^T \cdot \widehat{\mathbf{b}} + \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor$, which we then extract (call it "c") to recover the anamorphic message using a regular Dual-Regev decryption algorithm on $(\mathbf{r}_1, c)$. We now give more details.

The Dual-Regev 2-message public key anamorphic encryption extension $\Pi_{\mathsf{aDReg}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ works as follows: the key generation $\mathsf{aDReg.aGen}$ sets $(\mathsf{apk}, \mathsf{ask}) = (\mathsf{pk}, \mathsf{sk})$ output by $\mathsf{DReg.Gen}$. It then selects a secret vector $\widehat{\mathbf{k}}$, and sets $\mathsf{tk} := \widehat{\mathbf{k}}$, $\mathsf{dk} := (\mathbf{A}, \widehat{\mathbf{b}} = \mathbf{A} \cdot \widehat{\mathbf{k}})$. To encrypt a regular message vector $\mathbf{msg} = (\mathsf{msg}_1, \mathsf{msg}_2) \in \mathcal{M}^2$, along with an anamorphic message $\mathsf{amsg} \in \hat{\mathcal{M}}$, where $\mathcal{M} = \hat{\mathcal{M}} = \{0,1\}$, the anamorphic encryption algorithm $\mathsf{aDReg.aEnc}$ samples a uniformly random $\mathbf{s}_1$ in $\mathbb{Z}_q^n$, $\mathbf{e}_1^T, \mathbf{e}_2^T$ from χ^m and $\widehat{y}$ from χ. It then computes $\mathsf{act}_1 = (\mathbf{r}_1, c_1)$ as $\mathsf{DReg.Enc}(\mathsf{apk}, \mathsf{msg}_1; \mathbf{s}_1, \mathbf{e}_1^T)$. To encode amsg in $\mathbf{r}_2$ (first component of act_2), it first computes a non-zero solution to $\mathbf{a}_1 \cdot \mathbf{x} = \mathbf{s}_1^T \cdot \widehat{\mathbf{b}} + \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor - e_{2,1} \mod q$, where $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_n]^T$ and $\mathbf{e}_2^T = [e_{2,1}, \ldots, e_{2,m}]$, and sets $\mathbf{s}_2 = \mathbf{x}$. We know that such a solution exists (by the rank-nullity theorem). Then, $\mathsf{act}_2 = (\mathbf{r}_2, c_2)$ is computed as $\mathsf{DReg.Enc}(\mathsf{apk}, \mathsf{msg}_2; \mathbf{s}_2, \mathbf{e}_2^T)$. Finally, the anamorphic ciphertext $\mathbf{act}$ is $(\mathsf{act}_1, \mathsf{act}_2)$. Let $\mathsf{proj} : \mathbb{Z}_q^m \to \mathbb{Z}_q$ be a projection onto $\mathbb{Z}_q$. To recover amsg, the anamorphic decryption algorithm $\mathsf{aDReg.aDec}$ extracts the first element of $\mathbf{r}_2$ as $\widehat{c} = \mathsf{proj}_1(\mathbf{r}_2^T)$, followed by computing $\mathsf{DReg.Dec}$ on $\widehat{\mathbf{r}} = \mathbf{r}_1$ and $\widehat{c}$, using tk. We note that amsg is hidden from the dictator's view as it has no information on the double keys. See Fig. 2 for our construction.

Correctness, Compatibility and Security Analysis. We discuss correctness and the proofs of the following lemma and theorem related to compatibility, anamorphic security and IND-CPA security of Π_{aDReg} to the Appendix C.

Lemma 2 *Dual-Regev is PKAE-compatible for Π_{aDReg} described in Figure 2 associated with the Dual-Regev 2-message encryption scheme Π_{2DReg}.*

Theorem 2 *Let us consider $m \geq 2n \log q$, $r = \omega(\sqrt{\log m})$, $q \geq 5r(m+1)$, $\alpha \leq 1/(r\sqrt{m+1} \cdot \omega(\sqrt{\log n}))$ and $\chi = \Psi_\alpha$. Then, under the (n, q, χ)-DLWE assumption of dimension m, the Dual-Regev 2-message anamorphic encryption extension Π_{aDReg} described in Fig. 2 satisfies anamorphic and IND-CPA security.*

3.3 Extension of FO-Based PKE

The well-known Fujusaki-Okamoto (FO) transformation [19,20] allows to turn a weakly secure PKE and SKE into a PKE that is IND-CCA secure in the Random Oracle Model. Let $\Pi_{\mathsf{PKE}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ and $\Pi_{\mathsf{SKE}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be the underlying encryption schemes. Then the FO transformation converts them into a scheme $\Pi_{\mathsf{FO}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ as follows: $\mathsf{FO.Gen}(1^\lambda)$ simply executes $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$. $\mathsf{FO.Enc}(\mathsf{pk}, \mathsf{msg})$ returns $c := (c_1, c_2) := (\mathsf{PKE.Enc}(\mathsf{pk}, r; H(r, \mathsf{msg})), \mathsf{SKE.Enc}(G(r), \mathsf{msg}))$, where $r \leftarrow \mathcal{M}_{\mathsf{PKE}}$ is randomly chosen from the message

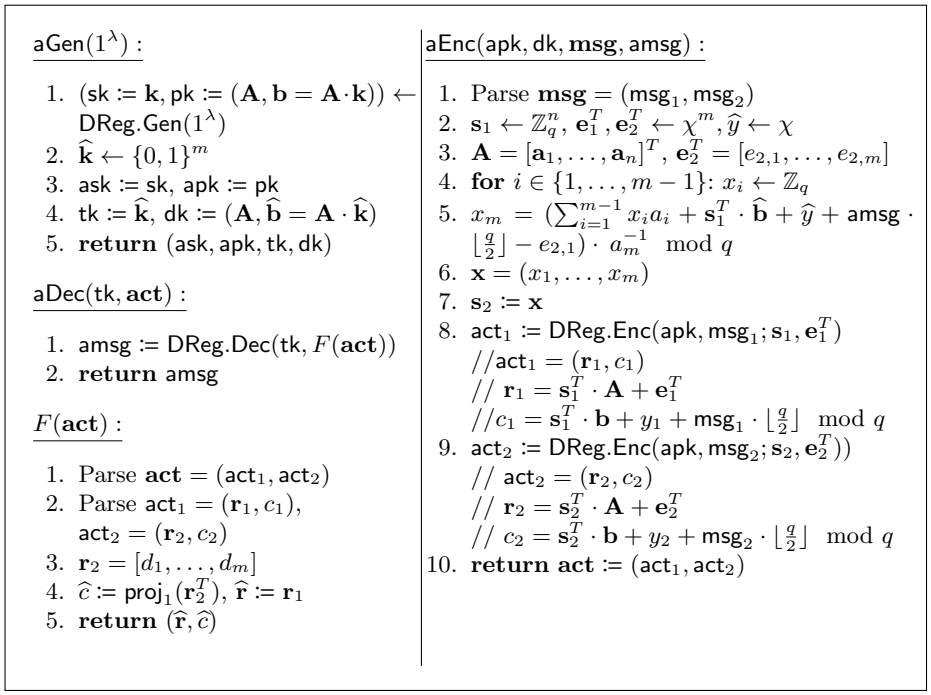

Fig. 2: 2-PKAEE Π_{aDReg}

space of PKE, and H and G are hash functions (modelled as random oracles) mapping to the space of random coins $\mathcal{R}_{\mathsf{PKE}}$ of PKE and the key space $\mathcal{K}_{\mathsf{SKE}}$ of SKE, respectively. To decrypt, $\mathsf{FO.Dec}(\mathsf{sk}, c)$ computes $r := \mathsf{PKE.Dec}(\mathsf{sk}, c_1)$ and returns $\mathsf{SKE.Dec}(G(r), c_2)$ if $c_1 = \mathsf{PKE.Enc}(\mathsf{pk}, r; H(r, \mathsf{msg}))$.

The idea to turn Π_{FO} into an anamorphic public key encryption scheme is simple: The FO transformation provides us with the recoverable randomness $r \leftarrow \mathcal{M}_{\mathsf{PKE}}$ which we can use to embed the components of a Π_{FO} ciphertext for amsg. For this purpose, the ciphertext components need to be pseudo-random. More precisely, we require IND$-CPA security for Π_{PKE} and wIND$-CPA security for Π_{SKE}, both saying that its hard for an adversary to distinguish the encryption of a chosen message from a random ciphertext. In the wIND$-CPA game, no encryption oracle is given. For both definitions please refer to Appendix A.5. Besides ciphertexts being pseudo-random, they also need to be compatible with $\mathcal{M}_{\mathsf{PKE}}$. To ensure the latter, we require the existence of a pair of PPT algorithms (Encode, Decode) that allows us to encode ciphertexts from $\mathcal{C}_{\mathsf{FO}} = \mathcal{C}_{\mathsf{PKE}} \times \mathcal{C}_{\mathsf{SKE}}$ to plaintext vectors from $\mathcal{M}_{\mathsf{PKE}}^{\mu}$ for some small $\mu \in \mathbb{N}$ (cf. Definition 4). The resulting μ-message anamorphic extension Π_{FO} is given in Definition 3.

Definition 4 (Pseudo-Random Encodings) *Let* $\mathsf{Setup}(1^\lambda)$ *be a PPT algorithm that outputs (polynomial-size descriptions of) two sets* $\mathbb{X}$ *and* $\mathbb{Y}$ *which both*

allow for efficient uniform sampling. Let $(\mathsf{Encode}, \mathsf{Decode})$ *be a pair of PPT algorithms such that* Encode *takes an element* $x \in \mathbb{X}$ *as input and outputs an element* $y \in \mathbb{Y}$ *or* $\perp$ *and* Decode *takes an element* $y \in \mathbb{Y}$ *as input and outputs an element* $x \in \mathbb{X}$ *or* $\perp$*. We call this pair a pseudo-random* $(\mathbb{X}, \mathbb{Y})$*-encoding associated with* Setup*, if it satisfies the following properties:*

1. *Correctness: For any* $(\mathbb{X}, \mathbb{Y}) \leftarrow \mathsf{Setup}(1^\lambda)$*, it holds that*

$$\Pr\left[x \neq x' : \ x \leftarrow \mathbb{X},\ y \leftarrow \mathsf{Encode}(x),\ x' \leftarrow \mathsf{Decode}(y)\right] \leq \mathsf{negl}(\lambda)$$

2. *Pseudo-randomness: For any* $(\mathbb{X}, \mathbb{Y}) \leftarrow \mathsf{Setup}(1^\lambda)$*, and any PPT algorithm* $\mathcal{A}$ *holds that*

$$\Pr\left[b = b' : \begin{array}{c} x \leftarrow \mathbb{X},\ y_0 \leftarrow \mathsf{Encode}(x),\ y_1 \leftarrow \mathbb{Y}, \\ b \leftarrow \{0,1\},\ b' \leftarrow \mathcal{A}(\mathbb{X}, \mathbb{Y}, y_b) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

$\underline{\mathsf{aGen}(1^\lambda):}$

1. $(\mathsf{ask}, \mathsf{apk}) \leftarrow \mathsf{FO.Gen}(1^\lambda)$
2. $(\mathsf{tk}', \mathsf{dk}) \leftarrow \mathsf{FO.Gen}(1^\lambda)$
3. $\mathsf{tk} \coloneqq (\mathsf{ask}, \mathsf{tk}')$
4. **return** $(\mathsf{ask}, \mathsf{apk}, \mathsf{dk}, \mathsf{tk})$

$\underline{\mathsf{aDec}(\mathsf{tk}, \mathbf{act}):}$

1. **for** $j \in \{1, \ldots, \mu\}$:
   Parse $\mathsf{act}_j = (c_1, c_2)$
   $r_j \coloneqq \mathsf{PKE.Dec}(\mathsf{ask}, c_1)$
   if $r_j = \perp$ **return** $\perp$
2. $c \coloneqq \mathsf{Decode}(r_1, \ldots, r_\mu)$
3. **if** $c = \perp$ **return** $\perp$
4. **return** $\mathsf{FO.Dec}(\mathsf{tk}', c)$

$\underline{\mathsf{aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg}):}$

1. Parse $\mathbf{msg} = (\mathsf{msg}_1, \ldots, \mathsf{msg}_\mu)$
2. $c \leftarrow \mathsf{FO.Enc}(\mathsf{dk}, \mathsf{amsg})$
3. $(r_1, \ldots, r_\mu) \leftarrow \mathsf{Encode}(c)$
   // In the negligible event that
   // $\mathsf{Encode}(c) = \perp$, we return to 2.
4. **for** $j \in \{1, \ldots, \mu\}$:
   $\mathsf{act}_j \coloneqq \mathsf{FO.Enc}(\mathsf{apk}, \mathsf{msg}_j; r_j)$
5. **return act** $\coloneqq (\mathsf{act}_1, \ldots, \mathsf{act}_\mu)$

Fig. 3: μ-PKAEE Π_{FOAE}

It is not hard to see that Π_{FO} satisfies anamorphic security as the embedded ciphertexts components containing amsg are indistinguishable from a regular $r \leftarrow \mathcal{M}_{\mathsf{PKE}}$. Moreover, the IND-CCA security of Π_{FO} for encrypting amsg (i.e., the "inner" layer encryption) implies sIND-RCCA for the overall construction Π_{FOAE}. Roughly speaking, that means that an adversary (including the dictator) is not able to maul amsg but it may tamper with the "inner" and "outer" layer ciphertexts. Note that Π_{FOAE} does not satisfy IND-CCA security as aDec does not check the integrity for the "outer" layer ciphertexts act_j. Hence, if the underlying Π_{PKE} is homomorphic, components of act_j could be re-randomized. We prove the following theorem in Appendix D.

Theorem 3 *Let* Π_{FO} *be an* IND-CCA *secure FO-based PKE where the underlying PKE* Π_{PKE} *is* IND$-CPA *secure and has an exponentially large message space* $\mathcal{M}_{\mathsf{PKE}}$ *(i.e.,* $|\mathcal{M}_{\mathsf{PKE}}| \geq 2^{\mathsf{poly}(\lambda)}$ *for non-constant* poly*), and the underlying SKE* Π_{SKE} *is* wIND$-CPA *secure. Furthermore, let* (Encode, Decode) *be a pseudo-random* $(\mathcal{C}_{\mathsf{FO}}, \mathcal{M}^{\mu}_{\mathsf{PKE}})$*-encoding, where* $\mathcal{C}_{\mathsf{FO}}$ *denotes the ciphertext space of* Π_{FO}*. Then* Π_{FOAE} *as defined in Figure 3 is a* μ*-message anamorphic extension satisfying anamorphic security and* sIND-RCCA *security in the ROM.*

An exemplary instantiation: FO of Hashed ElGamal and XOR. Let us consider an instantiation of Π_{FO} based on hashed ElGamal as PKE and XOR for the SKE part. Hashed ElGamal satisfies IND$-CPA security under DDH in the ROM and XORing a message with a random key of the same length satisfies wIND$-CPA security. In this case, FO.Enc(pk, msg) would produce a ciphertext $c = (g^{H(r,m)}, H'(\mathsf{pk}^{H(r,m)}) \oplus r, G(r) \oplus m)$, where $H : \{0,1\}^* \times \{0,1\}^* \to \mathbb{Z}_{|\mathbb{G}|}$, $H' : \mathbb{G} \to \{0,1\}^{\ell_1}$, $G : \{0,1\}^* \to \{0,1\}^{\ell_2}$, and $\mathbb{G}$ is the group over which ElGamal is defined. To obtain Π_{FOAE}, we also need instantiate an appropriate pseudo-random encoding. For this, certain parameters need to be fixed: Let us assume $\mathbb{G} = \mathbb{E}(\mathbb{F}_p)$ is an elliptic curve group of prime order q, where p is an λ-bit prime. Furthermore, let us set $\ell_1 = \ell_2 = \lambda$ (but also different parameter settingss like $\ell_1 = \frac{\lambda}{c_1}$ and $\frac{\lambda}{c_1 c_2}$ for $c_1, c_2 \in \mathbb{N}$ s.t. $\ell_1, \ell_2 \in \mathbb{N}$ are possible). In this case, we have to encode random elements of $\mathcal{C}_{\mathsf{FO}} = \mathbb{E}(\mathbb{F}_p) \times \{0,1\}^{\lambda} \times \{0,1\}^{\lambda}$ as random elements of $\mathcal{M}^{\mu}_{\mathsf{PKE}} = (\{0,1\}^{\lambda})^{\mu}$ for some $\mu \in \mathbb{N}$. Obviously, the only difficulty here is encoding the first component, i.e., a random point of $\mathbb{E}(\mathbb{F}_p)$ by one or more random λ-bit strings. For this purpose, we can deploy Elligator Squared [35]. This is a suitable encoding for essentially arbitrary elliptic curve groups $\mathbb{E}(\mathbb{F}_p)$ (also pairing-friendly) to $\mathbb{F}_p^2$. Encodings of random curve points can be efficiently computed and are statistically indistinguishable from uniformly random elements of $\mathbb{F}_p^2$. Moreover, [35] describes how encodings can be turned into random bit strings. For instance, if p is very close to 2^{λ} (which is often the case for standardized curves), and we use the basic λ-bit representation for elements of $\mathbb{F}_p$ as integers from $\{0, \ldots, p-1\}$, then we can simply interpret the output of the encoding as (statistically close to) uniform bitstrings from $\{0,1\}^{\lambda} \times \{0,1\}^{\lambda}$. In this way, we can obtain a pseudo-random encoding of $\mathbb{E}(\mathbb{F}_p) \times \{0,1\}^{\lambda} \times \{0,1\}^{\lambda}$ elements as $(\{0,1\}^{\lambda})^4$ elements.

4 Identity-Based Anamorphic Encryption Extension

In this section, we formally regard our new concept of reformulating public key anamorphic encryption through the lens of IBE. We further show how to realize this notion using the Boneh-Franklin IBE in Section 4.1. Appendix F shows how standardized IBE schemes based on the FO transformation can be turned into anamorphic schemes.

At a high-level, an IBAE is a quadruplet of PPT algorithms $\Pi_{\mathsf{IBAE}} =$ (aSetup, aKeyGen, aEnc, aDec) associated with a regular IBE $\Pi_{\mathsf{IBE}} =$ (Setup, KeyGen, Enc,

Dec) which could be deployed in either of the two modes : *regular* and *anamorphic*. In the regular mode, a dictator computes the master key-pairs $(\mathsf{mpk}, \mathsf{msk})$ using the Setup algorithm, and publishes the master public key mpk. It then derives user-specific private keys $\mathsf{sk}_{\mathsf{id}}$ based on the public identity id of users, using the master secret key msk on the KeyGen algorithm. A message msg can be encrypted with respect to id of a receiver using mpk on the Enc algorithm, which can eventually be decrypted using the respective private key $\mathsf{sk}_{\mathsf{id}}$ on Dec algorithm. As one can clearly see the dictator along with the receiver can recover msg since it generates the user-specific private keys. In the anamorphic mode, while the dictator generates the anamorphic master key-pairs $(\mathsf{ampk}, \mathsf{amsk})$ and derives the anamorphic private keys $\mathsf{ask}_{\mathsf{id}}$, the anamorphic clan lead generates the double master key-pairs $(\mathsf{dmpk}, \mathsf{dmsk})$ and the double private key $\mathsf{dsk}_{\mathsf{id}}$ of a user with identity id who is also an anamorphic clan member. To encrypt a regular message msg along with an anamorphic message amsg with respect to id, sender runs aEnc such that the output ciphertext decrypts to amsg using $\mathsf{dsk}_{\mathsf{id}}$ on aDec and msg using $\mathsf{ask}_{\mathsf{id}}$ on Dec. *Anamorphic security* essentially requires that the dictator should not be able to tell apart the output distributions of regular and anamorphic ciphertexts, along with the corresponding private key pairs. We now proceed more formally.

Definition 5 (μ-Message Identity-Based Anamorphic Encryption Extension) *For $\mu \in \mathbb{N}$, a μ-message anamorphic identity-based encryption extension (μ-IBAEE) with an anamorphic message space $\widehat{\mathcal{M}}$ associated with an identity-based encryption scheme $\Pi_{\mathsf{IBE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with a regular message space $\mathcal{M}$ is a quadruplet of PPT algorithms $\Pi_{\mathsf{IBAE}} = (\mathsf{aSetup}, \mathsf{aKeyGen}, \mathsf{aEnc}, \mathsf{aDec})$ defined as follows:*

$(\mathsf{ampk}, \mathsf{amsk}, \mathsf{dmpk}, \mathsf{dmsk}) \leftarrow \mathsf{aSetup}(1^\lambda)$: *The anamorphic key setup algorithm* aSetup *takes as input the security parameter 1^λ, and outputs an anamorphic master public key* ampk*, an anamorphic master secret key* amsk*, a double master public key* dmpk *and a double master secret key* dmsk*.*

$\mathsf{dsk}_{\mathsf{id}} \leftarrow \mathsf{aKeyGen}(\mathsf{dmsk}, \mathsf{id})$: *The anamorphic key generation algorithm* aKeyGen *takes as input the double master secret key* dmsk *and an arbitrary $\mathsf{id} \in \mathcal{ID}$, and outputs an double private key $\mathsf{dsk}_{\mathsf{id}}$ associated with* id*.*

$\mathbf{act} \leftarrow \mathsf{aEnc}(\mathsf{ampk}, \mathsf{dmpk}, \mathsf{id}, \mathbf{msg}, \mathsf{amsg})$: *The anamorphic encryption algorithm* aEnc *takes as input the anamorphic master public key* ampk*, double master public key* dmpk*, an identity* id*, a regular message vector $\mathbf{msg} \in \mathcal{M}^\mu$, and an anamorphic message $\mathsf{amsg} \in \widehat{\mathcal{M}}$, and outputs an anamorphic ciphertext vector* $\mathbf{act}$*.*

$\mathsf{amsg} \leftarrow \mathsf{aDec}(\mathsf{ask}_{\mathsf{id}}, \mathsf{dsk}_{\mathsf{id}}, \mathbf{act})$: *The anamorphic decryption algorithm* aDec *takes as input an anamorphic private key $\mathsf{ask}_{\mathsf{id}}$, a double private key $\mathsf{dsk}_{\mathsf{id}}$ and the anamorphic ciphertext vector* $\mathbf{act}$*, and outputs an anamorphic message* amsg *or the symbol $\perp$.*

Furthermore, the algorithms must satisfy the following properties.

Correctness: *For every pair of messages* $(\mathbf{msg}, \mathsf{amsg}) \in \mathcal{M}^{\mu} \times \widehat{\mathcal{M}}$*, and for every identity* $\mathsf{id} \in \mathcal{ID}$*, the following holds:*

$$\Pr\left[\mathsf{amsg} = \mathsf{aDec}(\mathsf{ask}_{\mathsf{id}}, \mathsf{dsk}_{\mathsf{id}}, \mathbf{act}) : \begin{array}{c}(\mathsf{ampk}, \mathsf{amsk}, \mathsf{dmpk}, \mathsf{dmsk}) \leftarrow \mathsf{aSetup}(1^{\lambda}) \\ \mathsf{ask}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(\mathsf{amsk}, \mathsf{id}) \\ \mathsf{dsk}_{\mathsf{id}} \leftarrow \mathsf{aKeyGen}(\mathsf{dmsk}, \mathsf{id}) \\ \mathbf{act} \leftarrow \mathsf{aEnc}(\mathsf{ampk}, \mathsf{dmpk}, \mathsf{id}, \mathbf{msg}, \mathsf{amsg})\end{array}\right] = 1$$

where probability is taken over the random coin tosses of aSetup *and* aEnc*.*

Anamorphic security: *For every PPT dictator* $\mathcal{D}$ *it holds that*

$$\left|\Pr[\mathsf{GReal}^{\mathcal{D}}_{\mathsf{IBE}}(\lambda) = 1] - \Pr[\mathsf{GAnam}^{\mathcal{D}}_{\mathsf{IBAE}}(\lambda) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security games are defined as follows:

$\underline{\mathsf{GReal}^{\mathcal{D}}_{\mathsf{IBE}}(\lambda):}$

1. $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^{\lambda})$
2. ***return*** $\mathcal{D}^{\mathsf{OEnc}(\cdot,\cdot,\cdot)}(\mathsf{mpk}, \mathsf{msk})$
   where $\mathsf{OEnc}(\mathsf{id}, \mathbf{msg}, \mathsf{amsg}) = \mathbf{ct}$*,*
   $\mathsf{ct}_i = \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, \mathsf{msg}_i)$*,*
   $\mathbf{msg} = \{\mathsf{msg}_i\}_{i\in[\mu]}, \mathbf{ct} = \{\mathsf{ct}_i\}_{i\in[\mu]}$

$\underline{\mathsf{GAnam}^{\mathcal{D}}_{\mathsf{IBAE}}(\lambda):}$

1. $(\mathsf{ampk}, \mathsf{amsk}, \mathsf{dmpk}, \mathsf{dmsk}) \leftarrow \mathsf{aSetup}(1^{\lambda})$
2. ***return*** $\mathcal{D}^{\mathsf{OaEnc}(\cdot,\cdot,\cdot)}(\mathsf{ampk}, \mathsf{amsk})$
   where
   $\mathsf{OaEnc}(\mathsf{id}, \mathbf{msg}, \mathsf{amsg})$
   $= \mathsf{aEnc}(\mathsf{ampk}, \mathsf{dmpk}, \mathsf{id}, \mathbf{msg}, \mathsf{amsg})$

We now formally regard the IBAE-compatible notion for an IBE. Building upon our previous insight, the notion captures *two* properties: *on-the-fly double key generation* property simply points out that the double master key pairs $(\mathsf{dmpk}, \mathsf{dmsk})$ can simply be derived from the regular IBE master key pairs $(\mathsf{mpk}, \mathsf{msk})$, while the *ciphertext extractibility* property invariably indicates that IBAE.aDec is essentially the same as IBE.Dec under the existence of an efficiently computable function F. More explicitly, $F(\mathsf{ask}_{\mathsf{id}}, \mathbf{act})$ produces a valid ciphertext that extracts to amsg using $\mathsf{dsk}_{\mathsf{id}}$ on IBE.Dec.

Definition 6 (IBAE-Compatible) *An identity-based encryption scheme* $\Pi_{\mathsf{IBE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be IBAE-compatible for the quadruplet* $\Pi_{\mathsf{IBAE}} = (\mathsf{aSetup}, \mathsf{aKeyGen}, \mathsf{aEnc}, \mathsf{aDec})$ *if the following holds:*

- *On-the-fly double key generation: A part of output* $(\mathsf{dmpk}, \mathsf{dmsk})$ *of* IBAE.$\mathsf{aSetup}(1^{\lambda})$ *is the output of* $\mathsf{IBE.Setup}(1^{\lambda})$*. In particular, if* $\mathsf{IBAE.aSetup}(1^{\lambda}) = (\mathsf{ampk}, \mathsf{amsk}, \mathsf{dmpk}, \mathsf{dmsk})$ *and* $\mathsf{IBE.Setup}(1^{\lambda}) = (\mathsf{mpk}, \mathsf{msk})$*,* $\mathsf{IBE.Setup}(1^{\lambda}) = (\mathsf{mpk}', \mathsf{msk}')$ *then* $\mathsf{ampk} = \mathsf{mpk}, \mathsf{amsk} = \mathsf{msk}, \mathsf{dmpk} = \mathsf{mpk}', \mathsf{dmsk} = \mathsf{msk}'$ *and there exists an efficiently computable function* G *such that* $\mathsf{dmpk} = G(\mathsf{mpk}, \mathsf{msk}, \mathsf{mpk}', \mathsf{msk}')$*.*
- *Ciphertext extractibility: There exists an efficiently computable function* F *such that* $\mathsf{IBE.Dec}(\mathsf{dsk}_{\mathsf{id}}, F(\mathsf{ask}_{\mathsf{id}}, \mathbf{act})) = \mathsf{amsg}$*, where* $\mathbf{act} = \mathsf{IBAE.aEnc}(\mathsf{ampk}, \mathsf{dmpk}, \mathsf{id}, \mathbf{msg}, \mathsf{amsg})$ *for an* $\mathsf{id} \in \mathcal{ID}$*.*

4.1 Extension of the Boneh-Franklin BasicIdent Scheme

We now present an IBAE construction based on the seminal Boneh-Franklin basic IBE scheme [8]. We focus on a simple version that encrypts single bits. The $\mathsf{BasicIdent}$ identity-based encryption scheme [8] is a quadruplet of PPT algorithms $\Pi_{\mathsf{BF}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ defined over a security parameter $\lambda \in \mathbb{N}$ as follows: let prime $q > 2^\lambda$ be a polynomial in λ. The setup algorithm Setup on input 1^λ samples groups $\mathbb{G}$ and $\mathbb{G}_T$ of order q and a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, along with an arbitrary group generator $P \in \mathbb{G}$. Define $\mathsf{H}_1 : \{0,1\}^* \to \mathbb{G}^*$, $\mathsf{H}_2 : \mathbb{G}_T \to \{0,1\}$. Then for a uniformly random s in $\mathbb{Z}_q^*$, compute $P_{\mathsf{pub}} \coloneqq P^s$. The algorithm finally returns a master secret key $\mathsf{msk} \coloneqq s$ and a master public key $\mathsf{mpk} \coloneqq (\mathbb{G}, \mathbb{G}_T, P, e, q, P_{\mathsf{pub}}, \mathsf{H}_1, \mathsf{H}_2)$. The key generation algorithm KeyGen on input $\mathsf{id} \in \{0,1\}^*$ and msk, computes $Q_{\mathsf{id}} \coloneqq \mathsf{H}_1(\mathsf{id})$, and returns a private key $\mathsf{sk}_{id} \coloneqq Q_{\mathsf{id}}^s$ corresponding to id. To encrypt a message $\mathsf{msg} \in \mathcal{M}$ towards a public identity id, where $\mathcal{M} = \{0,1\}$, the encryption algorithm Enc samples a uniformly random κ in $\mathbb{Z}_q^*$ and returns ciphertext ct as $(\mathsf{ct}_1, \mathsf{ct}_2) = (P^\kappa, \mathsf{msg} \oplus \mathsf{H}_2(g_{\mathsf{id}}^\kappa))$, where $g_{\mathsf{id}} = e(Q_{\mathsf{id}}, P_{\mathsf{pub}})$. The decryption algorithm Dec simply recovers msg by computing $\mathsf{ct}_2 \oplus \mathsf{H}_2(e(\mathsf{sk}_{\mathsf{id}}, \mathsf{ct}_1))$. The $\mathsf{BasicIdent}$ scheme is known to be $\mathsf{IND\text{-}CPA}$ secure under the DBDH assumption for $\mathbb{G}$. Following our prior formalization, we can similarly define the Boneh-Franklin 2-message IBE scheme $\Pi_{\mathsf{2BF}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ which encrypts two messages at once.

Boneh-Franklin 2-Message IBAE. To design a 2-IBAE associated with Π_{2BF}, we again bias the random choices made during encryption. Nevertheless, we note that the adaptation is not much identical to our PKAE constructions, and involves some subtleties. Elaborating on this, in our anamorphic extension of the ElGamal encryption scheme, we embed amsg within two regular ciphertexts $(\mathsf{ct}_1, \mathsf{ct}_2)$ by sampling their first components $r_1 = g^{\kappa_1}$ and $r_2 = g^{\kappa_2}$ such that $\kappa_2 \coloneqq \alpha \cdot \kappa_1 + \mathsf{amsg}$, where κ_1 is a uniformly random element of $\mathbb{Z}_q$. We then recover amsg by treating (r_1, r_2) as an ElGamal ciphertext, which works because r_2 is explicitly available in ct_2. However, embedding amsg within two regular Boneh-Franklin ciphertexts is not that straightforward. Observe that, in Boneh-Franklin ciphertext $\mathsf{ct}_2 = (r_2 = P^{\kappa_2}, c_2 = \mathsf{msg} \oplus \mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2}))$, the group element $g_{\mathsf{id}}^{\kappa_2}$ is not explicitly available. Rather, it only appears as an input to the hash function H_2, which is modeled as a random oracle in the security proof. Therefore, if we set κ_2 following the ElGamal encryption extension, then it seems impossible to recover the covert amsg due to the one-wayness of H_2. At this point, the only solution appears to be encoding amsg outside the hash function H_2. Then, a natural idea would be to apply rejection sampling technique such that amsg could be extracted from c_2 at the time of decryption. More specifically, we sample κ_2 subject to $\mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2}) = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1})$; call it $\widehat{c}$. We then treat $(r_1 = P^{\kappa_1}, \widehat{c} = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1}))$ as a Boneh-Franklin ciphertext and apply the regular decryption algorithm $\mathsf{BF.Dec}$ to recover amsg. While we present a 2-IBAE construction that supports anamorphic messages of single bits for the simplicity of exposition, nevertheless we remark that one could extend our construction to convey at most $\mathrm{O}(\log \lambda)$ bit-long anamorphic messages to ensure that encryption remains poly-

nomial. We now present more details. The Boneh-Franklin 2-IBAE extension $\Pi_{\mathsf{aBF}} = (\mathsf{aSetup}, \mathsf{aKeyGen}, \mathsf{aEnc}, \mathsf{aDec})$ works as follows: the anamorphic key generation algorithm $\mathsf{aBF.aSetup}$ initially sets $(\mathsf{ampk}, \mathsf{amsk}) = (\mathsf{mpk}, \mathsf{msk})$ as in the regular $\mathsf{BF.Setup}$, following which it samples a uniformly random $\alpha \in \mathbb{Z}_q^*$, and sets the double master secret key $\mathsf{dmsk} \coloneqq \alpha$ and the double master public key $\mathsf{dmpk} \coloneqq P^\alpha$. The anamorphic key generation algorithm $\mathsf{aBF.aKeyGen}$ takes as input dmsk and the user identity id and generates the double private key $\mathsf{dsk}_{\mathsf{id}}$ associated with id. On input a regular message vector $\mathbf{msg} = (\mathsf{msg}_1, \mathsf{msg}_2) \in \mathcal{M}^2$, and an anamorphic message $\mathsf{amsg} \in \widehat{\mathcal{M}}$, where $\mathcal{M} = \widehat{\mathcal{M}} = \{0,1\}$, the anamorphic encryption algorithm $\mathsf{aBF.aEnc}$ computes $\mathsf{act}_1 = (r_1, c_1)$ using a uniformly random κ_1 on msg_1 as $\mathsf{BF.Enc}(\mathsf{ampk}, \mathsf{msg}_1, \mathsf{id}; \kappa_1)$. However, for encoding amsg in c_2 (second component of act_2), the algorithm chooses κ_2 subject to the constraint $\mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2}) = \mathsf{amsg} \oplus \mathsf{H}_2(e(Q_{\mathsf{id}}, \mathsf{ampk})^{\kappa_1})$ and computes $\mathsf{act}_2 = (r_2, c_2)$ as $\mathsf{BF.Enc}(\mathsf{ampk}, \mathsf{msg}_2, \mathsf{id}; \kappa_2)$. Finally, the anamorphic ciphertext vector is $\mathbf{act} = (\mathsf{act}_1, \mathsf{act}_2)$. To recover amsg, the anamorphic decryption algorithm $\mathsf{aBF.aDec}$ first extracts $\mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2})$ by computing $\mathsf{H}_2(e(\mathsf{ask}_{\mathsf{id}}, r_2))$. Note that $\mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2})$ is set to be equal to $\mathsf{amsg} \oplus \mathsf{H}_2(e(Q_{\mathsf{id}}, \mathsf{ampk})^{\kappa_1})$, call it $\widehat{c}$. Then it computes $\mathsf{BF.Dec}$ on $\widehat{r} = r_1$ and $\widehat{c}$. Recall that the dictator has no information on the double private key $\mathsf{dsk}_{\mathsf{id}}$ associated with id, and hence is oblivious to the presence of amsg. We present our construction in Figure 4.

Correctness, Compatibility and Security Analysis. We discuss correctness and the proofs of the following lemma and theorem related to compatibility, anamorphic security and IND-CPA security of Π_{aBF} in Appendix E.

Lemma 3 *Boneh-Franklin is IBAE-compatible for Π_{aBF} described in Figure 4 associated with the Boneh-Franklin 2-message encryption scheme Π_{2BF}.*

Theorem 4 *Under the DBDH assumption for $(\mathbb{G}, \mathbb{G}_T, P, e, q)$, the Boneh-Franklin 2-message identity-based anamorphic encryption extension Π_{aBF} described in Fig. 4 satisfies anamorphic and IND-CPA security (Definition 5 and Appendix A.6).*

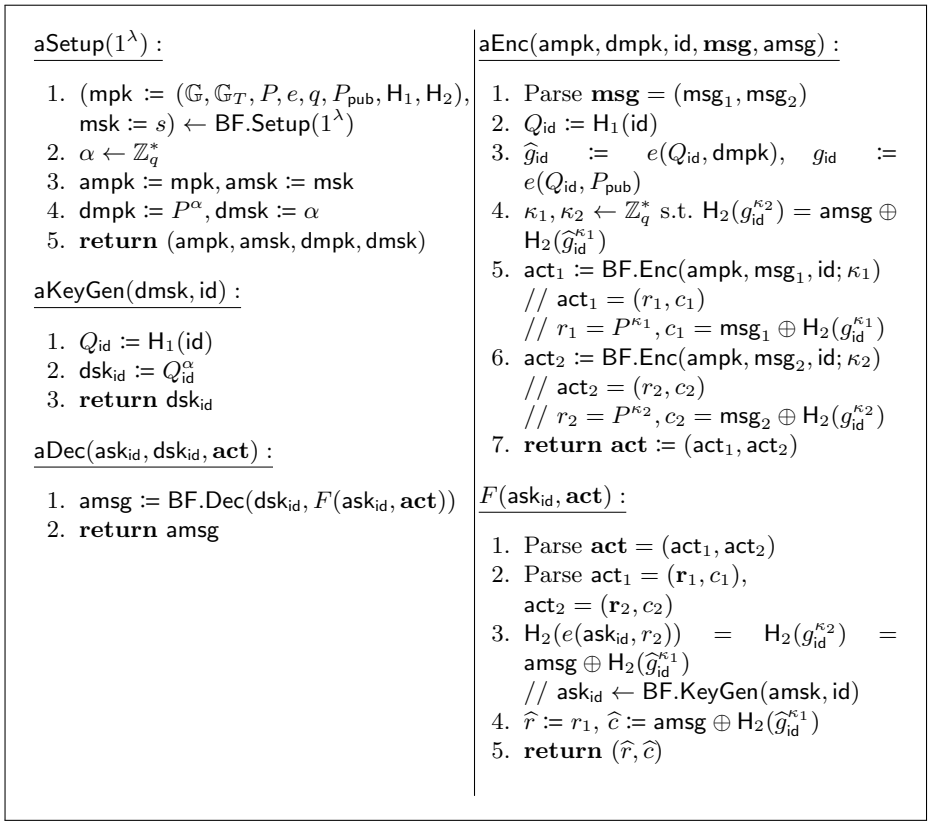

Fig. 4: 2-IBAEE Π_{aBF}

5 Public Key Anamorphic Signature Extension

In this section, we formalize the notion of μ-message anamorphic extension of a signature scheme along with definitions of its security. For brevity, we refrain from defining public key anamorphic signatures first and immediately establish security notions for the extension. To keep our exposition and constructions of this new concept simple, we refrain from considering a robustness notion for now. However, Appendix J will informally discuss such a notion and the difficulties to achieve it regarding our designs.

Definition 7 (μ-Message Public Key Anamorphic Signature Extension) *A tuple $\Pi_{\mathsf{PKASE}} = (\mathsf{aGenS}, \mathsf{aGenR}, \mathsf{aSig}, \mathsf{aDec})$ of PPT algorithms is a μ-message public key anamorphic signature extension (μ-PKASE) with anamorphic message space $\widehat{\mathcal{M}}$ associated with a signature scheme $\Pi_{\mathsf{SS}} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sig}, \mathsf{Verify})$ with normal message space $\mathcal{M}$ and is defined as follows, where $\mathsf{gp} \leftarrow \mathsf{SS.Setup}(1^\lambda)$:*

$\underline{(\mathsf{ask}, \mathsf{avk}, \mathsf{td}) \leftarrow \mathsf{aGenS}(\mathsf{gp}) :}$ *The anamorphic key generation algorithm* aGenS *is executed by the sender of the signature, takes as input the global parameters* gp *generated by* Setup *of the associated signature scheme, and outputs a signature key pair* (ask, avk) *as well as some secret trapdoor information* td.

$\underline{(\mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGenR}(\mathsf{gp}) :}$ *The anamorphic key generation algorithm* aGenR *is executed by the receiver of the signature, takes as input the global parameters* gp *generated by* Setup *of the associated signature scheme, and outputs a double key pair* (dk, tk).

$\underline{\mathbf{asig} \leftarrow \mathsf{aSig}(\mathsf{ask}, \mathsf{td}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg}) :}$ *The anamorphic signing algorithm* aSig *takes as input the sender's signing key* ask *with associated trapdoor information* td*, and the receiver's encryption double key* dk*, a regular message vector* $\mathbf{msg} \in \mathcal{M}^\mu$ *and an anamorphic message* $\mathsf{amsg} \in \widehat{\mathcal{M}}$*, and outputs an anamorphic signature vector* **asig**.

$\underline{\mathsf{amsg} \leftarrow \mathsf{aDec}(\mathsf{tk}, \mathsf{avk}, \mathbf{asig}, \mathbf{msg}) :}$ *The anamorphic decryption algorithm takes as input the receiver's secret decryption double key* tk*, the sender's signature verification key* avk*, and an anamorphic signature vector* **asig** *along with a message vector* **msg**, *and outputs an anamorphic message* amsg *or* $\perp$.

It must satisfy the following properties.

Correctness: *For all* $\lambda \in \mathbb{N}$, $\mathsf{gp} \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathbf{msg}, \mathsf{amsg}) \in \mathcal{M}^\mu \times \widehat{\mathcal{M}}$*, it holds that*

$$\Pr\left[\mathsf{amsg} \neq \mathsf{aDec}(\mathsf{tk}, \mathsf{avk}, \mathbf{asig}, \mathbf{msg}) \;:\; \begin{array}{c}(\mathsf{ask}, \mathsf{avk}, \mathsf{td}) \leftarrow \mathsf{aGenS}(\mathsf{gp}) \\ (\mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGenR}(\mathsf{gp}) \\ \mathbf{asig} \leftarrow \mathsf{aSig}(\mathsf{ask}, \mathsf{td}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})\end{array}\right] \leq \mathsf{negl}(\lambda)$$

where the probability is taken over the random coins of aGenS, aGenR, aSig.

Anamorphic security: *For every PPT dictator* $\mathcal{D}$ *it holds that*

$$\left|\Pr[\mathsf{GReal}^{\mathcal{D}}_{\mathsf{SS}}(\lambda) = 1] - \Pr[\mathsf{GAnam}^{\mathcal{D}}_{\mathsf{PKASE}}(\lambda) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security games are defined as follows, and $\mathsf{Sig}(\mathsf{sk}, \mathbf{msg}) := (\mathsf{Sig}(\mathsf{sk}, \mathsf{msg}))_{\mathsf{msg} \in \mathbf{msg}}$*:*

$\underline{\mathsf{GReal}^{\mathcal{D}}_{\mathsf{SS}}(\lambda) :}$	$\underline{\mathsf{GAnam}^{\mathcal{D}}_{\mathsf{PKASE}}(\lambda) :}$
1. $\mathsf{gp} \leftarrow \mathsf{Setup}(1^\lambda)$	1. $\mathsf{gp} \leftarrow \mathsf{Setup}(1^\lambda)$
2. $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\mathsf{gp})$	2. $(\mathsf{ask}, \mathsf{avk}, \mathsf{td}) \leftarrow \mathsf{aGenS}(\mathsf{gp})$
3. ***return*** $\mathcal{D}^{\mathsf{OSig}(\cdot,\cdot)}(\mathsf{sk}, \mathsf{vk})$ *where* $\mathsf{OSig}(\mathbf{msg}, \mathsf{amsg}) := \mathsf{Sig}(\mathsf{sk}, \mathbf{msg})$.	3. $(\mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGenR}(\mathsf{gp})$
	4. ***return*** $\mathcal{D}^{\mathsf{OaSig}(\cdot,\cdot)}(\mathsf{ask}, \mathsf{avk})$ *where* $\mathsf{OaSig}(\mathbf{msg}, \mathsf{amsg}) := \mathsf{aSig}(\mathsf{ask}, \mathsf{td}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})$.

Anamorphic security roughly guarantees that the two distributions $\{\mathsf{sk}, \mathsf{vk}, \mathsf{Sig}(\mathsf{sk}, \mathbf{msg})\}$ and $\{\mathsf{ask}, \mathsf{avk}, \mathsf{aSig}(\mathsf{ask}_S, \mathsf{dk}_R, \mathsf{msg}, \mathsf{amsg})\}$ are indistinguishable.

Preserving EUF/SUF-CMA security. We note that anamorphic security clearly also holds for an adversary not having access to sk. Such an adversary does not notice whether it interacts with the regular signature scheme or the anamorphic version. This clearly implies that the EUF-CMA or SUF-CMA security of Π_{SS} carries over to the individual signatures of the anamorphic extension.
Choice of gp. As opposed to PKAE, the benefits of deploying *asymmetric* crypto to realize the anamorphic channel for signatures would be highly reduced if each and every signer chose its own unique gp. In this case, a receiver of an anamorphic message needs to generate an individual key pair $(\mathsf{dk}, \mathsf{tk})$ for each signer it wants to receive anamorphic messages from. To this end, we make the realistic assumption that signers only choose from a small set of gp settings, e.g., standardized elliptic curves, or the dictator fixes some gp for everyone.

IND-CPA security of anamorphic signatures. Another natural security property for an anamorphic signature scheme is to guarantee that within a group of people that communicate via this embedded covert channel, no information about a message is leaked to parties other than the intended receiver. More concretely, if a party A knows dk_B of party B because it previously has sent a hidden message to it, then no information about the hidden message contained in an anamorphic signature from party C to party B should be leaked to party A. This notion is captured by the following IND-CPA-style security definition.

Definition 8 (IND-CPA Security for PKASE) *A μ-PKASE scheme Π_{PKASE} associated with signature scheme Π_{SS} is said to satisfy* IND-CPA *security if for every PPT adversary $\mathcal{A}$ it holds that*

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

where the security game is defined as follows:

$\underline{\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\mathsf{PKASE}}(\lambda,\beta):}$

1. $\mathsf{gp} \leftarrow \mathsf{Setup}(1^\lambda)$
2. $(\mathsf{ask}, \mathsf{avk}, \mathsf{td}) \leftarrow \mathsf{aGenS}(\mathsf{gp})$
3. $(\mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGenR}(\mathsf{gp})$
4. $(\mathbf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1) \leftarrow \mathcal{A}^{\mathsf{OaSig}(\cdot,\cdot)}(\mathsf{avk}, \mathsf{dk})$
5. $\mathbf{asig}_\beta \leftarrow \mathsf{aSig}(\mathsf{ask}, \mathsf{td}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg}_\beta)$
6. ***return*** $\beta' \leftarrow \mathcal{A}^{\mathsf{OaSig}(\cdot,\cdot)}(\mathsf{asig}_\beta)$ *where* $\mathsf{OaSig}(\mathbf{msg}, \mathsf{amsg}) := \mathsf{aSig}(\mathsf{ask}, \mathsf{td}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})$.

(s)IND-(R)CCA security of anamorphic signatures. Similar to the anamorphic encryption case, we can also define stronger security properties for anamorphic signatures by providing a (restricted) aDec oracle or ask as input to $\mathcal{A}$. So we define IND-CCA security by providing an $\mathsf{aDec}(\mathsf{tk}, \mathsf{avk}, \cdot, \cdot)$ oracle (rejecting if queried with challenge $(\mathbf{asig}_\beta, \mathbf{msg})$), sIND-CCA by additionally giving ask as input to $\mathcal{A}$, and sIND-RCCA like sIND-CCA but restricting the oracle to reject if $\mathsf{aDec}(\mathsf{tk}, \mathsf{avk}, \mathbf{asig}, \mathbf{msg}) \in \{\mathsf{amsg}_0, \mathsf{amsg}_1\}$.

5.1 Waters Signatures with Linear Encryption

Subsequently, we present a PKASE construction based on Water's signatures [36] and linear encryption (LE) [7]. LE is a variant of ElGamal relying on the Decision linear (DLIN) assumption that can be instantiated in a group $\mathbb{G}$ where the DDH assumption is easy, e.g., in symmetric pairing groups. We consider the Waters scheme over symmetric pairings groups $(\mathbb{G}, \mathbb{G}_T, g, e, q)$ for simplicity, but it can straightforwardly be translated to an asymmetric pairing setting (e.g., type-3 pairings). However, we note that the anamorphic extension still requires LE.[5]

We first recall that Waters signatures are pairing-based signatures EUF-CMA secure under the CDH assumption. They rely on the Waters hash $H(m) \coloneqq h_0 \prod_{i=1}^{n} h_i^{m_i}$, where m_i denotes the i'th bit of message $m \in \{0,1\}^n$ and $h_i \leftarrow \mathbb{G}$. A signature consists of $(\sigma_1, \sigma_2) \coloneqq (g^r, g^{\alpha\beta} H(m)^r)$ for random $r \in \mathbb{Z}_q$, secret key $g^{\alpha\beta}$ and public key $(g^\alpha, g^\beta, h_0, \ldots, h_n)$. A signature is verified by checking the equality $e(g^\alpha, g^\beta) \cdot e(\sigma_1, H(m)) = e(g, \sigma_2)$. Also recall that in linear encryption the public key is $(u, v, h) \in \mathbb{G}^3$ and the secret key $(x, y) \in \mathbb{Z}_q$ such that $u^x = v^y = h$ holds. Encrypting a message $m \in \mathbb{G}$ amounts to choosing random κ_1, κ_2 and computing ciphertext $(c_1, c_2, c_2) \coloneqq (u^{\kappa_1}, v^{\kappa_2}, h^{\kappa_1+\kappa_2} \cdot m)$ and decryption computes $m = c_3 \cdot (c_1^x \cdot c_2^y)^{-1}$. LE is IND-CPA secure under DLIN.

The essential idea is to embed a ciphertext (c_1, c_2, c_3) of an anamorphic message amsg as the first components of three Waters signatures as this component is uniformly chosen from $\mathbb{G}$ and a LE ciphertext is indistinguishable from a random $\mathbb{G}^3$ element under DLIN (cf. IND$-CPA, Definition 19). However, we face the problem that the discrete logarithm r of this component is needed to compute the second component of the signature and for the c_i the DLs are unknown. To circumvent this problem, we make the assumption that every user generates its own parameters for the Waters hash and thus knows a_i such that $h_i = g^{a_i}$ for $i \in \{0, \ldots, n\}$ and that this is accepted by the dictator.[6] Using the a_i allows us to compute the second signature component without knowing the DL r.

The resulting scheme is presented in Figure 5. We note that the same strategy can also be applied to the SUF-CMA secure variant by Boneh, Shen and Waters [9]. The security of anamorphic signature extensions following this strategy is implied by our master theorem (Theorem 5) presented in Section 5.3.

5.2 BBS with ElGamal and Pseudo-Random Encodings

BBS signatures, as introduced in [7], are a very well-known signature scheme and currently in the process of *standardization* by the IETF [28]. A formal proof of its SUF-CMA security under the k-SDH assumption in the standard model has only recently been given in [34]. We consider BBS defined in asymmetric (type-3) pairing groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, q)$ of prime order q. The public key is g_2^x for secret key $x \in \mathbb{Z}_q$. For a signature on message $m \in \mathbb{Z}_q^\ell$, one samples a random

[5] We cannot rely on DDH as elements of the ElGamal ciphertext would be required in both source groups.

[6] We are aware that this is a somewhat strong assumption as this could be suspicious for the dictator. Our second PKASE from BBS does not suffer from this limitation

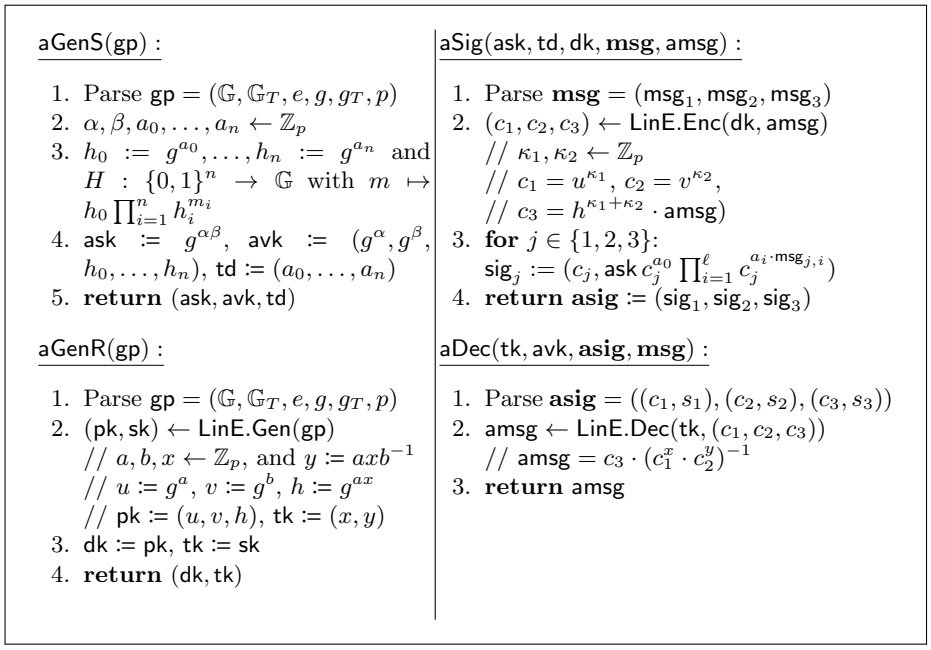

Fig. 5: 3-PKASE Π_{aWaters}

$d \in \mathbb{Z}_q$ and computes $\sigma = (\sigma_1, \sigma_2) = (d, H(m)^{\frac{1}{x+d}}) \in \mathbb{Z}_q \times \mathbb{G}_1$. Verification is straightforward by checking whether $e(\sigma_2, g_2) = e(H(m), g_2^{\sigma_1} \cdot g_2^x)$. Regarding our construction, we may omit the detail that the hash $H(m) := g_1 \cdot \prod_{i=1}^{\ell} h_i^{m_i}$ is as in Waters scheme (though with $m_i \in \mathbb{Z}_q$) as this irrelevant. We set $\ell = 1$ in the following.

The basic idea (to obtain an IND-CPA secure extension) is to apply ElGamal encryption over $\mathbb{G}_1$ and to embed a ciphertext, which is pseudo-random over $\mathbb{G}_1^2$ under the DDH assumption, into a random component of two BBS signatures. The first component d of such a signature is randomly chosen, but unfortunately over $\mathbb{Z}_q$ instead of $\mathbb{G}_1$. To this end, we need to specify a pseudo-random $(\mathbb{G}_1^2, \mathbb{Z}_q^\mu)$-encoding. Again, we can deploy Elligator Squared [35] which also provides a suitable encoding for pairing-friendly curves $\mathbb{E}(\mathbb{F}_p)$ to $\mathbb{F}_p^2$. For instance, let us consider BN curves [4] for instantiating the asymmetric pairing-based setting of our BBS signatures. Then $\mathbb{G}_1 = \mathbb{E}(\mathbb{F}_p)$ is a cyclic group of prime order q. Note that for a random $h \in \mathbb{G}_1$, Elligator Squared will return a close to random element from $\mathbb{F}_p^2$ but not necessarily $\mathbb{F}_q^2$ as desired. However, since p is very close to q, more precisely $p - q \approx \sqrt{p}$ (cf. Hasse's bound), the probability, assuming the standard element representation for $\mathbb{F}_p$ and $\mathbb{F}_q$ as $\{0, \ldots, p-1\}$ and $\{0, \ldots, q-1\}$,

that we hit an element which is not in $\mathbb{F}_q^2$ is negligible, namely at most $\frac{2}{\sqrt{p}}$.[7] In this context, observe that we are actually not forced to do ElGamal encryption in the pairing group that BBS uses. We could alternatively use ElGamal (or any other secure encryption) over any other group as long as we are able find a suitable pseudo-random encoding of a ciphertext as $\mathbb{Z}_q^\mu$ element.

The 4-message anamorphic extension Π_{aBBS} resulting from a pseudo-random $(\mathbb{G}_1^2, \mathbb{Z}_q^4)$-encoding (e.g., using Elligator Squared for $\mathbb{G}_1$ of BN curves) is shown in Figure 6. Note that no trapdoor is required here to compute the second signature component. Π_{aBBS} achieves anamorphic security and IND-CPA security. To achieve sIND-RCCA security, it suffices to replace plain ElGamal by an FO-based PKE from ElGamal and XOR-based symmetric encryption and extend the pseudo-random encoding described above to also cover the symmetric ciphertext component. For q close to 2^λ, a random bitstring $G(r) \oplus \mathsf{amsg} \in \{0,1\}^\lambda$ can simply be interpreted as $\mathbb{F}_q$-element s_5 which causes an encoding error (integer $> p-1$) with negligible probability. This leads to a 5-message anamorphic signature extension satisfying sIND-RCCA security. To additionally achieve IND-CCA security, one needs to make use of the SUF-CMA security of BBS, by checking the validity of the signatures **asig** wrt. to the messages **msg** in aDec.

Also for other variants of BBS signatures like BBS+ [2], such anamorphic extensions can be obtained in the same way. The security of all these extensions is covered by our master theorem in Section 5.3.

5.3 Anamorphism-Friendly Signatures with Compliant Encryption

Generalizing the ideas from Section 5.1 and Section 5.2, the following definition covers signature schemes which are canonical candidates for an anamorphic extension. The signatures generated by these schemes have at least one uniformly (and freshly) chosen component and allow the rest of the signature to be computed (cf. RemSig below) by treating the uniform components in a black-box way, that means, without using additional information related to the generation of these components (e.g., the DL d used to create a random group element g^d). To enable the latter, some trapdoor information from the key generation might be used (cf. $\mathsf{SS.Gen}'$ below). An encryption scheme is compliant to such a signature scheme, in the sense that it easily allows for the construction of an anamorphic extension, if its ciphertext components are pseudo-random (IND$-CPA, see Definition 19) and can be completely embedded in one or multiple signatures.

Definition 9 (Anamorphism-Friendly Sign. & Compliant Enc.) *Let Π_{SS} be a signature scheme with a signature space that can be written as $S_0^{e_0} \times \ldots \times S_{m-1}^{e_{m-1}} \times R$, where $S_i \neq S_j$ and $e_i > 0$, such that the following conditions hold:*

1. *Random components: $m - 1 \geq 0$ and for signatures $(s_0, \ldots, s_{m-1}, r)$ generated by SS.Sig holds that all $s_i \leftarrow S_i^{e_i}$ are uniformly and freshly chosen.*

[7] To be compliant to the definition of our pseudo-random encoding, we could output $\perp$ in this case.

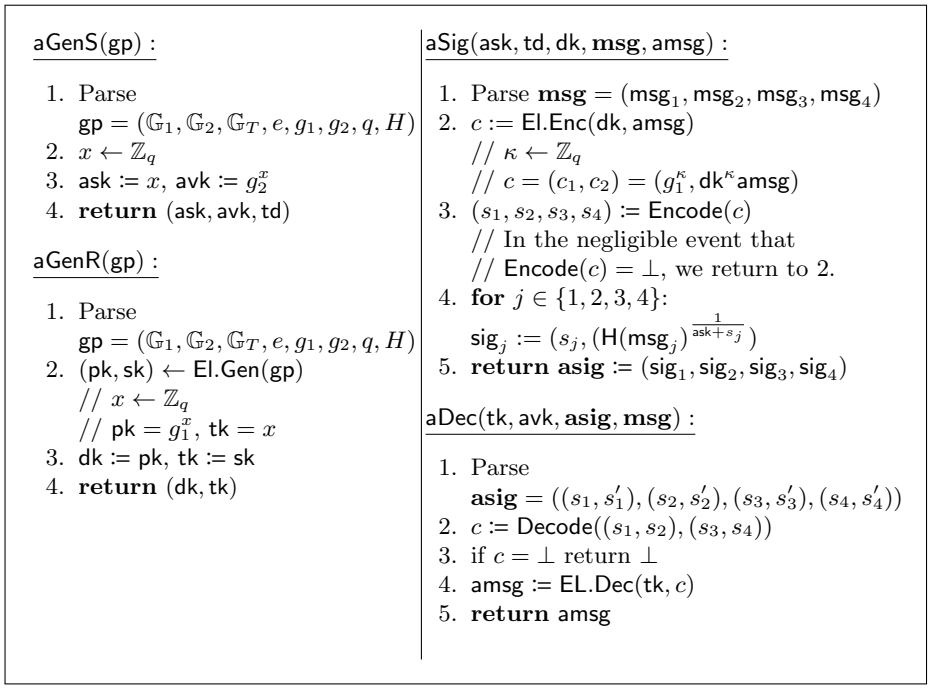

Fig. 6: 4-PKASE Π_{aBBS}

2. *Black-box computation of remainder: There exist two PPT algorithms* $\mathsf{SS.Gen}'$, $\mathsf{SS.RemSig}$ *such that* $(\mathsf{sk}, \mathsf{vk}) \overset{c}{\approx} (\mathsf{sk}', \mathsf{vk}')$, *where* $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{SS.Gen}(\mathsf{gp})$ *and* $(\mathsf{sk}', \mathsf{vk}', \mathsf{td}) \leftarrow \mathsf{SS.Gen}(\mathsf{gp})$, *and*

$$\Pr\left[b = b' : \begin{array}{c} \mathsf{gp} \leftarrow \mathsf{SS.Setup}(1^\lambda) \\ (\mathsf{sk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{SS.Gen}'(\mathsf{gp}) \\ \mathsf{msg} \leftarrow \mathcal{A}(\mathsf{sk}, \mathsf{vk}) \\ (s_0, \ldots, s_{m-1}, r_0) \leftarrow \mathsf{SS.Sig}(\mathsf{sk}, \mathsf{msg}) \\ r_1 \leftarrow \mathsf{SS.RemSig}(\mathsf{sk}, \mathsf{td}, \mathsf{msg}, (s_0, \ldots, s_{m-1})) \\ b \leftarrow \{0,1\} \\ b' \leftarrow \mathcal{A}(s_0, \ldots, s_{m-1}, r_b) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

.

In this case, we call Π_{SS} *anamorphism-friendly. We call an encryption scheme* Π_{ES} *compliant to* Π_{SS} *if it is* IND$-CPA *secure and its message space can be written as* $S_0^{e_0'} \times \ldots \times S_{m-1}^{e_{m-1}'}$ *with* $e_i' > 0$.

In this situation, we can securely embed a ciphertext containing the anamorphic message into the components of one or multiple signatures. This is specified by the aSig algorithm in Figure 7. In Step 3, as many ciphertext components

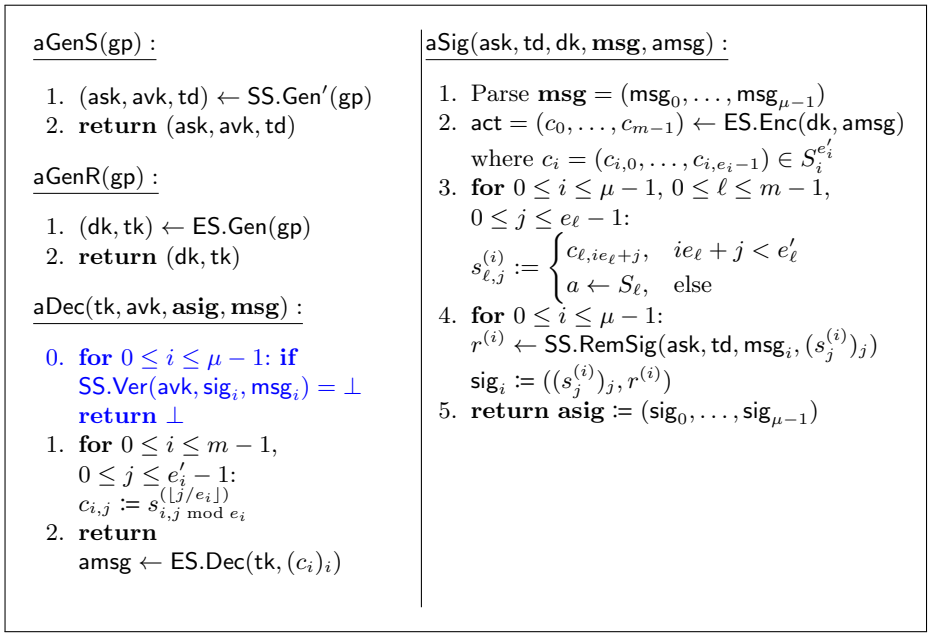

Fig. 7: Generic μ-PKASE Π_{aGAE} and variant $\Pi_{\mathsf{aGAE}'}$

as possible are packed into the same signature. As it may happen that the designated signature components cannot be completely "filled" with ciphertext components, we use truly random elements in this case. The remainder of the signature is computed in Step 4 using RemSig. When receiving μ signatures, the ciphertext components are first unpacked in Step 2 of aDec before amsg is recovered. By following this blueprint using an anamorphism-friendly signature and compliant encryption, we obtain a secure anamorphic extension Π_{aGAE} also satisfying IND-CPA security. If the underlying encryption additional satisfies IND-RCCA security then Π_{aGAE} achieves sIND-RCCA security. Note that an IND-CCA secure PKE still leads to an sIND-RCCA but not IND-CCA secure extension when only following Steps 1 and 2 of of aDec in Figure 7 as the adversary (against the extension) may tamper with the $r^{(i)}$ of the challenge which would lead to a different **asig** decrypting to amsg_β. To additionally achieve IND-CCA security, aDec needs to verify the signatures **asig** (Step 0 in Figure 7) and the signature scheme needs to be SUF-CMA secure. Still sIND-CCA security cannot be achieved in this way as, given ask, the adversary may re-use the challenge $c_{i,j}$ to generate a new signature. We prove the following theorem in Appendix G.

Theorem 5 *Let Π_{SS} be an anamorphism-friendly signature scheme and Π_{ES} an encryption scheme compliant to Π_{SS}. Then Π_{aGAE} as defined in Figure 7 is a μ-message anamorphic signature extension, where $\mu = \max\{\lceil \frac{e'_i}{e_i} \rceil\}$, satisfying anamorphic and* IND-CPA *security. If Π_{ES} additionally achieves* IND-RCCA

security then Π_{aGAE} *achieves* sIND-RCCA *security. Moreover, if* Π_{ES} *satisfies* IND-CCA *security and* Π_{SS} *satisfies* SUF-CMA *security, then the variant* $\Pi_{\mathsf{aGAE}'}$ *in Figure 7 achieves* IND-CCA *security.*

The anamorphic extension of Waters signatures using linear encryption presented in Section 5.1 is an instance of our general blueprint and thus its security follows from Theorem 5. The same holds for BBS and the encryption scheme obtained by combining ElGamal with a suitable pseudo-random encoding from Section 5.2. We prove the following theorems in Appendix H.

Theorem 6 *Waters signatures (as defined in Sec. 5.1) are anamorphism-friendly and linear encryption is compliant to them under the* DLIN *assumption.*

Theorem 7 *BBS signatures (as defined in Sec. 5.2) are anamorphism-friendly and ElGamal encryption, where ciphertexts are encoded during encryption and decoded during decryption using a pseudo-random* $\mathbb{Z}_p^k$*-encoding, is compliant to BBS under the* DDH *assumption.*

References

1. Ananth, P., Deshpande, A., Kalai, Y.T., Lysyanskaya, A.: Fully homomorphic NIZK and NIWI proofs. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019: 17th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 11892, pp. 356–385. Springer, Cham (Dec 2019). https://doi.org/10.1007/978-3-030-36033-7_14
2. Au, M.H., Susilo, W., Mu, Y.: Constant-size dynamic k-TAA. In: Prisco, R.D., Yung, M. (eds.) SCN 06: 5th International Conference on Security in Communication Networks. Lecture Notes in Computer Science, vol. 4116, pp. 111–125. Springer, Berlin, Heidelberg (Sep 2006). https://doi.org/10.1007/11832072_8
3. Banfi, F., Gegier, K., Hirt, M., Maurer, U., Rito, G.: Anamorphic encryption, revisited. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024, Part II. Lecture Notes in Computer Science, vol. 14652, pp. 3–32. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58723-8_1
4. Barreto, P.S.L.M., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Preneel, B., Tavares, S. (eds.) SAC 2005: 12th Annual International Workshop on Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 3897, pp. 319–331. Springer, Berlin, Heidelberg (Aug 2006). https://doi.org/10.1007/11693383_22
5. Bellare, M., Rogaway, P.: Optimal asymmetric encryption. In: Santis, A.D. (ed.) Advances in Cryptology – EUROCRYPT'94. Lecture Notes in Computer Science, vol. 950, pp. 92–111. Springer, Berlin, Heidelberg (May 1995). https://doi.org/10.1007/BFb0053428
6. Boneh, D.: The decision diffie-hellman problem. In: International algorithmic number theory symposium. pp. 48–63. Springer (1998)
7. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) Advances in Cryptology – CRYPTO 2004. Lecture Notes in Computer Science, vol. 3152, pp. 41–55. Springer, Berlin, Heidelberg (Aug 2004). https://doi.org/10.1007/978-3-540-28628-8_3

8. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, vol. 2139, pp. 213–229. Springer, Berlin, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_13
9. Boneh, D., Shen, E., Waters, B.: Strongly unforgeable signatures based on computational Diffie-Hellman. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006: 9th International Conference on Theory and Practice of Public Key Cryptography. Lecture Notes in Computer Science, vol. 3958, pp. 229–240. Springer, Berlin, Heidelberg (Apr 2006). https://doi.org/10.1007/11745853_15
10. Canetti, R., Krawczyk, H., Nielsen, J.B.: Relaxing chosen-ciphertext security. In: Boneh, D. (ed.) Advances in Cryptology – CRYPTO 2003. Lecture Notes in Computer Science, vol. 2729, pp. 565–582. Springer, Berlin, Heidelberg (Aug 2003). https://doi.org/10.1007/978-3-540-45146-4_33
11. Catalano, D., Giunta, E., Migliaro, F.: Anamorphic encryption: New constructions and homomorphic realizations. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024, Part II. Lecture Notes in Computer Science, vol. 14652, pp. 33–62. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58723-8_2
12. Catalano, D., Giunta, E., Migliaro, F.: Generic anamorphic encryption, revisited: New limitations and constructions. Cryptology ePrint Archive, Paper 2024/1119 (2024), https://eprint.iacr.org/2024/1119
13. Catalano, D., Giunta, E., Migliaro, F.: Limits of black-box anamorphic encryption. Cryptology ePrint Archive, Paper 2024/1098 (2024), https://eprint.iacr.org/2024/1098
14. Chen, L., Cheng, Z.: Security proof of sakai-kasahara's identity-based encryption scheme. In: Smart, N.P. (ed.) Cryptography and Coding, 10th IMA International Conference, Cirencester, UK, December 19-21, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3796, pp. 442–459. Springer (2005). https://doi.org/10.1007/11586821_29, https://doi.org/10.1007/11586821_29
15. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) Advances in Cryptology – CRYPTO'98. Lecture Notes in Computer Science, vol. 1462, pp. 13–25. Springer, Berlin, Heidelberg (Aug 1998). https://doi.org/10.1007/BFb0055717
16. Diffie, W., Hellman, M.: New directions in cryptography (1976) (2021)
17. ElGamal, T.: On computing logarithms over finite fields. In: Williams, H.C. (ed.) Advances in Cryptology – CRYPTO'85. Lecture Notes in Computer Science, vol. 218, pp. 396–402. Springer, Berlin, Heidelberg (Aug 1986). https://doi.org/10.1007/3-540-39799-X_28
18. ETSI: Etsi tr 103 719: Guide to identity-based cryptography. https://www.etsi.org/deliver/etsi_tr/103700_103799/103719/01.01.01_60/tr_103719v010101p.pdf (2022)
19. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M.J. (ed.) Advances in Cryptology – CRYPTO'99. Lecture Notes in Computer Science, vol. 1666, pp. 537–554. Springer, Berlin, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-48405-1_34
20. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. Journal of Cryptology **26**(1), 80–101 (Jan 2013). https://doi.org/10.1007/s00145-011-9114-1
21. Galindo, D.: Boneh-Franklin identity based encryption revisited. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005: 32nd International Colloquium on Automata, Languages and Programming. Lecture Notes in

Computer Science, vol. 3580, pp. 791–802. Springer, Berlin, Heidelberg (Jul 2005). https://doi.org/10.1007/11523468_64
22. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC 2008. pp. 197–206. ACM (2008)
23. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 75–92. Springer, Berlin, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40041-4_5
24. Kitagawa, T., Yang, P., Hanaoka, G., Zhang, R., Watanabe, H., Matsuura, K., Imai, H.: Generic transforms to acquire cca-security for identity based encryption: The cases of fopkc and REACT. In: Batten, L.M., Safavi-Naini, R. (eds.) Information Security and Privacy, 11th Australasian Conference, ACISP 2006, Melbourne, Australia, July 3-5, 2006, Proceedings. Lecture Notes in Computer Science, vol. 4058, pp. 348–359. Springer (2006). https://doi.org/10.1007/11780656_29, https://doi.org/10.1007/11780656_29
25. Koppula, V., Waters, B.: Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part II. Lecture Notes in Computer Science, vol. 11693, pp. 671–700. Springer, Cham (Aug 2019). https://doi.org/10.1007/978-3-030-26951-7_23
26. Kutylowski, M., Persiano, G., Phan, D.H., Yung, M., Zawada, M.: Anamorphic signatures: Secrecy from a dictator who only permits authentication! In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023, Part II. Lecture Notes in Computer Science, vol. 14082, pp. 759–790. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38545-2_25
27. Kutylowski, M., Persiano, G., Phan, D.H., Yung, M., Zawada, M.: The self-anti-censorship nature of encryption: on the prevalence of anamorphic cryptography. Proceedings on Privacy Enhancing Technologies **2023**(4), 170–183 (2023)
28. Looker, T., Kalos, V., Whitehead, A., Lodder, M.: The BBS Signature Scheme. Internet-Draft draft-irtf-cfrg-bbs-signatures-07, Internet Engineering Task Force (Sep 2024), https://datatracker.ietf.org/doc/draft-irtf-cfrg-bbs-signatures/07/, work in Progress
29. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd Annual ACM Symposium on Theory of Computing. pp. 427–437. ACM Press (May 1990). https://doi.org/10.1145/100216.100273
30. Persiano, G., Phan, D.H., Yung, M.: Anamorphic encryption: Private communication against a dictator. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology – EUROCRYPT 2022, Part II. Lecture Notes in Computer Science, vol. 13276, pp. 34–63. Springer, Cham (May / Jun 2022). https://doi.org/10.1007/978-3-031-07085-3_2
31. Persiano, G., Phan, D.H., Yung, M.: Public-key anamorphism in (CCA-secure) public-key encryption and beyond. Cryptology ePrint Archive, Paper 2024/1327 (2024). https://doi.org/10.1007/978-3-031-68379-4_13, https://eprint.iacr.org/2024/1327
32. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM) **56**(6), 1–40 (2009)
33. Sakai, R., Ohgishi, S., Kasahara, M.: Cryptosystems based on pairing over elliptic curve (in japanese). In: Smart, N.P. (ed.) The 2001 Symposium on Cryptography and Information Security. Lecture Notes in Computer Science (2001)

34. Tessaro, S., Zhu, C.: Revisiting BBS signatures. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, Part V. Lecture Notes in Computer Science, vol. 14008, pp. 691–721. Springer, Cham (Apr 2023). https://doi.org/10.1007/978-3-031-30589-4_24
35. Tibouchi, M.: Elligator squared: Uniform points on elliptic curves of prime order as uniform random strings. In: Christin, N., Safavi-Naini, R. (eds.) FC 2014: 18th International Conference on Financial Cryptography and Data Security. Lecture Notes in Computer Science, vol. 8437, pp. 139–156. Springer, Berlin, Heidelberg (Mar 2014). https://doi.org/10.1007/978-3-662-45472-5_10
36. Waters, B.R.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. Lecture Notes in Computer Science, vol. 3494, pp. 114–127. Springer, Berlin, Heidelberg (May 2005). https://doi.org/10.1007/11426639_7
37. Yang, P., Kitagawa, T., Hanaoka, G., Zhang, R., Matsuura, K., Imai, H.: Applying fujisaki-okamoto to identity-based encryption. In: Fossorier, M.P.C., Imai, H., Lin, S., Poli, A. (eds.) Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, 16th International Symposium, AAECC-16, Las Vegas, NV, USA, February 20-24, 2006, Proceedings. Lecture Notes in Computer Science, vol. 3857, pp. 183–192. Springer (2006). https://doi.org/10.1007/11617983_18, https://doi.org/10.1007/11617983_18

SUPPLEMENTAL MATERIALS

A Additional Preliminaries

A.1 Notation

Let $\lambda \in \mathbb{N}$ denote the security parameter. We use PPT to denote a randomized algorithm running in time $a \cdot \lambda^c$ for some constants a and c. A function $f : \mathbb{N} \to \mathbb{R}^+$ is called negligible in λ, if it grows slower than λ^{-c} for every constant c. We use $\mathsf{negl}(\lambda)$ to denote a generic negligible function. We denote the set of positive integers upto n as $[n] \coloneqq \{1, \ldots, n\}$. For a finite set S, we use $x \leftarrow S$ to denote the uniformly sampling x from S. For a probabilistic algorithm $\mathcal{A}$, we let $y \leftarrow \mathcal{A}(x)$ denote the process of running $\mathcal{A}(x)$ with fresh randomness and assigning the result to y. We represent empty strings using ϵ. We use lowercase bold letters to denote vectors (e.g. $\mathbf{k}$) and uppercase bold letters for matrices (e.g. $\mathbf{A}$).

A.2 Public Key Encryption

We first recall the standard syntax of PKE which we require to have indistinguishable encryptions under the chosen plaintext attack (IND-CPA). We note that the scheme may also be defined with respect to a global setup $\mathsf{gp} \leftarrow \mathsf{Setup}(1^\lambda)$ generating parameters gp used by Gen.

Definition 10 (Public Key Encryption) *We say that a triplet* $\Pi_{\mathsf{PKE}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *of PPT algorithms with a message space* $\mathcal{M}$ *is an* IND-CPA *secure public key encryption (PKE) scheme if the following holds:*

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$: *The key generation algorithm* aGen *takes as input the security parameter* 1^λ*, and outputs a secret key* sk *and a public key* pk.

$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$: *The encryption algorithm* Enc *takes as input the public key* pk*, and a message* $\mathsf{msg} \in \mathcal{M}$*, and outputs a ciphertext* ct.

$\mathsf{msg} \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: *The decryption algorithm* Dec *takes as input the secret key* sk *and the ciphertext* ct*, and outputs a message* msg.

Furthermore, the algorithms must satisfy the following correctness and security properties.

Correctness: *For every message* $\mathsf{msg} \in \mathcal{M}$*, the following holds:*

$$\Pr\left[\mathsf{msg} = \mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) : \begin{matrix}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{msg})\end{matrix}\right] = 1$$

IND-CPA security: *For every PPT adversary $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for all* $\lambda \in \mathbb{N}$*, the following holds:*

$$\left|\Pr[\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{PKE}}}(\lambda, 0) = 1] - \Pr[\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{PKE}}}(\lambda, 1) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security game is defined as follows:

$\underline{\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{PKE}}}(\lambda, \beta) :}$

1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$
2. $(\mathsf{msg}_0, \mathsf{msg}_1) \leftarrow \mathcal{A}(\mathsf{pk})$
3. $\mathsf{ct}_\beta = \mathsf{Enc}(\mathsf{pk}, \mathsf{msg}_\beta)$
4. ***return*** $\beta' \leftarrow \mathcal{A}(\mathsf{ct}_\beta)$

A.3 Signature Scheme

We now recall the definition of a basic signature scheme and note that the scheme may also be defined with respect to a global setup with parameters gp.

Definition 11 (Signature Scheme) *We say that a triplet* $\Pi_{\mathsf{s}} = (\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ *of PPT algorithms with a message space* $\mathcal{M}$ *is a signature scheme if the following holds:*

$\underline{(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda) :}$ *The key generation algorithm* Gen *takes as input the security parameter* 1^λ*, and outputs a signing key* sk *and a verification key* vk*.*
$\underline{\mathsf{sig} \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{msg}) :}$ *The signing algorithm* Sig *takes as input the signing key* sk*, and a message* $\mathsf{msg} \in \mathcal{M}$ *and outputs a signature* sig*.*
$\underline{1 \leftarrow \mathsf{Ver}(\mathsf{vk}, \mathsf{msg}, \mathsf{sig}) :}$ *The verification algorithm takes input the verification key* vk*, the message* msg *and the signature* sig*, and outputs* 1*.*

Furthermore, the algorithms must satisfy the following correctness and security properties.

Correctness: *For every message* $\mathsf{msg} \in \mathcal{M}$*, the following holds:*

$$\Pr\left[\mathsf{Ver}(\mathsf{vk}, \mathsf{msg}, \mathsf{sig}) = 1 : \begin{matrix}(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda) \\ \mathsf{sig} \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{msg})\end{matrix}\right] = 1$$

Security: *The signature scheme* Π_{SS} *is secure against chosen-message attacks if for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for all* $\lambda \in \mathbb{N}$*, the following holds:*

$$\left|\Pr[\mathsf{GSig\text{-}CMA}^{\mathcal{A}}_{\Pi_{\mathsf{SS}}}(\lambda) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security game is defined as follows:

$\mathsf{GSig\text{-}CMA}^{\mathcal{A}}_{\Pi_{\mathsf{SS}}}(\lambda)$:

1. $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$
2. $(\mathsf{msg}^*, \mathsf{sig}^*) \leftarrow \mathcal{A}^{\mathsf{OSig}(\mathsf{sk},\cdot)}(\mathsf{vk})$
   where $\mathsf{OSig}(\mathsf{sk}, \mathsf{msg}) = (\mathsf{msg}, \mathsf{sig})$
3. **if** $\mathsf{Ver}(\mathsf{vk}, \mathsf{msg}^*, \mathsf{sig}^*) = 1$ *and*
   $(\mathsf{msg}^*, \mathsf{sig}^*)$ *has not returned by* OSig
   return 1*;*
   else return 0

A.4 Computational Assumptions

Next, we recall the various hardness assumptions used in our constructions. We begin with the DDH assumption by Boneh [6], which is used in our ElGamal based public key anamorphic encryption.

Definition 12 (The DDH Assumption [6]) *Let $\lambda \in \mathbb{N}$ be the security parameter and let q be a polynomial in λ. Let $\mathbb{G} = \langle g \rangle$ be a group of order $q > 2^\lambda$. Then for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, the following holds:*

$$|\Pr[a, b \leftarrow \mathbb{Z}_q : \mathcal{A}(g^a, g^b, g^{ab}) = 1] - \Pr[a, b, c \leftarrow \mathbb{Z}_q : \mathcal{A}(g^a, g^b, g^c) = 1]| \leq \mathsf{negl}(\lambda)$$

where the probability is taken over the coin tosses of $\mathcal{A}$ and random choices of a, b, c and g.

We now state the CDH assumption used in our PKAS construction based on Water's signatures.

Definition 13 (The CDH Assumption [16]) *Let $q \in \mathbb{N}$ be a large prime and let $\mathbb{G} = \langle g \rangle$ be a multiplicative cyclic group of order q. Let $a, b \leftarrow \mathbb{Z}_q$. Then the* CDH *assumption is to compute g^{ab}.*

Next, we recall the DLWE assumption by Regev [32] used in our Dual-Regev based public key anamorphic encryption.

Definition 14 (The DLWE Assumption [32]) *Let $q \in \mathbb{N}$ be a large prime and let $n, m \in \mathbb{N}$. Let χ be a noise distribution over $\mathbb{Z}_q$. The (n, q, χ)-* DLWE *assumption of dimension m states that the following distributions are computationally indistinguishable:*

$$(\mathbf{A}, \mathbf{s}^T \cdot \mathbf{A} + \mathbf{e}^T) \overset{c}{\approx} (\mathbf{A}, \boldsymbol{u}) : \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \chi^m, \mathbf{u} \leftarrow \mathbb{Z}_q^m$$

We now recall the DLIN assumption by Boneh, Boyen and Shacham [7] used in our Waters signature based public key anamorphic signature.

Definition 15 (The DLIN Assumption [7]) *Let $q \in \mathbb{N}$ be a large prime and let $\mathbb{G}$ be a multiplicative cyclic group of order q, where u, v, h are uniformly random generators of $\mathbb{G}$. Let a, b be uniformly random elements of $\mathbb{Z}_q$. Then for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, the following holds:*

$$|\Pr[u, v, h \leftarrow \mathbb{G} : \mathcal{A}(u, v, h, u^a, v^b, h^{a+b}) = 1] - \Pr[u, v, h, \eta \leftarrow \mathbb{G} : \mathcal{A}(u, v, h, u^a, v^b, \eta) = 1]| \leq \mathsf{negl}(\lambda)$$

where the probability is taken over the uniformly random choice of parameters to $\mathcal{A}$, and over the coin tosses of $\mathcal{A}$.

Definition 16 (The DBDH Assumption [8]) *Let $q \in \mathbb{N}$ be a large prime and let $(\mathbb{G}, \mathbb{G}_T, g, e, q)$ be a pairing group, where g is a generator of $\mathbb{G}$. Then for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, the following holds:*

$$|\Pr[a, b, c \leftarrow \mathbb{Z}_q : \mathcal{A}(g^a, g^b, g^c, e(g, g)^{abc}) = 1] - \Pr[a, b, c, t \leftarrow \mathbb{Z}_q : \mathcal{A}(g^a, g^b, g^c, e(g, g)^t) = 1]| \leq \mathsf{negl}(\lambda)$$

where the probability is taken over the coin tosses of $\mathcal{A}$, and random choices of a, b, c, t, g.

A.5 Additional IND-CPA Security Definitions

Definition 17 *Let Π_{PKE} be a PKE with ciphertext space denoted by $\mathcal{C}$. We call* PKE IND$-CPA *secure if for every PPT adversary $\mathcal{A}$ it holds that*

$$\left|\Pr[\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{PKE}}}(\lambda, 1) = 1] - \Pr[\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{PKE}}}(\lambda, 0) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security game is defined as follows:

$\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{PKE}}}(\lambda, \beta)$:

1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$
2. $\mathsf{msg} \leftarrow \mathcal{A}(\mathsf{pk})$
3. $\mathsf{ct}_0 \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$
4. $\mathsf{ct}_1 \leftarrow \mathcal{C}$
5. ***return*** $\beta' \leftarrow \mathcal{A}(\mathsf{ct}_\beta)$

Definition 18 *Let Π_{SKE} be an SKE with ciphertext space denoted by $\mathcal{C}$. We call Π_{SKE}* wIND$-CPA *secure if for every PPT adversary $\mathcal{A}$ it holds that*

$$\left|\Pr[\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{SKE}}}(\lambda, 1) = 1] - \Pr[\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{SKE}}}(\lambda, 0) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security game is defined as follows:

$\underline{\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{SKE}}}(\lambda,\beta):}$

1. $k \leftarrow \mathsf{Gen}(1^\lambda)$
2. $\mathsf{msg} \leftarrow \mathcal{A}(1^\lambda)$
3. $\mathsf{ct}_0 \leftarrow \mathsf{Enc}(k, \mathsf{msg})$
4. $\mathsf{ct}_1 \leftarrow \mathcal{C}$
5. ***return*** $\beta' \leftarrow \mathcal{A}(\mathsf{ct}_\beta)$

A.6 IND-CPA and (s)IND-CCA Security for IBAE

An IBAE satisfies IND-CPA security if for every PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, the following holds:

$$\left|\Pr[\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{IBAE}}}(\lambda,0)=1] - \Pr[\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{IBAE}}}(\lambda,1)=1]\right| \leq \mathsf{negl}(\lambda)$$

where the security game is defined as follows:

$\underline{\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{IBAE}}}(1^\lambda,\beta):}$

1. $(\mathsf{mpk}, \mathsf{msk}, \mathsf{ampk}, \mathsf{amsk}) \leftarrow \mathsf{aSetup}(1^\lambda)$
2. $(\mathsf{id}^*, \mathbf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1) \leftarrow \mathcal{A}^{\mathsf{OaKeyGen}(\cdot)}(\mathsf{ampk}, \mathsf{mpk})$
3. $\mathbf{act}_\beta = \mathsf{aEnc}(\mathsf{mpk}, \mathsf{ampk}, \mathsf{id}^*, \mathbf{msg}, \mathsf{amsg}_\beta)$
4. **return** $\beta' \leftarrow \mathcal{A}^{\mathsf{OaKeyGen}(\cdot)}(\mathbf{act}_\beta)$
   where $\mathsf{OaKeyGen}(\mathsf{id}) = \mathsf{aKeyGen}(\mathsf{amsk}, \mathsf{id})$
   and id^* is not queried to $\mathsf{OaKeyGen}(\cdot)$.

B Analysis of Our ElGamal 2-PKAEE

In this section, we show the correctness and prove Lemma 1, Theorem 1 related to Section 3.1.

Correctness: We now show the correctness of $\Pi_{\mathsf{aEl}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ based on the ElGamal PKE scheme. Let $\mathbf{act} = (\mathsf{act}_1, \mathsf{act}_2)$ be the output of $\mathsf{aEl.aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})$, where $\mathsf{act}_1 = (r_1, c_1)$ and $\mathsf{act}_2 = (r_2, c_2)$ are generated using randomness κ and $\mathsf{dk}_1^\kappa \cdot g^{\mathsf{amsg}}$. The double encryption key $\mathsf{dk} = (\mathsf{dk}_1, \mathsf{dk}_2) = (g^\alpha, g^{\alpha x})$. Therefore, we have $r_1 = g^\kappa$ and $r_2 = g^{\alpha\kappa + \mathsf{amsg}}$. The anamorphic decryption algorithm knows $\mathsf{tk} = \alpha$, the double decryption key corresponding to dk. Thus, it can compute $g^{\mathsf{amsg}} = r_2 \cdot (r_1)^{-\mathsf{tk}}$. Since $\widehat{\mathcal{M}}$ is polynomially bounded, the decryption algorithm recovers $\mathsf{amsg} \in \widehat{\mathcal{M}}$ by computing the discrete logarithm of g^{amsg}, which is a poly-time algorithm.

Lemma 1 The ElGamal PKE scheme is PKAE-compatible for the triplet $\Pi_{\mathsf{aEl}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ described in Figure 1 associated with an ElGamal 2-message encryption scheme $\Pi_{\mathsf{2El}} = (\mathsf{Enc}, \mathsf{Dec})$.

Proof. First, we observe that for the key pair $(\mathsf{pk}, \mathsf{sk})$ of a regular ElGamal PKE scheme, if we set tk as a uniformly random element α in $\mathbb{Z}_q$, and $\mathsf{dk} = (g^{\mathsf{tk}}, \mathsf{pk}^{\mathsf{tk}})$, then it satisfies the *on-the-fly key generation* property. Next, to see that Π_{aEl} satisfies the *ciphertext extractibility* property is pretty straightforward. The function F defined in Figure 1 extracts the first components $r_1 = g^{\kappa}$ and $r_2 = g^{\alpha\kappa+\mathsf{amsg}}$ from $\mathsf{act}_1 = (r_1, c_1), \mathsf{act}_2 = (r_2, c_2)$. Then, applying the regular ElGamal decryption algorithm $\mathsf{EL.Dec}$ with tk would simply recover amsg by computing $g^{\mathsf{amsg}} = r_2 \cdot (r_1)^{-\mathsf{tk}}$, which is poly-time as the anamorphic message space is poly-bounded. □

Theorem 1 Under the DDH assumption in a group $\mathbb{G}$, the ElGamal 2-message public key anamorphic encryption extension $\Pi_{\mathsf{aEl}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ described in Figure 1 satisfies anamorphic and $\mathsf{IND\text{-}CPA}$ security (Defined in Appendix A).

Proof. We prove the theorem in two parts.

Anamorphic security. We use a sequence of hybrid games $\mathsf{Game}\ 0, \mathsf{Game}\ (1, j)$ for $j \in [0, Q]$ to prove this theorem. In each hybrid game, we assume $\mathcal{D}$ is a PPT dictator and Q be the total number of queries made by $\mathcal{D}$ to the encryption oracle.

$\underline{\mathsf{Game}\ 0}$**:** This is the anamorphic $\mathsf{GAnam}^{\mathcal{D}}_{\Pi_{\mathsf{aEl}}}(\lambda)$. The challenger generates $(\mathsf{ask} = x, \mathsf{apk} = g^x, \mathsf{tk}, \mathsf{dk}) \leftarrow \mathsf{aEl.aGen}(1^\lambda)$, and sends the key pair $(\mathsf{apk}, \mathsf{ask})$ to $\mathcal{D}$. Then, $\mathcal{D}$ makes polynomial number of queries to the encryption oracle with a pair of messages $\{(\mathbf{msg}_i, \mathsf{amsg}_i)\}_{i\in[Q]}$. At the i-th query, $\mathcal{D}$ receives a ciphertext $\mathbf{act}_i$ computed as follows:

1. Parse $\mathbf{msg}_i = (\mathsf{msg}_{i_1}, \mathsf{msg}_{i_2})$
2. $\alpha, \kappa_1 \leftarrow \mathbb{Z}_q, \kappa_2 \coloneqq \alpha \cdot \kappa_1 + \mathsf{amsg}_i$.
3. $\mathsf{act}_{i_1} \coloneqq \mathsf{El.Enc}(\mathsf{apk}, \mathsf{msg}_{i_1}; \kappa_1) = (g^{\kappa_1}, g^{x\kappa_1} \cdot \mathsf{msg}_{i_1})$
4. $\mathsf{act}_{i_2} \coloneqq \mathsf{El.Enc}(\mathsf{apk}, \mathsf{msg}_{i_2}; \kappa_2) = (g^{\kappa_2}, g^{x\kappa_2} \cdot \mathsf{msg}_{i_2})$
5. **return** $\mathbf{act}_i \coloneqq (\mathsf{act}_{i_1}, \mathsf{act}_{i_2})$

$\underline{\mathsf{Game}\ (1, j)}$ **for** $j \in [0, Q]$**:** We define $\mathsf{Game}\ (1, 0)$ to be identical to $\mathsf{Game}\ 0$, and $\mathsf{Game}\ (1, j)$ is exactly the same as $\mathsf{Game}\ (1, 0)$ except that the first j (anamorphic) ciphertexts $\{\mathbf{act}_i\}_{i\in[1,j]}$ are computed as follows:

1. Parse $\mathbf{msg}_i = (\mathsf{msg}_{i_1}, \mathsf{msg}_{i_2})$
2. $\kappa_1, \kappa_2 \leftarrow \mathbb{Z}_q$.
3. $\mathsf{act}_{i_1} \coloneqq \mathsf{El.Enc}(\mathsf{apk}, \mathsf{msg}_{i_1}; \kappa_1) = (g^{\kappa_1}, g^{x\kappa_1} \cdot \mathsf{msg}_{i_1})$
4. $\mathsf{act}_{i_2} \coloneqq \mathsf{El.Enc}(\mathsf{apk}, \mathsf{msg}_{i_2}; \kappa_2) = (g^{\kappa_2}, g^{x\kappa_2} \cdot \mathsf{msg}_{i_2})$
5. **return** $\mathbf{act}_i \coloneqq (\mathsf{act}_{i_1}, \mathsf{act}_{i_2})$

We show that $\mathsf{Game}\ (1, j)$ is indistinguishable from $\mathsf{Game}\ (1, j-1)$ by the DDH assumption. The only difference between $\mathsf{Game}\ (1, j)$ and $\mathsf{Game}\ (1, j-1)$ is in the computation of κ_2. In particular, κ_2 is set as $(\alpha \cdot \kappa_1 + \mathsf{amsg}_i)$ in $\mathsf{Game}\ (1, j-1)$, whereas κ_2 is sampled uniformly random from $\mathbb{Z}_q$ in $\mathsf{Game}\ (1, j)$. Thus the challenger can simulate $\mathsf{Game}\ (1, j)$ with $(g^\alpha, g^{\kappa_1}, g^t)$ when $t = \alpha \cdot \kappa_1$, and

$\mathsf{Game}\ (1, j)$, when t is uniformly random in $\mathbb{Z}_q$. Then by the DDH assumption in $\mathbb{G}$, the following holds:

$$|\Pr[\alpha, \kappa_1 \leftarrow \mathbb{Z}_q : \mathcal{A}(g^\alpha, g^{\kappa_1}, g^{\alpha \cdot \kappa_1}) = 1] - \Pr[\alpha, \kappa_1, t \leftarrow \mathbb{Z}_q : \mathcal{A}(g^\alpha, g^{\kappa_1}, g^t) = 1]| \leq \mathsf{negl}(\lambda)$$

Since $t = \alpha \cdot \kappa_1$ is indistinguishable from uniformly random, $\kappa_2 = t + \mathsf{amsg}_i$ is statistically close to t. In other words, the hybrids $\mathsf{Game}\ (1, j)$ and $\mathsf{Game}\ (1, j-1)$ are indistinguishable.
Finally, we observe that $\mathsf{Game}\ (1, Q)$ is exactly the same as $\mathsf{GReal}^{\mathcal{D}}_{\Pi_{\mathsf{2El}}}(\lambda)$. This completes the proof.

IND-CPA security. We prove the theorem by designing a simulator that plays the role of the challenger in the security game.

We start with a contradiction that there exists a PPT adversary $\mathcal{A}$ that breaks the $\mathsf{IND\text{-}CPA}$ security of Π_{aEl} with a non-negligible probability. We then design a simulator $\mathcal{B}$ that given a DDH instance $(g^\kappa, g^\alpha, g^t)$ in the group $\mathbb{G}$, interacts with $\mathcal{A}$ to solve whether $t = \alpha\kappa$ or t is uniformly random in $\mathbb{Z}_q$.

$\mathcal{B}$ prepares the simulation by sampling a uniformly random x, and setting $\mathsf{ask} = x$, $\mathsf{apk} = g^x$, $\mathsf{dk} = (g^\alpha, g^{\alpha x})$, and sends $(\mathsf{apk}, \mathsf{dk})$ to $\mathcal{A}$. When $\mathcal{A}$ returns $(\mathbf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1)$, $\mathcal{B}$ calculates challenge ciphertext $\mathbf{act}^* = \mathsf{aEl.aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg}_\beta)$ for a uniformly random bit $\beta \in \{0, 1\}$, and sends it to $\mathcal{A}$. Finally $\mathcal{A}$ outputs its guess β', and $\mathcal{B}$ returns $(\beta == \beta')$. The simulation is formally specified as follows:

$\underline{\mathsf{Simulator}\ \mathcal{B}(1^\lambda, g^\kappa, g^\alpha, g^t)}$
1: $x \leftarrow \mathbb{Z}_q$
2: $\mathsf{ask} \coloneqq x, \mathsf{apk} \coloneqq g^x, \mathsf{dk} = (g^\alpha, g^{\alpha x})$
3: $(\mathbf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1) \leftarrow \mathcal{A}(\mathsf{apk}, \mathsf{dk})$
4: $\beta \leftarrow \{0, 1\}$
5: Parse $\mathbf{msg} = (\mathsf{msg}_1, \mathsf{msg}_2)$
6: Parse $\mathsf{dk} = (\mathsf{dk}_1, \mathsf{dk}_2)$
7: $\mathsf{act}^*_1 \coloneqq \mathsf{El.Enc}(\mathsf{apk}, \mathsf{msg}_1; \kappa) = (r_1 \coloneqq g^\kappa, c_1 \coloneqq \mathsf{apk}^\kappa \cdot \mathsf{msg}_1)$
8: $\mathsf{act}^*_2 \coloneqq \mathsf{El.Enc}(\mathsf{apk}, \mathsf{msg}_2; \kappa\alpha + \mathsf{amsg})$
$= (r_2 \coloneqq \mathsf{dk}_1^\kappa \cdot g^{\mathsf{amsg}_\beta}, c_2 \coloneqq \mathsf{dk}_2^\kappa \cdot \mathsf{apk}^{\mathsf{amsg}_\beta} \cdot \mathsf{msg}_2)$
$= (r_2 \coloneqq g^{t + \mathsf{amsg}_\beta}, c_2 \coloneqq (g^t)^x \cdot \mathsf{apk}^{\mathsf{amsg}_\beta} \cdot \mathsf{msg}_2)$
9: $\beta' \leftarrow \mathcal{A}(\mathbf{act}^* = (\mathsf{act}^*_1, \mathsf{act}^*_2))$
10: **return** $\beta == \beta'$

If $t = \alpha\kappa$, then the distribution of $(\mathsf{apk}, \mathsf{ask})$ and the challenge ciphertext $\mathbf{act}^*$ corresponds exactly to $\mathcal{A}$'s view as in the experiment $\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\mathsf{PKAE}}(\lambda, \beta)$. This is because even though x is chosen randomly during simulation, $(\mathsf{apk}, \mathsf{ask})$ is a valid pair, and hence $\mathbf{act}^*$ is a valid encryption of amsg_β. If $t \leftarrow \mathbb{Z}_q$, then with all but negligible probability act^* is neither an encryption of amsg_0 nor amsg_1. In fact, $t + \mathsf{amsg}_\beta$ is uniformly distributed in $\mathbb{G}$, and hence β is hidden from $\mathcal{A}$'s view. Therefore, if $\mathcal{A}$ breaks the $\mathsf{IND\text{-}CPA}$ security of Π_{aEl} with a non-negligible probability then $\mathcal{B}$ can also break the DDH assumption with a non-negligible probability. □

C Analysis of Our Dual-Regev 2-PKAEE

In this section, we show the correctness and prove Lemma 2, Theorem 2 related to Section 3.2.

Correctness: We now show the correctness of the $\Pi_{\mathsf{aDReg}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ based on the Dual-Regev PKE scheme. Let the encryption double key be $\mathsf{dk} = (\mathsf{dk}_1, \mathsf{dk}_2) = (\mathbf{A}, \widehat{\mathbf{b}} = \mathbf{A} \cdot \widehat{\mathbf{k}})$. Let $\mathbf{act} = (\mathsf{act}_1, \mathsf{act}_2)$ output by $\mathsf{aDReg.aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})$, where $\mathsf{act}_1 = (\mathbf{r}_1, c_1)$ and $\mathsf{act}_2 = (\mathbf{r}_2, c_2)$ generated using randomness $\mathbf{s}_1$ and $\mathbf{s}_2$ such that $\mathbf{r}_1 = \mathbf{s}_1^T \cdot \mathsf{dk}_1 + \mathbf{e}_1^T$ and $\mathbf{r}_2 = \mathbf{s}_2^T \cdot \mathsf{dk}_1 + \mathbf{e}_2^T$, where $\mathbf{s}_2$ is set as a solution to $\mathbf{a}_1 \cdot \mathbf{x} = \mathbf{s}_1^T \cdot \mathsf{dk}_2 + \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor - e_{2,1} \mod q$, where $\mathsf{dk}_1 = [\mathbf{a}_1, \ldots, \mathbf{a}_n]^T \in \mathbb{Z}_q^{n \times m}$, $\mathbf{e}_2^T = [e_{2,1}, \ldots, e_{2,m}] \in \chi^m$ and $\widehat{y} \in \chi$. Then, observe that the first element of the vector $\mathbf{r}_2^T$ is $\mathbf{s}_1^T \cdot \mathsf{dk}_2 + \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor \mod q$. The anamorphic decryption algorithm knows $\mathsf{tk} = \widehat{\mathbf{k}}$, the decryption double key corresponding to dk. It calls the function $F(\mathbf{act})$ which extracts the first element of $\mathbf{r}_2^T$ using a projection function and assigns the value to $\widehat{c}$. Thus it can compute amsg from $\widehat{\mathbf{r}} = \mathbf{r}_1 = \mathbf{s}_1^T \cdot \mathsf{dk}_1 + \mathbf{e}_1^T$ and $\widehat{c} = \mathbf{s}_1^T \cdot \mathsf{dk}_2 + \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor \mod q$, by returning $\mathsf{amsg} = 0$ if $|\widehat{c} - \widehat{\mathbf{r}} \cdot \mathsf{tk}| \leq \frac{q}{4}$, and $\mathsf{amsg} = 1$ otherwise. Note that, $|\widehat{c} - \widehat{\mathbf{r}} \cdot \mathsf{tk}|$ simply reduces to $\mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor$ plus an LWE noise $\widehat{y} - \mathbf{e}_2^T \cdot \widehat{\mathbf{k}} \mod q$, whose norm is smaller than $\frac{q}{5}$, except with a negligible probability in λ.

Lemma 2 The Dual-Regev PKE scheme is PKAE-compatible for the triplet $\Pi_{\mathsf{aDReg}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ described in Figure 2 associated with a Dual-Regev 2-message encryption scheme $\Pi_{\mathsf{2DReg}} = (\mathsf{Enc}, \mathsf{Dec})$.

Proof. First, observe that $\mathsf{aDReg.aGen}(1^\lambda)$ outputs $\mathsf{ask} = \mathbf{k}$, $\mathsf{apk} = (\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{k})$, $\mathsf{tk} = \widehat{\mathbf{k}}$ and $\mathsf{dk} = (\mathbf{A}, \widehat{\mathbf{b}} = \mathbf{A} \cdot \widehat{\mathbf{k}})$, where $\mathbf{k}$ and $\widehat{\mathbf{k}}$ are sampled uniformly at random from $\{0,1\}^m$. Therefore, the key pair $(\mathsf{apk}, \mathsf{ask})$ has identical distributions with $\{(\mathsf{pk}, \mathsf{sk}) : \mathsf{sk} = \widetilde{\mathbf{k}}, \mathsf{pk} = (\mathbf{A}, \widetilde{\mathbf{b}} = \mathbf{A} \cdot \widetilde{\mathbf{k}})$ s.t. $\widetilde{\mathbf{k}} \leftarrow \{0,1\}^m\}$, which is essentially the distribution of secret and public keys of a Dual-Regev PKE scheme. Hence, it satisfies *on-the-fly double key generation* property. Next, observe that the function F defined in Figure 2 takes as input $\mathbf{act} \leftarrow \mathsf{aDReg.aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})$, and parses it as $\mathsf{act}_1 = (\mathbf{r}_1, c_1)$ and $\mathsf{act}_2 = (\mathbf{r}_2, c_2)$. It then extracts $\mathbf{r}_1$ (call it $\widehat{\mathbf{r}}$) and the first component of $\mathbf{r}_2^T$ (call it $\widehat{c}$). Note that $(\widehat{\mathbf{r}}, \widehat{c})$ is a valid Dual-Regev ciphertext as $\widehat{\mathbf{r}} = \mathbf{s}_1^T \cdot \mathbf{A} + \mathbf{e}_1^T$ and $\widehat{c} = \mathbf{s}_1^T \cdot \widehat{\mathbf{b}} + \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor \mod q$. Thus, amsg can be extracted by executing $\mathsf{DReg.Dec}$ using $\mathsf{tk} = \widehat{\mathbf{k}}$, which implies that the PKAE satisfies the *ciphertext extractibility* property. Hence we conclude that the Dual-Regev PKE is PKAE-compatible. □

Theorem 2 Let us consider $m \geq 2n \log q$, $r = \omega(\sqrt{\log m})$, $q \geq 5r(m+1)$, $\alpha \leq 1/(r\sqrt{m+1} \cdot \omega(\sqrt{\log n}))$ and $\chi = \Psi_\alpha$. Then, under the (n, q, χ)-DLWE assumption of dimension m, the Dual-Regev 2-message public key anamorphic encryption extension $\Pi_{\mathsf{aDReg}} = (\mathsf{aGen}, \mathsf{aEnc}, \mathsf{aDec})$ described in Figure 2 satisfies anamorphic and IND-CPA security (Defined in Appendix A).

Proof. We prove the theorem in two parts.

Anamoprhic security. We use a sequence of hybrid games $\mathsf{Game}\ 0, \mathsf{Game}\ (1,j)$ for $j \in [0,Q]$ to prove this theorem. In each hybrid game, we assume $\mathcal{D}$ is a PPT dictator and Q be the total number of queries made by the dictator to the encryption oracle.

$\underline{\mathsf{Game}\ 0}$**:** This is the anamorphic $\mathsf{GAnam}^{\mathcal{D}}_{\Pi_{\mathsf{aDReg}}}(\lambda)$. The challenger generates $(\mathsf{ask} = \mathbf{k}, \mathsf{apk} = (\mathbf{A}, \mathbf{b} = \mathbf{A}\cdot\mathbf{k}), \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aDReg.aGen}(1^\lambda)$, and sends the key pair $(\mathsf{apk}, \mathsf{ask})$ to $\mathcal{D}$. Then $\mathcal{D}$ makes polynomial number of queries to the encryption oracle with a pair of messages $\{(\mathbf{msg}_i, \mathsf{amsg}_i)\}_{i\in[Q]}$. At the i-th query, the dictator receives a ciphertext $\mathbf{act}_i$ computed as follows:

1. Parse $\mathbf{msg}_i = (\mathsf{msg}_{i_1}, \mathsf{msg}_{i_2})$
2. $\widehat{\mathbf{k}} \leftarrow \{0,1\}^m$, $\mathbf{s}_1 \leftarrow \mathbb{Z}_q^n, \mathbf{e}_1^T, \mathbf{e}_2^T \leftarrow \chi^m, \widehat{y} \leftarrow \chi$
3. $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_n]^T \in \mathbb{Z}_q^{n\times m}, \mathbf{e}_2^T = [e_{2,1}, \ldots, e_{2,m}] \in \chi^m$
4. $\widehat{\mathbf{b}} \coloneqq \mathbf{A}\cdot\widehat{\mathbf{k}}$
5. **for** $i \in \{1, \ldots, m-1\}$, $x_i \leftarrow \mathbb{Z}_q$
6. **end for**
7. $x_m = (\sum_{i=1}^{m-1} x_i a_i + \mathbf{s}_1^T\cdot\widehat{\mathbf{b}} + \widehat{y} + \mathsf{amsg}_i \cdot \lfloor \frac{q}{2} \rfloor - e_{2,1})\cdot a_m^{-1} \mod q$
8. $\mathbf{x} = (x_1, \ldots, x_m)$
9. $\mathbf{s}_2 \coloneqq \mathbf{x}$
10. $\mathsf{act}_{i_1} \coloneqq \mathsf{DReg.Enc}(\mathsf{apk}, \mathsf{msg}_{i_1}; \mathbf{s}_1, \mathbf{e}_1^T)$
$= (\mathbf{s}_1^T\cdot\mathbf{A} + \mathbf{e}_1^T, \mathbf{s}_1^T\cdot\mathbf{b} + y_1 + \mathsf{msg}_{i_1}\cdot\lfloor\frac{q}{2}\rfloor \mod q)$
11. $\mathsf{act}_{i_2} \coloneqq \mathsf{DReg.Enc}(\mathsf{apk}, \mathsf{msg}_{i_2}; \mathbf{s}_2, \mathbf{e}_2^T)$
$= (\mathbf{s}_2^T\cdot\mathbf{A} + \mathbf{e}_2^T, \mathbf{s}_2^T\cdot\mathbf{b} + y_2 + \mathsf{msg}_{i_2}\cdot\lfloor\frac{q}{2}\rfloor \mod q)$
12. **return** $\mathbf{act}_i \coloneqq (\mathsf{act}_{i_1}, \mathsf{act}_{i_2})$

$\underline{\mathsf{Game}\ (1,j)}$ **for** $j \in [0,Q]$**:** We define $\mathsf{Game}\ (1,0)$ to be identical to $\mathsf{Game}\ 0$ and $\mathsf{Game}\ (1,j)$ is exactly the same as $\mathsf{Game}\ (1,0)$ except that the first j (anamorphic) ciphertexts $\{\mathbf{act}_i\}_{i\in[1,j]}$ are computed as follows:

1. Parse $\mathbf{msg}_i = (\mathsf{msg}_{i_1}, \mathsf{msg}_{i_2})$
2. $\mathbf{s}_1, \mathbf{s}_2 \leftarrow \mathbb{Z}_q^n, \mathbf{e}_1^T, \mathbf{e}_2^T \leftarrow \chi^m$
3. $\mathsf{msg}_{i_1} \leftarrow \{0,1\}$, $\mathsf{msg}_{i_2} \coloneqq \mathsf{msg}_i \oplus \mathsf{msg}_{i_1}$
4. $\mathsf{act}_{i_1} \coloneqq \mathsf{DReg.Enc}(\mathsf{apk}, \mathsf{msg}_{i_1}; \mathbf{s}_1, \mathbf{e}_1^T)$
$= (\mathbf{s}_1^T\cdot\mathbf{A} + \mathbf{e}_1^T, \mathbf{s}_1^T\cdot\mathbf{b} + y_1 + \mathsf{msg}_{i_1}\cdot\lfloor\frac{q}{2}\rfloor \mod q)$
5. $\mathsf{act}_{i_2} \coloneqq \mathsf{DReg.Enc}(\mathsf{apk}, \mathsf{msg}_{i_2}; \mathbf{s}_2, \mathbf{e}_2^T)$
$= (\mathbf{s}_2^T\cdot\mathbf{A} + \mathbf{e}_2^T, \mathbf{s}_2^T\cdot\mathbf{b} + y_2 + \mathsf{msg}_{i_2}\cdot\lfloor\frac{q}{2}\rfloor \mod q)$
6. **return** $\mathbf{act}_i \coloneqq (\mathsf{act}_{i_1}, \mathsf{act}_{i_2})$

We show that $\mathsf{Game}\ (1,j)$ is indistinguishable from $\mathsf{Game}\ (1,j-1)$ by the DLWE assumption. The only difference between $\mathsf{Game}\ (1,j)$ and $\mathsf{Game}\ (1,j-1)$ is in the computation of the vector $\mathbf{s}_2$. In $\mathsf{Game}\ (1,j)$, $\mathbf{s}_2$ is sampled uniformly at random from $\mathbb{Z}_q^n$, whereas in $\mathsf{Game}\ (1,j-1)$, $\mathbf{s}_2$ is set as $\mathbf{x} = (x_1, \ldots, x_m)$ such that the first $(m-1)$ elements of the sequence are uniformly random values in $\mathbb{Z}_q$, and $x_m = (\sum_{i=1}^{m-1} x_i a_i + \mathbf{s}_1^T\cdot\widehat{\mathbf{b}} + \widehat{y} + \mathsf{amsg}_i\cdot\lfloor\frac{q}{2}\rfloor - e_{2,1})\cdot a_m^{-1} \mod q$, where the a_i's are uniformly random elements of $\mathbb{Z}_q^m$. Note that, by the DLWE assumption, $(\mathbf{s}_1^T\cdot\widehat{\mathbf{b}} + \widehat{y}) \mod q$ is indistinguishable from uniformly random in $\mathbb{Z}_q$. Since $(\mathbf{s}_1^T\cdot\widehat{\mathbf{b}} + \widehat{y})$ appears uniformly random, and is not present anywhere

in the game, x_m appears uniformly random in $\mathbb{Z}_q$. In other words, the hybrids $\mathsf{Game}\ (1, j)$ and $\mathsf{Game}\ (1, j-1)$ are indistinguishable.
Finally, we observe that $\mathsf{Game}\ (1, Q)$ is exactly the same as $\mathsf{GReal}^{\mathcal{D}}_{\Pi_{\mathsf{2DReg}}}(\lambda)$. This completes the proof.

IND-CPA security. We prove the theorem by showing that an adversary's view in the IND-CPA game is computationally indistinguishable from uniformly random. For the sake of contradiction, suppose there exists a PPT adversary $\mathcal{A}$ that breaks down IND-CPA security of Π_{aDReg} with a non-negligible probability σ. Then we design a simulator $\mathcal{B}$ that breaks the DLWE assumption. More precisely, given $(\mathbf{A}, \widehat{\mathbf{b}})$ with $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\widehat{\mathbf{b}} \in \mathbb{Z}_q^n$, $\mathcal{B}$ determines whether $\widehat{\mathbf{b}}$ is chosen uniformly at random or $\widehat{\mathbf{b}} = \mathbf{A} \cdot \widehat{\mathbf{k}}$, where $\widehat{\mathbf{k}} \in \{0,1\}^m$, by interacting with $\mathcal{A}$.

To prepare the simulation, $\mathcal{B}$ randomly samples vector $\mathbf{k} \in \{0,1\}^m$, and sets $\mathsf{ask} \coloneqq \mathbf{k}, \mathsf{apk} \coloneqq (\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{k}), \mathsf{dk} = (\mathbf{A}, \widehat{\mathbf{b}})$. On feeding $\mathcal{A}$ with $(\mathsf{apk}, \mathsf{dk})$, it returns $(\mathbf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1)$, and for a uniformly random bit $\beta \in \{0,1\}$, $\mathcal{B}$ calculates the challenge ciphertext $\mathbf{act}^* = \mathsf{aDReg.aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg}_\beta)$. Finally, when the adversary returns β', $\mathcal{B}$ outputs $(\beta == \beta')$. The simulation is formally specified as follows:

$\underline{\mathsf{Simulator}\ \mathcal{B}(1^\lambda, \mathbf{A}, \widehat{\mathbf{b}})}$
1: $\mathbf{k} \leftarrow \{0,1\}^m$
2: $\mathsf{ask} \coloneqq \mathbf{k}, \mathsf{apk} \coloneqq (\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{k}), \mathsf{dk} = (\mathbf{A}, \widehat{\mathbf{b}})$
3: $(\mathbf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1) \leftarrow \mathcal{A}(\mathsf{apk}, \mathsf{dk})$
4: $\beta \leftarrow \{0,1\}$
4: Parse $\mathsf{dk} = (\mathsf{dk}_1, \mathsf{dk}_2)$
5: Parse $\mathbf{msg} = (\mathsf{msg}_1, \mathsf{msg}_2)$
6: $\mathbf{s}_1 \leftarrow \mathbb{Z}_q^n, \mathbf{e}_1^T, \mathbf{e}_2^T \leftarrow \chi^m, \widehat{y} \leftarrow \chi$
7: $\mathsf{dk}_1 = [\mathbf{a}_1, \ldots, \mathbf{a}_n]^T \in \mathbb{Z}_q^{n \times m}, \mathbf{e}_2^T = [e_{2,1}, \ldots, e_{2,m}] \in \chi^m$
8: **for** $i \in \{1, \ldots, m-1\}$, $x_i \leftarrow \mathbb{Z}_q$
9: **end for**
10: $x_m = (\sum_{i=1}^{m-1} x_i a_i + \mathbf{s}_1^T \cdot \mathsf{dk}_2 + \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor - e_{2,1}) \cdot a_m^{-1} \mod q$
11: $\mathbf{x} = (x_1, \ldots, x_m)$
12: $\mathbf{s}_2 \coloneqq \mathbf{x}$
13: $\mathsf{act}_1^* \coloneqq \mathsf{DReg.Enc}(\mathsf{apk}, \mathsf{msg}_1; \mathbf{s}_1, \mathbf{e}_1^T) = (\mathbf{r}_1, c_1)$
// $\mathbf{r}_1 \coloneqq \mathbf{s}_1^T \cdot \mathsf{dk}_1 + \mathbf{e}_1^T$, $c_1 \coloneqq \mathbf{s}_1^T \cdot \mathbf{b} + y_1 + \mathsf{msg}_1 \cdot \lfloor \frac{q}{2} \rfloor \mod q$
14: $\mathsf{act}_2^* \coloneqq \mathsf{DReg.Enc}(\mathsf{pk}, \mathsf{msg}_2; \mathbf{s}_2, \mathbf{e}_2^T) = (\mathbf{r}_2, c_2)$
// $\mathbf{r}_2 \coloneqq \mathbf{s}_2^T \cdot \mathsf{dk}_1 + \mathbf{e}_2^T$, $c_2 \coloneqq \mathbf{s}_2^T \cdot \mathbf{b} + y_2 + \mathsf{msg}_2 \cdot \lfloor \frac{q}{2} \rfloor \mod q$
15: $\beta' \leftarrow \mathcal{A}(\mathbf{act}^* = (\mathsf{act}_1^*, \mathsf{act}_2^*))$
16: **return** $\beta == \beta'$

When $\widehat{\mathbf{b}} = \mathbf{A} \cdot \widehat{\mathbf{k}}$, then the distribution of $(\mathsf{ask}, \mathsf{apk})$ and the challenge ciphertext $\mathbf{act}^*$ corresponds to $\mathcal{A}$'s view as in the experiment $\mathsf{GIND\text{-}CPA}^{\mathcal{A}}_{\mathsf{PKAE}}(\lambda, \beta)$. This follows from the anamorphic key-pair being perfectly valid, which makes $\mathbf{act}^*$ a valid encryption of amsg_β under $\mathsf{dk} = (\mathbf{A}, \widehat{\mathbf{b}})$. If $\widehat{\mathbf{b}}$ is uniformly random in $\mathbb{Z}_q^n$, then (in line 10) $\mathbf{s}_1^T \cdot \widehat{\mathbf{b}} + \widehat{y}$ becomes a DLWE sample which is close to uniformly

random over $\mathbb{Z}_q$ assuming the LWE assumption. Hence, the challenge ciphertext $\mathbf{act}^*$ hide amsg_β. Thus, $\mathcal{A}$ has no information on the encrypted anamorphic message. Therefore, if $\mathcal{A}$ breaks the IND-CPA security of Π_{aDReg} with a non-negligible probability then $\mathcal{B}$ can also break the DLWE assumption with a non-negligible probability. □

D Analysis of Our FO-Based PKAEE

Theorem 3 Let Π_{FO} be an IND-CCA secure FO-based PKE where the underlying PKE Π_{PKE} is IND$-CPA secure and has an exponentially large message space $\mathcal{M}_{\mathsf{PKE}}$ (i.e., $|\mathcal{M}_{\mathsf{PKE}}| \geq 2^{\mathsf{poly}(\lambda)}$ for non-constant poly), and the underlying SKE Π_{SKE} is wIND$-CPA secure. Furthermore, let $(\mathsf{Encode}, \mathsf{Decode})$ be a pseudo-random $(\mathcal{C}_{\mathsf{FO}}, \mathcal{M}^{\mu}_{\mathsf{PKE}})$-encoding, where $\mathcal{C}_{\mathsf{FO}}$ denotes the ciphertext space of Π_{FO}. Then Π_{FOAE} as defined in Figure 3 is a μ-message anamorphic extension satisfying anamorphic security and sIND-RCCA security in the ROM.

Proof. We prove this theorem in two parts.

Anamorphic Security. In the following sequence of hybrid games let $\mathcal{D}$ be a PPT dictator and Q its total number of oracle queries.

- Game 0**:** This is the real game $\mathsf{GReal}^{\mathcal{D}}_{\mathsf{FO}}(\lambda)$ in the anamorphic security definition, where the challenger generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$ and handles encryption queries $\mathsf{OEnc}(\mathbf{msg}, \mathsf{amsg})$ of the dictator by computing $(\mathsf{FO.Enc}(\mathsf{pk}, \mathsf{msg}_i)$ for $1 \leq i \leq \mu$. The adversary also has access to the random oracles H and G.
- Game 1**:** In this game the challenger generates the key pair by computing $(\mathsf{apk}, \mathsf{ask}, \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{FOAE.aGen}(1^\lambda)$ and provides $(\mathsf{apk}, \mathsf{ask})$ as input to $\mathcal{D}$.
- Game $(2, j)$ **for** $j \in [0, Q]$**:** In this series of games we slightly modify Game 1. We define Game $(2, 0)$ to be identical to Game 1. In Game $(2, j)$, the challenger handles the first j encryption oracle calls as follows: In such a call, the oracle first randomly chooses an elements c from $\mathcal{C}_{\mathsf{FO}}$ and applies $\mathsf{Encode}(c)$ to obtain $(r_1, \ldots, r_\mu)$. Then the latter values are used as encryption randomness: $(\mathsf{FO.Enc}(\mathsf{apk}, \mathsf{msg}_i; r_i)$ for $1 \leq i \leq \mu$. The remaining oracle calls are left untouched in comparison to Game $(2, 0)$. Note that in Game $(2, Q)$ we have modified all encryption computations in this way.
- Game $(3, j)$ **for** $j \in [0, Q]$**:** In this series of games, we modify Game $(2, Q)$. We define Game $(3, 0)$ to be identical to Game $(2, Q)$. In Game $(3, j)$, the challenger handles the first j encryption oracle calls as follows: In such a call, instead of choosing c fully at random from $\mathcal{C}_{\mathsf{FO}}$, we replace the components representing the ciphertext of PKE by an actual encryption of a random message r'. That means, we compute $c = (c_{\mathsf{PKE}}, c_{\mathsf{SKE}}) \in \mathcal{C}_{\mathsf{FO}} = \mathcal{C}_{\mathsf{PKE}} \times \mathcal{C}_{\mathsf{SKE}}$ as $c = (\mathsf{PKE.Enc}(\mathsf{dk}, r'), c_{\mathsf{SKE}})$, where $r' \leftarrow \mathcal{M}_{\mathsf{PKE}}$ and $c_{\mathsf{SKE}} \leftarrow \mathcal{C}_{\mathsf{SKE}}$.
- Game $(4, j)$ **for** $j \in [0, Q]$**:** In this series of games, we modify Game $(3, Q)$. We define Game $(4, 0)$ to be identical to Game $(3, Q)$. In Game $(4, j)$, the challenger handles the first j encryption oracle calls as follows: In such a call, we now also replace the components of c representing the ciphertext part of

SKE. That means, we compute c as $c = (\mathsf{PKE.Enc}(\mathsf{dk}, r'), \mathsf{SKE.Enc}(k, \mathsf{amsg}))$, where $r' \leftarrow \mathcal{M}_{\mathsf{PKE}}$ and $k \leftarrow \mathsf{SKE.Gen}(1^\lambda)$.

Game 5**:** In this game, in all oracle calls when computing c, the encryption randomness of PKE and the key k of SKE is generated using the random oracles H and G as in the FO construction, i.e., $c = (\mathsf{PKE.Enc}(\mathsf{dk}, r'; H(r', \mathsf{amsg})), \mathsf{SKE.Enc}(G(r'), \mathsf{amsg}))$, $r' \leftarrow \mathcal{M}_{\mathsf{PKE}}$.

Game 6**:** This is the anamorphic game $\mathsf{GAnam}^{\mathcal{D}}_{\mathsf{FOAE}}(\lambda)$, where the challenger generates $(\mathsf{apk}, \mathsf{ask}, \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{FOAE.aGen}(1^\lambda)$ and handles encryption queries $\mathsf{OaEnc}(\mathbf{msg}, \mathsf{amsg})$ of the dictator by computing $\mathsf{FOAE.aEnc}(\mathsf{apk}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})$.

Game 0 and Game 1 are identical as $(\mathsf{ask}, \mathsf{apk})$ are generated in aGen by FO.Gen. Game $(2, 0)$ is indistinguishable from Game $(2, Q)$, as a distinguisher $\mathcal{D}$ for Game $(2, j)$ and Game $(2, j+1)$ (for random $j \in \{0, \ldots, Q-1\}$) can be turned in to an adversary against the pseudo-randomness of the encoding. Similarly, Game $(3, 0)$ is indistinguishable from Game $(3, Q)$, as from a distinguisher $\mathcal{D}$ for Game $(3, j)$ and Game $(3, j+1)$ (for random j) an IND$-CPA adversary against Π_{PKE} could be constructed. By a similar hybrid argument, Game $(4, 0)$ is indistinguishable from Game $(4, Q)$, as a distinguisher $\mathcal{D}$ for Game $(4, j)$ and Game $(4, j+1)$ (for random j) leads to an wIND$-CPA adversary against Π_{SKE}. Next, Game $(4, Q)$ and Game 5 are computationally distinguishable: The random oracle call $H(r', \mathsf{amsg})$, for *uniformly chosen* r', will not result in a freshly chosen value if (r', amsg) has been input to H before. If there are Q_H queries to H in total, this could happen with probability at most $\frac{Q_H}{|\mathcal{M}_{\mathsf{PKE}}|}$ for each call to OEnc and thus with probability $\frac{Q \cdot Q_H}{|\mathcal{M}_{\mathsf{PKE}}|}$ for all calls. Note that our assumption is that the message space is of exponential size. A similar argument holds for random oracle calls $G(r')$. Finally, Game 5 and Game 6 are identical.

sIND-RCCA **Security.** This follows immediately from the IND-CCA security of Π_{FO}. The reduction algorithm $\mathcal{B}$ is given dk as input from its IND-CCA challenger and has access to a $\mathsf{Dec}(\mathsf{tk}', \cdot)$ oracle. It can generate $(\mathsf{apk}, \mathsf{ask}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$ itself, and thus provide $(\mathsf{apk}, \mathsf{dk}, \mathsf{ask})$ as input to the sIND-RCCA adversary $\mathcal{A}$.

Upon receiving $(\mathsf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1)$ from $\mathcal{A}$, $\mathcal{B}$ provides $(\mathsf{amsg}_0, \mathsf{amsg}_1)$ to its own IND-CCA challenger to obtain c^*. The ciphertext c^* is then transformed into a challenge ciphertext vector act^* as specified in aEnc in Figure 3.

To simulate the $\mathsf{aDec}(\mathsf{tk}, \cdot)$ oracle, where $\mathsf{tk} = (\mathsf{ask}, \mathsf{tk}')$, $\mathcal{B}$ uses the self-chosen ask to decrypt the public key components of act in order to obtain $(r_1, \ldots, r_\mu)$ (cf. aDec in Figure 3). Then it decodes them to obtain c. If $c = c^*$, $\mathcal{B}$ rejects, i.e., it sends $\perp$ as oracle response to $\mathcal{A}$. Otherwise it queries its own $\mathsf{Dec}(\mathsf{tk}', \cdot)$ to decrypt c. If this decryption yields amsg_1 or amsg_2, $\mathcal{B}$ sends $\perp$ as oracle response to $\mathcal{A}$. Note that this simulation of an sIND-RCCA aDec oracle is perfect: The only thing which is done differently by the simulation is that it checks if $c = c^*$ and returns $\perp$ if this equation holds. However, by correctness of Π_{FO} it follows that such a c would have been decrypted to amsg_1 or amsg_2 and thus the corresponding act would have also been rejected by the real aDec oracle.

Finally $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

E Analysis of Our Boneh-Franklin 2-IBAEE

In this section, we show the correctness and prove Lemma 3, Theorem 1 related to Section 4.1.

Correctness: We now show the correctness of the $\Pi_{\mathsf{aBF}} = (\mathsf{aSetup}, \mathsf{aKeyGen}, \mathsf{aEnc}, \mathsf{aDec})$ scheme based on the Boneh-Franklin IBE scheme. Let $\mathbf{act} = (\mathsf{act}_1\mathsf{act}_2)$ be the output of $\mathsf{aEnc}(\mathsf{ampk}, \mathsf{dmpk}, \mathsf{id}, \mathbf{msg}, \mathsf{amsg})$, where $\mathsf{act}_1 = (r_1, c_1)$ and $\mathsf{act}_2 = (r_2, c_2)$. Here, $r_1 = P^{\kappa_1}$ and $r_2 = P^{\kappa_2}$ are generated using a uniformly random κ_1 in $\mathbb{Z}_q^*$, while κ_2 is such that $\mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2}) = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1})$, where $\widehat{g}_{\mathsf{id}} \coloneqq e(Q_{\mathsf{id}}, \mathsf{dmpk} = P^{\alpha})$ and $g_{\mathsf{id}} \coloneqq e(Q_{\mathsf{id}}, P_{\mathsf{pub}} = P^{s})$. The anamorphic decryption algorithm $\mathsf{aBF.aDec}$ knows both the anamorphic private key $\mathsf{ask}_{\mathsf{id}} = Q_{\mathsf{id}}^{s}$ and the double private key $\mathsf{dsk}_{\mathsf{id}} = Q_{\mathsf{id}}^{\alpha}$. It first calls $F(\mathsf{ask}_{\mathsf{id}}, \mathbf{act})$, which extracts $\mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2})$ by computing $\mathsf{H}_2(e(\mathsf{ask}_{\mathsf{id}}, r_2))$. Note that $e(\mathsf{ask}_{\mathsf{id}}, r_2) = e(Q_{\mathsf{id}}^{s}, P^{\kappa_2}) = e(Q_{\mathsf{id}}, P^{s})^{\kappa_2} = (g_{\mathsf{id}}^{\kappa_2})$. Finally, amsg is recovered by executing $\mathsf{BF.Dec}$ on $\widehat{r} = r_1$ and $\widehat{c} = \mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2}) = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1})$. To see why it works, observe that $\mathsf{BF.Dec}(\mathsf{dsk}_{\mathsf{id}}, F(\mathsf{ask}_{\mathsf{id}}, \mathbf{act}))$ computes $\widehat{c} \oplus \mathsf{H}_2(e(\mathsf{dsk}_{\mathsf{id}}, \widehat{r})) = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1}) \oplus \mathsf{H}_2(e(Q_{\mathsf{id}}^{\alpha}, P^{\kappa_1})) = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1}) \oplus \mathsf{H}_2(e(Q_{\mathsf{id}}, P^{\alpha})^{\kappa_1}) = \mathsf{amsg}$.

Lemma 3 The Boneh-Franklin identity-based encryption scheme is IBAE-compatible for the quadruplet $\Pi_{\mathsf{aBF}} = (\mathsf{aSetup}, \mathsf{aKeyGen}, \mathsf{aEnc}, \mathsf{aDec})$ described in Figure 4 associated with a Boneh-Franklin 2-message identity-based encryption scheme $\Pi_{\mathsf{2BF}} = (\mathsf{Enc}, \mathsf{Dec})$.

Proof. First, observe that $\mathsf{aBF.aSetup}(1^{\lambda})$ outputs $\mathsf{amsk} = s$, $\mathsf{ampk} = (\mathbb{G}, \mathbb{G}_T, P, e, q, P_{\mathsf{pub}} = P^{s}, \mathsf{H}_1, \mathsf{H}_2)$, $\mathsf{dmsk} = \alpha$ and $\mathsf{dmpk} = P^{\alpha}$, where s and α are uniformly random in $\mathbb{Z}_q^*$. Therefore, the key pair $(\mathsf{ampk}, \mathsf{amsk})$ has identical distributions with $\{(\mathsf{mpk}, \mathsf{msk}) : \mathsf{msk} = \widetilde{s}, \mathsf{mpk} = (\mathbb{G}', \mathbb{G}'_T, P', e', q', P'_{\mathsf{pub}} = P^{\widetilde{s}}, \mathsf{H}'_1, \mathsf{H}'_2)$ such that $\widetilde{s} \leftarrow \mathbb{Z}_q^*$, which is essentially the distribution of master secret and master public keys of a Boneh-Franklin IBE scheme. Hence, it satisfies *on-the-fly double key generation* property. Next, observe that the function F defined in Figure 4 takes as input $\mathbf{act} \leftarrow \mathsf{aBF.aEnc}(\mathsf{ampk}, \mathsf{dmpk}, \mathsf{id}, \mathbf{msg}, \mathsf{amsg})$, and parses it as $\mathsf{act}_1 = (r_1, c_1)$ and $\mathsf{act}_2 = (r_2, c_2)$. It then extracts r_1 (call it $\widehat{r}$) and the component $\mathsf{H}_2(g_{\mathsf{id}}^{\kappa_2}) = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1})$(call it $\widehat{c}$). Note that $(\widehat{\mathbf{r}}, \widehat{c})$ is a valid Boneh-Franklin ciphertext as $\widehat{r} = r_1$ and $\widehat{c} = \mathsf{amsg} \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}}^{\kappa_1})$. Thus, amsg can be extracted by executing $\mathsf{BF.Dec}$ using $\mathsf{dsk}_{\mathsf{id}} = Q_{\mathsf{id}}^{\alpha}$, which implies that the IBAE satisfies the *ciphertext extractibility* property. Hence we conclude that the Boneh-Franklin IBE is IBAE-compatible. □

Theorem 4 Under the DBDH assumption for $(\mathbb{G}, \mathbb{G}_T, P, e, q)$, the Boneh-Franklin 2-message identity-based anamorphic encryption extension Π_{aBF} described in Figure 4 satisfies anamorphic and $\mathsf{IND\text{-}CPA}$ security (Definition 5 and Appendix A.6).

Proof. We prove the theorem in two parts.

Anamorphic security. We use a sequence of hybrid games $\mathsf{Game}\ 0, \mathsf{Game}\ (1,j)$ for $j \in [0,Q]$ to prove this theorem. In each hybrid game, we assume $\mathcal{D}$ is a PPT dictator and Q be the total number of queries made by $\mathcal{D}$ to the encryption oracle.

$\underline{\mathsf{Game}\ 0}$**:** This is the anamorphic $\mathsf{GAnam}^{\mathcal{D}}_{\Pi_{2\mathsf{BF}}}(\lambda)$. The challenger generates $(\mathsf{dmsk} = \alpha, \mathsf{dmpk} = P^{\alpha}) \leftarrow \mathsf{aBF.aSetup}(1^{\lambda})$, and sends the key pair $(\mathsf{ampk}, \mathsf{amsk})$ to $\mathcal{D}$. Then, $\mathcal{D}$ makes polynomial number of queries to the encryption oracle with a tuple $\{(\mathsf{id}_i, \mathbf{msg}_i, \mathsf{amsg}_i)\}_{i\in[Q]}$. At the i-th query, $\mathcal{D}$ receives a ciphertext $\mathbf{act}_i$ computed as follows:

1. Parse $\mathbf{msg}_i = (\mathsf{msg}_{i_1}, \mathsf{msg}_{i_2})$
2. $Q_{\mathsf{id}_i} \coloneqq \mathsf{H}_1(\mathsf{id}_i)$
3. $\widehat{g}_{\mathsf{id}_i} \coloneqq e(Q_{\mathsf{id}_i}, \mathsf{dmpk})$, $g_{\mathsf{id}_i} \coloneqq e(Q_{\mathsf{id}_i}, P_{\mathsf{pub}})$
4. $\kappa_1, \kappa_2 \leftarrow \mathbb{Z}_q^*$ s.t. $\mathsf{H}_2(g_{\mathsf{id}_i}^{\kappa_2}) = \mathsf{amsg}_i \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}_i}^{\kappa_1})$
5. $\mathsf{act}_1 \coloneqq \mathsf{BF.Enc}(\mathsf{ampk}, \mathsf{msg}_{i_1}, \mathsf{id}_i; \kappa_1)$
6. $\mathsf{act}_2 \coloneqq \mathsf{BF.Enc}(\mathsf{ampk}, \mathsf{msg}_{i_2}, \mathsf{id}_i; \kappa_2)$
7. **return** $\mathbf{act}_i \coloneqq (\mathsf{act}_{i_1}, \mathsf{act}_{i_2})$

$\underline{\mathsf{Game}\ (1,j)}$ **for** $j \in [0,Q]$**:** We define $\mathsf{Game}\ (1,0)$ to be identical to $\mathsf{Game}\ 0$, and $\mathsf{Game}\ (1,j)$ is exactly the same as $\mathsf{Game}\ (1,0)$ except that the first j (anamorphic) ciphertexts $\{\mathbf{act}_i\}_{i\in[1,j]}$ are computed as follows:

1. Parse $\mathbf{msg}_i = (\mathsf{msg}_{i_1}, \mathsf{msg}_{i_2})$
2. $\kappa_1, \kappa_2 \leftarrow \mathbb{Z}_q^*$.
3. $\mathsf{act}_{i_1} \coloneqq \mathsf{BF.Enc}(\mathsf{ampk}, \mathsf{msg}_{i_1}, \mathsf{id}_i; \kappa_1) = (r_1 = P^{\kappa_1}, c_1 = \mathsf{msg}_{i_1} \oplus \mathsf{H}_2(g_{\mathsf{id}_i}^{\kappa_1}))$
4. $\mathsf{act}_{i_2} \coloneqq \mathsf{BF.Enc}(\mathsf{ampk}, \mathsf{msg}_{i_2}, \mathsf{id}_i; \kappa_2) = (r_2 = P^{\kappa_2}, c_2 = \mathsf{msg}_{i_2} \oplus \mathsf{H}_2(g_{\mathsf{id}_i}^{\kappa_2}))$
5. **return** $\mathbf{act}_i \coloneqq (\mathsf{act}_{i_1}, \mathsf{act}_{i_2})$

We show that $\mathsf{Game}\ (1,j)$ is indistinguishable from $\mathsf{Game}\ (1,j-1)$ by the DBDH assumption. The only difference between $\mathsf{Game}\ (1,j)$ and $\mathsf{Game}\ (1,j-1)$ is in the computation of κ_1, κ_2. In particular, κ_1, κ_2 are sampled from $\mathbb{Z}_q^*$ such that $\mathsf{H}_2(g_{\mathsf{id}_i}^{\kappa_2}) = \mathsf{amsg}_i \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}_i}^{\kappa_1})$ in $\mathsf{Game}\ (1,j-1)$, whereas κ_1, κ_2 are sampled uniformly random from $\mathbb{Z}_q^*$ in $\mathsf{Game}\ (1,j)$. We first show that $\widehat{g}_{\mathsf{id}_i}^{\kappa_1}$ is indistinguishable from a uniformly random element in $\mathbb{G}_T$ in $\mathcal{D}$'s view. Since H_1 is modeled as ROM, we can set $Q_{\mathsf{id}_i} = P^r$ for some $r \leftarrow \mathbb{Z}_q$. Without loss of generality, we assume that κ_1 is first sampled uniformly at random from $\mathbb{Z}_q$ and then κ_2 is picked from $\mathbb{Z}_q^*$ such that $\mathsf{H}_2(g_{\mathsf{id}_i}^{\kappa_2}) = \mathsf{amsg}_i \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}_i}^{\kappa_1})$ holds. Thus the challenger can simulate $\mathsf{Game}\ (1,j)$ with $(P^r, P^{\alpha}, P^{\kappa_1}, e(P,P)^t)$ when $t = r\alpha\kappa_1$, and $\mathsf{Game}\ (1,j)$, when t is uniformly random in $\mathbb{Z}_q$, where it sets $\widehat{g}_{\mathsf{id}_i}^{\kappa_1} = e(P,P)^t$. Therefore, by the DBDH assumption the following holds:

$$|\Pr[r,\alpha,\kappa_1 \leftarrow \mathbb{Z}_q : \mathcal{A}(P^r, P^{\alpha}, P^{\kappa_1}, e(P,P)^{r\cdot\alpha\cdot\kappa_1}) = 1] - \Pr[r,\alpha,\kappa_1,t \leftarrow \mathbb{Z}_q : \mathcal{A}(P^r, P^{\alpha}, P^{\kappa_1}, e(P,P)^t) = 1]| \leq \mathsf{negl}(\lambda)$$

Since $t = r\cdot\alpha\cdot\kappa_1$ is indistinguishable from uniformly random and H_2 is modeled as ROM, $\mathsf{H}_2(\widehat{g}_{\mathsf{id}_i}^{\kappa_1}) \oplus \mathsf{amsg}_i$ is statistically close to $\mathsf{H}_2(\widehat{g}_{\mathsf{id}_i}^{\kappa_1})$. Furthermore, at this stage, $g_{\mathsf{id}_i}^{\kappa_2}$ and $\widehat{g}_{\mathsf{id}_i}^{\kappa_1}$ are independently distributed where $\kappa_1, \kappa_2 \leftarrow \mathbb{Z}_q^*$ and hence the hybrids $\mathsf{Game}\ (1,j)$ and $\mathsf{Game}\ (1,j-1)$ are indistinguishable.
Finally, we observe that $\mathsf{Game}\ (1,Q)$ is exactly the same as $\mathsf{GReal}^{\mathcal{D}}_{\Pi_{2\mathsf{BF}}}(\lambda)$. This completes the proof.

IND-CPA security. We prove the theorem by designing a simulator that plays the role of the challenger in the IND-CPA security game of the Boneh-Franklin BasicIdent scheme against ampk, which is IND-CPA secure under the DBDH assumption. Note that, the anamorphic master public key ampk and the master public key mpk of the BasicIdent scheme are identically distributed since Boneh-Franklin BasicIdent scheme is IBAE-compatible by Lemma 3. In other words, if an adversary breaks the IND-CPA security of Π_{IBAE} then it can be used to break the IND-CPA security of Boneh-Franklin BasicIdent scheme.

We start with a contradiction that there exists a PPT adversary $\mathcal{A}$ that breaks the IND-CPA security of Π_{aBF} with a non-negligible probability. We then design another adversary $\mathcal{B}$ that breaks IND-CPA security of the BasicIdent scheme.

The challenger of $\mathcal{A}$ runs $\mathsf{aSetup}(1^\lambda)$ and sends $(\mathsf{ampk}, \mathsf{dmpk})$ to $\mathcal{A}$. Then, $\mathcal{B}$ receives dmpk from $\mathcal{A}$. Whenever $\mathcal{B}$ queries a secret key corresponding to an identity id, $\mathcal{A}$ sends id to it's challenger and gets a secret key $\mathsf{dsk}_{\mathsf{id}} \leftarrow \mathsf{aKeyGen}(\mathsf{dmsk}, \mathsf{id})$. As a reply, $\mathcal{B}$ receives $\mathsf{dsk}_{\mathsf{id}}$ from $\mathcal{A}$. Finally, $\mathcal{B}$ submits a challenge query for the tuple $(\mathsf{id}^*, \mathsf{amsg}_0, \mathsf{amsg}_1)$. Then, $\mathcal{A}$ prepares its challenge tuple by sampling a message of the form $\mathbf{msg} = (\mathsf{msg}_1, \mathsf{msg}_2)$ and sends $(\mathsf{id}^*, \mathbf{msg}, \mathsf{amsg}_0, \mathsf{amsg}_1)$ to the challenger. The challenger then outputs $\mathbf{act} = (\mathsf{act}_1, \mathsf{act}_2)$ where $\mathsf{act}_1 = (r_1 = P^{\kappa_1}, c_1 = \mathsf{msg}_1 \oplus \mathsf{H}_2(g_{\mathsf{id}^*}^{\kappa_1}))$ and $\mathsf{act}_2 = (r_2 = P^{\kappa_2}, c_2 = \mathsf{msg}_2 \oplus \mathsf{H}_2(g_{\mathsf{id}^*}^{\kappa_2})$. Now, $\mathcal{A}$ sets the challenge ciphertext for $\mathcal{B}$ as $\mathsf{act}^* = (r_1 = P^{\kappa_1}, c_2 \oplus \mathsf{msg}_2 = \mathsf{H}_2(g_{\mathsf{id}^*}^{\kappa_2}))$. Next, $\mathcal{B}$ can submit more private key queries to which $\mathcal{A}$ replies as before. At the end, $\mathcal{B}$ outputs a bit which is also the output of $\mathcal{A}$.

We observe that $\mathcal{A}$ is a valid IND-CPA adversary of Π_{aBF} since $\mathcal{B}$ can only query for secret keys corresponding to $\mathsf{id} \neq \mathsf{id}^*$. Next, by construction, we have

$$c_2 = \mathsf{msg}_2 \oplus \mathsf{H}_2(g_{\mathsf{id}^*}^{\kappa_2}) = \mathsf{msg}_2 \oplus \mathsf{amsg}_\beta \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}^*}^{\kappa_1})$$

where β is the challenge bit used by the challenger of $\mathcal{A}$. Therefore, $\mathsf{act}^* = (r_1 = P^{\kappa_1}, c_2 \oplus \mathsf{msg}_2 = \mathsf{amsg}_\beta \oplus \mathsf{H}_2(\widehat{g}_{\mathsf{id}^*}^{\kappa_1}))$ is a valid ciphertext of the Boneh-Franklin BasicIdent scheme encrypting amsg_β using ampk. Therefore, if $\mathcal{A}$ breaks the IND-CPA security of Π_{aBF} with a non-negligible probability then $\mathcal{B}$ can also break the IND-CPA security of the Boneh-Franklin BasicIdent scheme.

□

F Anamorphic Extension of FO-Based IBE

Variants of the FO transformation are also a viable tool to construct CCA secure IBE schemes. This has started with the intial construction due to Boneh and Franklin on IBE [8] and several works have explicitly investigated various FO-type transformations for lifting OW-CPA or IND-CPA secure IBE to CCA secure ones [21,37,24]. Analogous to the discussion of the FO transformation in Section 3.3, it is natural to ask whether we can equivalently apply these results to the AIBE setting. We first want to briefly recall how the variant of the FO transform that is used in the Boneh-Franklin (BF) IBE and the Sakai-Kasahara

(SK) IBE [33, 14] exactly work. We note that both works as specified below are suggested in an ETSI technical report for standardization [18][8].

For our presentation let $\Pi_{\mathsf{IBE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ and $\Pi_{\mathsf{SKE}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be the underlying IBE and symmetric encryption schemes. Analogous to Section 3.3 we now denote by $\Pi_{\mathsf{FO,IBE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ the IND-CCA secure version where $\mathsf{FO}_{\mathsf{IBE}}.\mathsf{Setup}$ and $\mathsf{FO}_{\mathsf{IBE}}.\mathsf{KeyGen}$ just run $\mathsf{IBE}.\mathsf{Setup}$ and $\mathsf{IBE}.\mathsf{KeyGen}$ respectively. $\mathsf{FO}_{\mathsf{IBE}}.\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, \mathsf{msg})$ returns ciphertext $c = (c_1, c_2)$ where $c_1 = \mathsf{IBE}.\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, \mathsf{msg}; \sigma)$ and $c_2 = \mathsf{SKE}.\mathsf{Enc}(G(r), \mathsf{msg})$ for uniformly random $r \in \{0,1\}^n$ (and suitable n) and $\sigma = H(r, \mathsf{msg}) \in \mathcal{M}_{\mathsf{IBE}}$. Note that H and G are hash functions modeled as random oracles mapping to the space of random coins $\mathcal{R}_{\mathsf{IBE}}$ of IBE and the key space $\mathcal{K}_{\mathsf{SKE}}$ of SKE, respectively. For decryption, $\mathsf{FO}_{\mathsf{IBE}}.\mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, c)$ parses $c = (c_1, c_2)$ computes $r' = \mathsf{IBE}.\mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, c_1)$ and $\mathsf{msg}' = \mathsf{SKE}.\mathsf{Dec}(G(r'), c_2)$ and returns msg' if $c_1' = \mathsf{IBE}.\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, H(r', \mathsf{msg}'))$ or $\perp$ otherwise.

Definition 19 *Let Π_{IBE} be IBE with ciphertext space denoted by $\mathcal{C}$. We call* IBE IND$-CPA *secure if for every PPT adversary $\mathcal{A}$ it holds that*

$$\left|\Pr[\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{IBE}}}(\lambda, 1) = 1] - \Pr[\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{IBE}}}(\lambda, 0) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where the security game is defined as follows:

$\underline{\mathsf{GIND\$\text{-}CPA}^{\mathcal{A}}_{\Pi_{\mathsf{IBE}}}(1^\lambda, \beta) :}$

1. $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$
2. $(\mathsf{id}^*, \mathsf{msg}) \leftarrow \mathcal{A}^{\mathsf{OKeyGen}(\cdot)}(\mathsf{mpk})$
3. $\mathsf{ct}_0 = \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}^*, \mathsf{msg})$
4. $\mathsf{ct}_1 \leftarrow \mathcal{C}$
5. ***return*** $\beta' \leftarrow \mathcal{A}^{\mathsf{OKeyGen}(\cdot)}(\mathsf{ct}_\beta)$
   where $\mathsf{OKeyGen}(\mathsf{id}) = \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$
   and id^* *is not queried to* $\mathsf{OKeyGen}(\cdot)$.

We present the anamorphic IBE extension based on $\Pi_{\mathsf{FO,IBE}}$ in Figure 8.

Theorem 8 *Let $\Pi_{\mathsf{FO,IBE}}$ be an* IND-ID-CCA *secure FO-based IBE where the underlying IBE Π_{IBE} is* IND$-CPA *secure and has an exponentially large message space $\mathcal{M}_{\mathsf{IBE}}$ (i.e., $|\mathcal{M}_{\mathsf{IBE}}| \geq 2^{\mathsf{poly}(\lambda)}$ for non-constant* poly*), and the underlying SKE Π_{SKE} is* wIND$-CPA *secure. Furthermore, let* (Encode, Decode) *be a pseudo-random $(\mathcal{C}_{\mathsf{FO}_{\mathsf{IBE}}}, \mathcal{M}^{\mu}_{\mathsf{IBE}})$-encoding, where $\mathcal{C}_{\mathsf{FO}_{\mathsf{IBE}}}$ denotes the ciphertext space of $\Pi_{\mathsf{FO,IBE}}$. Then $\Pi_{\mathsf{FOAE,IBE}}$ as defined in Figure 8 is a μ-message anamorphic identity-based encryption extension satisfying anamorphic security and* sIND-RCCA *security in the ROM.*

[8] The only difference with the standard, as with the papers, is that they explicitly consider XOR as symmetric encryption scheme SKE.

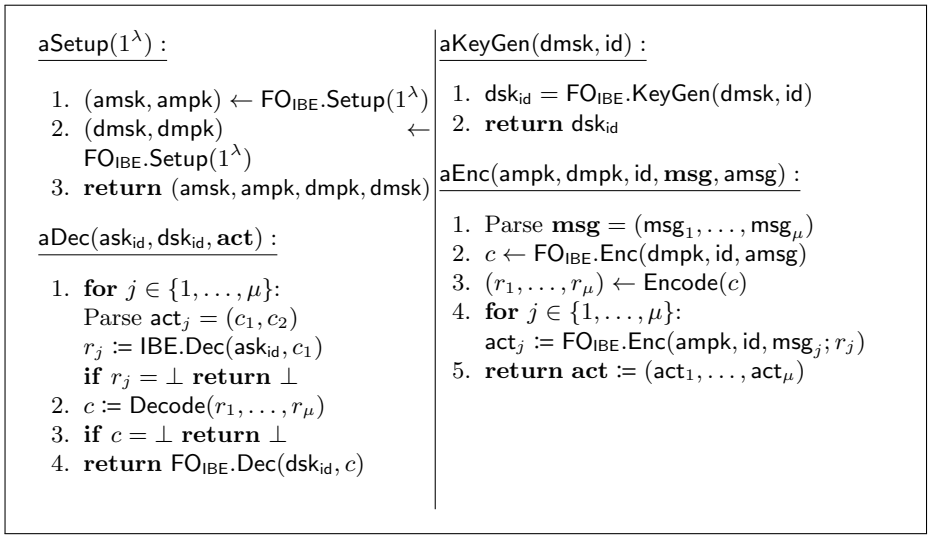

Fig. 8: μ-AIBE $\Pi_{\mathsf{FOAE,IBE}}$

The proof of this theorem works analogous to that of Theorem 3 and is thus omitted. The intuition is that $\Pi_{\mathsf{FO,IBE}}$ satisfies anamorphic security as the embedded ciphertexts components containing amsg are indistinguishable from a regular $r \leftarrow \mathcal{M}_{\mathsf{IBE}}$ due to the pseud-randomness of the pseudo-random encoding. Moreover, the IND-CCA security of $\Pi_{\mathsf{FO,IBE}}$ for encrypting amsg (i.e., the "inner" layer encryption) implies sIND-RCCA for the overall construction $\Pi_{\mathsf{FOAE,IBE}}$ as an adversary (including the dictator) is not able to maul amsg.

Instantiation with BF Full-Ident. Let us consider $\Pi_{\mathsf{FO,IBE}}$ based on Boneh-Frankin (BF) as IBE and XOR for the SKE part. BF satisfies IND\$-CPA security under DBDH in the ROM and XORing a message with a random key of the same length satisfies wIND\$-CPA security. When instantiated in type-3 pairings $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, q)$, let us assume that c_1 elements are in $\mathbb{G}_1$ and secret keys sk$_{\mathsf{id}}$ are in $\mathbb{G}_2$. Note that, FO$_{\mathsf{IBE}}$.Enc(mpk, msg) would produce a ciphertext $c = (g_1^{H(r,m)}, H'(g_{2,\mathsf{id}}^{H(r,m)}) \oplus r, G(r) \oplus m)$, where $H : \{0,1\}^* \times \{0,1\}^* \to \mathbb{Z}_q$, $H' : \mathbb{G}_T \to \{0,1\}^{\ell_1}$, $G : \{0,1\}^* \to \{0,1\}^{\ell_2}$. As for Π_{FOAE}, we can use Elligator Squared [35] as the pseudo-random encoding and considering BN curves [4] for instantiating the type-3 pairing we have $\mathbb{G}_1 = \mathbb{E}(\mathbb{F}_p)$ is a cyclic group of prime order q. Moreover, if we again set $\ell_1 = \ell_2 = \lambda$, then we end up with a 4-message anamorphic extension.

G Analysis of Anamorphism-Friendly Signatures with Compliant Encryption

Theorem 5 Let Π_{SS} be an anamorphism-friendly signature scheme and Π_{ES} an encryption scheme compliant to Π_{SS}. Then Π_{aGAE} as defined in Figure 7 is a secure μ-message anamorphic signature extension, where $\mu = \max\{\lceil \frac{e'_i}{e_i} \rceil\}$. Moreover, Π_{aGAE} also satisfies IND-CPA security. If Π_{ES} additionally achieves IND-RCCA security then Π_{aGAE} achieves sIND-RCCA security.

Proof. We prove this theorem in two parts.

Anamorphic Security. In the following sequence of hybrid games let $\mathcal{D}$ be a PPT dictator and Q its total number of oracle queries.

Game 0**:** This is the real game $\mathsf{GReal}^{\mathcal{D}}_{\mathsf{SS}}(\lambda)$ in the anamorphic security definition, where the challenger generates $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{SS.Gen}(\mathsf{gp})$ and handles signature queries $\mathsf{OSig}(\mathbf{msg}, \mathsf{amsg})$ of the dictator by computing $(\mathsf{SS.Sig}(\mathsf{sk}, \mathsf{msg}_i)$ for $0 \leq i \leq \mu - 1$.

Game 1**:** In this game the challenger generates the key pair by computing $(\mathsf{ask}, \mathsf{avk}, \mathsf{td}) \leftarrow \mathsf{SS.Gen}'(\mathsf{gp})$ and provides $(\mathsf{ask}, \mathsf{avk})$ as input to $\mathcal{D}$.

Game $(2, j)$ **for** $j \in [0, \mu Q]$**:** In this series of games we slightly modify Game 1. We define Game $(2, 0)$ to be identical to Game 1. In Game $(2, j)$, the challenger handles the first j signature computations (in order to respond to oracle calls) as follows: It computes $(s_0, \ldots, s_{m-1}, r) \leftarrow \mathsf{SS.Sig}(\mathsf{ask}, \mathsf{msg}_j)$ but replaces r with $r' \leftarrow \mathsf{SS.RemSig}(\mathsf{ask}, \mathsf{td}, \mathsf{msg}_j, (s_0, \ldots, s_{m-1}))$, i.e., it returns $(s_0, \ldots, s_{m-1}, r')$. The remaining signature computations are left untouched in comparison to Game $(2, 0)$. Note that in Game $(2, \mu Q)$ we have modified all signature computations in this way.

Game 3**:** In this game, we modify all signature computations done by the challenger (in comparison to Game $(2, \mu Q)$) at once: instead of calling SS.Sig to compute $(s_0, \ldots, s_{m-1})$ being input to SS.RemSig, we choose them uniformly at random, i.e., $(s_0, \ldots, s_{m-1}) \leftarrow S_0^{e_0} \times S_{m-1}^{e-1}$.

Game $(4, j)$ **for** $j \in [0, Q]$**:** We define Game $(4, 0)$ to be identical to Game 3, except that the challenger additionally generates the key pair $(\mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{ES.Gen}(\mathsf{gp})$ of the encryption scheme. In Game $(4, j)$, the challenger handles the first j oracle calls $\mathsf{OSig}(\mathbf{msg}, \mathsf{amsg})$ as follows: it computes $(c_0, \ldots, c_{m-1}) \leftarrow \mathsf{ES.Enc}(\mathsf{dk}, \mathsf{amsg})$ and embeds the ciphertext components into $(s_0, \ldots, s_{m-1})$ as defined by Steps 2 and 3 of the aSig definition in Figure 7. Then SS.RemSig is called as in Game 3. Note that in Game $(4, Q)$ we have embedded ciphertexts in all responses to oracle calls.

Game 5**:** This is the anamorphic game $\mathsf{GAnam}^{\mathcal{D}}_{\mathsf{GAE}}(\lambda)$, where the challenger generates $(\mathsf{ask}, \mathsf{avk}, \mathsf{td}) \leftarrow \mathsf{GAE.aGenS}(\mathsf{gp})$, $(\mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{GAE.aGenR}(\mathsf{gp})$ and handles signature queries $\mathsf{OaSig}(\mathbf{msg}, \mathsf{amsg})$ of the dictator by computing $\mathsf{aSig}(\mathsf{ask}, \mathsf{td}, \mathsf{dk}, \mathbf{msg}, \mathsf{amsg})$.

The indistinguishability of Game 0 and Game 1 follows immediately from the second property of an anamorphism-friendly signature. Game 1 and Game $(2, 0)$ are identical.

Game $(2,0)$ is indistinguishable from Game $(2,Q)$, as a distinguisher $\mathcal{D}$ for Game $(2,j)$ and Game $(2,j+1)$ (for random $j \in \{0,\ldots,\mu Q-1\}$) can be turned in to an adversary against the black-box computation of remainder property of an anamorphism-friendly signature. Game $(2,\mu Q)$ is perfectly indistinguishable from Game 3 due to the random component property of the signature scheme. Game 3 and Game $(4,0)$ are perfectly indistinguishable as the modification is purely conceptional. Game $(4,0)$ is indistinguishable from Game $(4,Q)$, as from a distinguisher $\mathcal{D}$ for Game $(4,j)$ and Game $(4,j+1)$ (for random $j \in \{0,\ldots,Q-1\}$) an adversary against the IND$-CPA security of Π_{ES} can be constructed. Finally, Game $(4,Q)$ and Game 5 are identical.

IND-CPA Security of Π_{aGAE}. This follows immediately from the IND$-CPA security of Π_{ES}: Let Game 0 be the IND-CPA game for $\beta = 1$. In Game 1, when computing the challenge signature $\mathsf{asig}_1 \leftarrow \mathsf{aSig}(\mathsf{ask},\mathsf{td},\mathsf{dk},\mathbf{msg},\mathsf{amsg}_1)$, we replace the ciphertexts $(c_0,\ldots,c_{m-1})$ to be embedded by aSig by randomly chosen values from the ciphertext space $S_0^{e_0} \times \ldots \times S_{m-1}^{e_{m-1}}$. Game 0 and Game 1 are indistinguishable if Π_{ES} is IND$-CPA. Note that the reduction algorithm can handle signature queries as it can choose ask,td itself. In Game 2, we immediately revert this change by embedding ciphertexts encrypting amsg_0, i.e., the challenger computes $\mathsf{asig}_0 \leftarrow \mathsf{aSig}(\mathsf{ask},\mathsf{td},\mathsf{dk},\mathbf{msg},\mathsf{amsg}_0)$. Again, Game 1 and Game 2 are indistinguishable if Π_{ES} is IND$-CPA. Game 2 is the IND-CPA game for $\beta = 0$.

sIND-RCCA Security of Π_{aGAE}. This follows immediately from the assumed IND-RCCA security of Π_{ES}. The reduction algorithm $\mathcal{B}$ is given gp,dk as input from its IND-RCCA challenger and has access to a $\mathsf{Dec}(\mathsf{tk},\cdot)$ oracle. It can generate $(\mathsf{avk},\mathsf{ask},\mathsf{td}) \leftarrow \mathsf{aGenR}(\mathsf{gp})$ itself, and thus provide $(\mathsf{avk},\mathsf{dk},\mathsf{ask})$ as input to the sIND-RCCA adversary $\mathcal{A}$. Clearly, $\mathcal{B}$ can handle $\mathcal{A}$'s $\mathsf{aSig}(\mathsf{ask},\mathsf{td},\mathsf{dk},\cdot,\cdot)$ oracle queries since it knows all necessary inputs. $\mathcal{B}$ can also handle $\mathsf{aDec}(\mathsf{tk},\mathsf{avk},\cdot,\cdot)$ queries by extracting the $c_{i,j}$ from the given **asig** as specified in Figure 7 and querying its own $\mathsf{Dec}(\mathsf{tk},\cdot)$ oracle this input. Clearly, this simulated aDec oracle behaves exactly as the real aDec oracle (in particular it rejects an input exactly when the real one does). To generate the challenge signature for the given $(\mathbf{msg},\mathsf{amsg}_0,\mathsf{amsg}_1)$, $\mathcal{B}$ provides $(\mathsf{amsg}_0,\mathsf{amsg}_1)$ to its own challenger and embeds the response into signatures for **msg** as specified by Steps 3 to 5 of aSig in Figure 7. Finally $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

IND-CCA Security of $\Pi_{\mathsf{aGAE}'}$. This proof is similar to the sIND-RCCA security proof for Π_{aGAE} we have seen above, where the SUF-CMA security of Π_{SS} now ensures that the simulation of the aDec using a IND-CCA Dec oracle works out.

So let us consider how the reduction algorithm $\mathcal{B}$ (the adversary against the IND-CCA security of Π_{ES}) handles $\mathcal{A}$'s queries of the $\mathsf{aDec}(\mathsf{tk},\mathsf{avk},\cdot,\cdot)$ IND-CCA oracle: For given $(\mathbf{asig},\mathbf{msg})$, it first checks whether $(\mathbf{asig},\mathbf{msg})$ equals the challenge signature and message vectors $(\mathbf{asig}^*,\mathbf{msg}^*)$ and rejects if it does. Then it verifies each of the μ signatures contained **asig** and returns $\perp$ if at least

one of these checks fail. Otherwise, it extracts the $c_{i,j}$ from the given $\mathbf{asig}$ as specified in Figure 7 and queries its own $\mathsf{Dec}(\mathsf{tk},\cdot)$ oracle for this input. Obviously, the only thing this simulation does differently compared to a real aDec oracle is calling the $\mathsf{Dec}(\mathsf{tk},\cdot)$ oracle instead of executing the $\mathsf{Dec}(\mathsf{tk},\cdot)$ algorithm. This can lead to the following simulation failure: The $\mathsf{Dec}(\mathsf{tk},\cdot)$ oracle on input of the challenge ciphertext $c^* = (c^*_{i,j})_{i,j}$ (generated by $\mathcal{B}$'s $\mathsf{IND\text{-}CCA}$ challenger) will reject while the algorithm will return the corresponding decryption result. Hence, we need to show that the checks (particularly the signature verification) performed before this point ensure that the $\mathsf{Dec}(\mathsf{tk},\cdot)$ oracle will never be queried with c^* as input.

Let us consider the different inputs $(\mathbf{asig},\mathbf{msg})$ to the aDec oracle which could lead to c^*:

1. $(\mathbf{asig},\mathbf{msg}) = (\mathbf{asig}^*,\mathbf{msg}^*)$: In this case $\mathcal{B}$ immediately rejects.
2. $\mathbf{asig} = \mathbf{asig}^*$ and $\mathbf{msg} \neq \mathbf{msg}^*$: In this case, we have at least one verifying signature sig_i^* for a non-challenge message $\mathsf{msg}_i \neq \mathsf{msg}_i^*$. There is only a negligible chance that a call to the aSig oracle (including msg_i as part of the input) could have produced the same signature since the used randomness for producing this signature is freshly chosen. In particular, if we write sig_i^* as $\mathsf{sig}_i^* = ((s_j^{(i)}, r^{(i)})$ as in Figure 7, for the newly produced signature the same $s_j^{(i)}$ would need to be chosen. Hence, in this case the $\mathsf{SUF\text{-}CMA}$ security is broken and thus we could build a adversary against the underlying signature scheme.
3. $\mathbf{asig} \neq \mathbf{asig}^*$ and $\mathbf{msg} = \mathbf{msg}^*$: In this case, we have at least one signature $\mathsf{sig}_i \neq \mathsf{sig}_i^*$ which verifies for the challenge message msg_i^*. Since also with this signature sig_i, the extraction of the ciphertext need to yield c^* which is composed of the $s_j^{(i)}$, it holds that at least one of the random components of sig_i needs to coincide with the one of sig_i^*. Similar to the previous case, we can argue that such sig_i cannot stem from a call to the aSig oracle and thus sig_i is a forgery of a new signature for the challenge message msg_i^*. Thus, in this case the $\mathsf{SUF\text{-}CMA}$ security is broken.
4. $\mathbf{asig} \neq \mathbf{asig}^*$ and $\mathbf{msg} \neq \mathbf{msg}^*$: In this case, we are either in one of the two previous situations (for which we have already shown that they violate $\mathsf{SUF\text{-}CMA}$ security) or for all i it holds that $\mathsf{sig}_i \neq \mathsf{sig}_i^*$ and $\mathsf{msg}_i \neq \mathsf{msg}_i^*$. So let us consider the latter. Using the same argument as in the previous case, it follows that there is a negligible probability that sig_i stems from a call to the aSig oracle and thus we either have a signature forgery for a new message or a message for which the aSig oracle has been queried before. Both cases violate the $\mathsf{SUF\text{-}CMA}$ security of the signature scheme.

To summarize, assuming $\mathsf{SUF\text{-}CMA}$ security, the $\mathsf{IND\text{-}CCA}$ adversary $\mathcal{B}$ against Π_{ES} can perfectly simulate the aDec oracle for $\mathcal{A}$ and thus $\mathcal{A}$'s $\mathsf{IND\text{-}CCA}$ challenger for Π_{aGAE}. Finally, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

To make the above argument more formal, one can define a sequence of games, where the first game is the $\mathsf{IND\text{-}CCA}$ game for $\Pi_{\mathsf{aGAE}'}$ for $\beta = 1$ and the

last game is the IND-CCA game for $\Pi_{\mathsf{aGAE}'}$ for $\beta = 0$. In the first three intermediate games, we modify the aDec oracle by considering the cases 2-4 from above and let the oracle reject if the corresponding event happens. Indistinguishability between the games can then be shown using the SUF-CMA security of Π_{SS} (based on the arguments made above). After that, we can be sure that an aDec oracle can be simulated by a Dec oracle and thus we can change the challenge message to be encrypted from amsg_1 to amsg_0 in the next and final game. Here indistinguishability follows from the IND-CCA security for Π_{ES}.

□

H Analysis of Waters and BBS Anamorphic Signature Extensions

Theorem 6 Waters signatures (as defined in Sec. 5.1) are anamorphism-friendly and linear encryption is compliant to them under the DLIN assumption.

Proof. We prove the theorem in two parts.

Anamorphism-Friendliness. The signature space of Waters is $\mathbb{G}^2$ where $(\mathbb{G}, \mathbb{G}_T, e, p)$ is a bilinear group setting. In the notation of Defintion 9, we can write it as $S_0 \times R$, with $S_0 = \mathbb{G}$ and $R = \mathbb{G}$. Considering only the first component of a signature $(g^d, g^{\alpha\beta} H(\mathsf{msg})^d)$, we observe that it is uniformly and freshly as $d \leftarrow \mathbb{Z}_p$. Hence, the first property is satisfied. Regarding the second property, we observe that $g^{\alpha\beta} H(\mathsf{msg})^d = g^{\alpha\beta}(h_0 \prod_{i=1}^{\ell} h_i^{m_i})^d$, where $h_i = g^{a_i}$ are elements chosen during signature key generation, can alternatively be computed by setting $\mathsf{td} \coloneqq (a_0, \ldots, a_n)$ and $\mathsf{RemSig}(\mathsf{ask}, \mathsf{td}, \mathsf{msg}, s = g^d) \coloneqq \mathsf{ask}\, s^{a_0} \prod_{i=1}^{\ell} s^{a_i \mathsf{msg}_i}$. Gen' outputs a key pair with the same distribution as Gen and additionally td. This implies the second property.

Compliance of Linear Encryption. It is easy to see that the linear encryption scheme as defined in Section 5.1 is IND$-CPA over the group $\mathbb{G}$ under the DLIN assumption. The DLIN problem asks an adversary to distinguish h^{x+y} from h^z, where $z \leftarrow \mathbb{Z}_p$, given (u, v, h, u^x, v^y), where $u, v, h \leftarrow \mathbb{G}$ and $x, y \leftarrow \mathbb{Z}_p$. First we observe that the public key (u, v, h) of the linear encryption scheme can be generated by choosing $a, b, x \leftarrow \mathbb{Z}_p$ and setting $y \coloneqq axb^{-1}$, $u \coloneqq g^a$, $v \coloneqq g^b$, $h \coloneqq g^{ax}$. In this way, (u, v, h) are indeed random generators as in the DLIN problem and they satisfy $u^x = v^y = h$ as needed for decryption. Now, the reduction to the DLIN assumption is very simple: given an instance of the DLIN problem (u, v, h, u^x, v^y, w), we compute the challenge ciphertext for the IND$-CPA adversary as $(c_1, c_2, c_3) = (u^x, v^y, w \cdot \mathsf{msg})$ and return the output of the adversary as our guess. Clearly, if w is random then also the ciphertext, if $w = h^{x+y}$ then it is a proper encryption of msg. Finally, observe that the ciphertext space of the linear encryption is $\mathbb{G}^3$ and is thus compliant with Waters signature scheme. □

Theorem 7 BBS signatures (as defined in Sec. 5.2) are anamorphism-friendly and ElGamal encryption, where ciphertexts are encoded during encryption and

decoded during decryption using a pseudo-random $\mathbb{Z}_p^k$-encoding, is compliant to BBS under the DDH assumption.

Proof. We prove the theorem in two parts.

Anamorphism-Friendliness. The signature space of BBS can be written in the form of Definition 9 as $S_0 \times R$, where $S_0 = \mathbb{Z}_p$ and $R = \mathbb{G}$. As the first component of each signature $(s_0, H(\mathsf{msg})^{\frac{1}{x+s_0}})$ is uniformly and freshly chosen, the first property is satisfied. The second one is also trivially satisfied by setting $\mathsf{td} \coloneqq \bot$ and $\mathsf{RemSig}(\mathsf{ask} = x, \mathsf{td}, \mathsf{msg}, s_0) \coloneqq H(\mathsf{msg})^{\frac{1}{x+s_0}}$.

Compliance of ElGamal+Pseudo-Random Encoding. Let us consider ElGamal encryption over $\mathbb{G}_1$ of an asymmetric bilinear groups setting $\mathsf{gp} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, , g_1, g_2, e, p)$ and Encode and Decode algorithms mapping elements from $\mathbb{G}_1$ to $\mathbb{Z}_p^k$ and vice versa. Then the encryption scheme defined by $\mathsf{Gen}(\mathsf{gp}) \coloneqq \mathsf{El.Gen}(\mathsf{gp})$, $\mathsf{Enc}(\mathsf{pk}, \mathsf{msg}) \coloneqq (\mathsf{Encode}(c_1), \mathsf{Encode}(c_2))$, where $(c_1, c_2) \leftarrow \mathsf{El.Enc}(\mathsf{pk}, \mathsf{msg})$, $\mathsf{Dec}(\mathsf{sk}, (c_1', c_2')) \coloneqq \mathsf{El.Dec}(\mathsf{sk}, (\mathsf{Decode}(c_1'), \mathsf{Decode}(c_2')))$ has ciphertext space $\mathbb{Z}_p^{2k}$, provides correctness up to a negligible error probability, and satisfies IND$-CPA security. Regarding IND$-CPA security, we first observe that ElGamal is IND$-CPA secure under the DDH assumption (the proof is similar to the one for linear encryption under DLIN), and that the output of Encode for some uniform $c_i \in \mathbb{G}_1$ is indistinguishable from an uniform element of $\mathbb{Z}_p^k$. Finally, observe that the ciphertext space of the combined encryption scheme is $\mathbb{Z}_p^{2k}$ and is thus compliant to the signature space component $S_0 = \mathbb{Z}_p$ of the BBS signature scheme. □

I Robustness of Public-Key Anamorphic Encryptions

Additionally, PKAE could also satisfy the following robustness property.

Robustness: For every PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\lambda)$, such that for all $\lambda \in \mathbb{N}$, the following holds:

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

where the security games are defined as follows:

$\mathsf{Robust}^{\mathcal{A}}_{\mathsf{pkae}}(\lambda, \beta)$:

1. $(\mathsf{apk}, \mathsf{ask}, \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{aGen}(1^\lambda)$
2. **return** $\mathcal{A}^{\mathcal{O}^\beta(\mathsf{apk}, \mathsf{ask}, \mathsf{tk}, \mathsf{dk}, \cdot)}(\mathsf{apk}, \mathsf{ask})$ where
   $\mathcal{O}^0(\mathsf{apk}, \mathsf{ask}, \mathsf{tk}, \mathsf{dk}, \mathsf{msg}) = \mathsf{aDec}(\mathsf{tk}, \mathsf{Enc}(\mathsf{apk}, \mathsf{msg}))$
   $\mathcal{O}^1(\mathsf{apk}, \mathsf{ask}, \mathsf{tk}, \mathsf{dk}, \mathsf{msg}) = \bot$

Robustness of Π_{aEl} from Section 3.1. To obtain robustness in the ElGamal anamorphic construction, we simply allow the decryption algorithm aEl.aDec to exhaustively search the anamorphic message space, and return $\bot$ if $\mathsf{amsg} \notin \widehat{\mathcal{M}}$.

To see why this technique is effective, observe that algorithm $\mathsf{2El.Enc}$ samples k_2 uniformly at random from $\mathbb{Z}_q$, which for a fixed α and a fixed amsg, collides with $\kappa\alpha + \mathsf{amsg}$ with probability $\frac{1}{q}$ (a negligible function in λ).

Robustness of Π_{aDReg} from Section 3.2. To obtain robustness in the Dual-Regev public key anamorphic 2-message extension, we could simply redefine the anamorphic decryption algorithm $\mathsf{aDReg.aDec}$ to include an additional sanity-check for the ciphertext $F(\mathbf{act}) = (\widehat{\mathbf{r}}, \widehat{c})$ as follows:

1. **if** $|\widehat{c} - \widehat{\mathbf{r}} \cdot \widehat{\mathbf{k}}| \leq \frac{7q}{10}$ **then**
2. **return** $\mathsf{amsg} \coloneqq \mathsf{aDReg.aDec}(\mathsf{tk}, F(\mathbf{act}))$
3. **else return** $\perp$

To see why this technique is effective, observe that

$$|\widehat{c} - \widehat{\mathbf{r}} \cdot \widehat{\mathbf{k}}| = \mathbf{s}_1^T \cdot \mathbf{A} \cdot \widehat{\mathbf{k}} + \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor - \mathbf{s}_1^T \cdot \mathbf{A} \cdot \widehat{\mathbf{k}} - \mathbf{e}_2^T \cdot \widehat{\mathbf{k}} \mod q$$

(the anamorphic decryption algorithm knows $\mathsf{tk} = \widehat{\mathbf{k}}$). If the ciphertext contains amsg, then $\mathbf{s}_1^T \cdot \mathbf{A} \cdot \widehat{\mathbf{k}}$ cancels out, and the above term reduces to $t = \widehat{y} + \mathsf{amsg} \cdot \lfloor \frac{q}{2} \rfloor - \mathbf{e}_2^T \cdot \widehat{\mathbf{k}} \mod q$. Now, following our parameter choices $m \geq 2n \log q$, $r = \omega(\sqrt{\log m})$, $q \geq 5r(m+1)$, $\alpha \leq 1/(r\sqrt{m+1} \cdot \omega(\sqrt{\log n}))$ and $\chi = \Psi_\alpha$, the norm of the error $\widehat{y} - \mathbf{e}_2^T \cdot \widehat{\mathbf{k}} \mod q$ is smaller than $\frac{q}{5}$ with overwhelming probability in λ. This implies that t is upper-bounded by $\frac{7q}{10}$.

When the ciphertext does not contain amsg, then $\widehat{c}$ is a uniformly random element of $\mathbb{Z}_q$, and hence the term $\mathbf{s}_1^T \cdot \mathbf{A} \cdot \widehat{\mathbf{k}}$ does not cancel out with high probability over the choices of $\mathbf{A}$ and $\mathbf{s}$, which implies that the norm of $\widehat{c} - \widehat{\mathbf{r}} \cdot \widehat{\mathbf{k}}$ is larger than $\frac{7q}{10}$.

J Robustness of Public-Key Anamorphic Signatures

For a PKASE or more precisely for anamorphism-friendly signatures with compliant encryption we could define robustness analogous to [3]. This captures that it must be hard to find μ messages $(\mathsf{msg}_1, \ldots, \mathsf{msg}_\mu)$ for which when generating a vector of normal signatures viewed as an anamorphic signature $\mathsf{asig} = (\mathsf{SS.Sig}(\mathsf{ask}, \mathsf{msg}_1), \ldots, \mathsf{SS.Sig}(\mathsf{ask}, \mathsf{msg}_\mu))$, which is then subsequently anamorphically decrypted into $\mathsf{amsg} := \mathsf{aDec}(\mathsf{tk}, \mathsf{asig})$, it holds that $\mathsf{amsg} \neq \perp$. Formally, as done for an anamorphic extension of a PKE in [3], this can be done by defining a distinguishing problem between an oracle implementing the aforementioned and an oracle that always simply returns $\perp$.

Unfortunately, it is not clear how robustness could generically be related to the $\mathsf{IND\$\text{-}CPA}$ security of the compliant PKE scheme used to construct the PKASE according to Definiton 9. When looking at our concrete constructions, however, we can observe the following.

It is easy to see that the 3-PKASE Π_{aWaters} in Figure 5 uses an ElGamal type encryption (namely linear encryption) and thus we can follow the ElGamal

public key anamorphic 2-message extension Π_{aEl} in Figure 1. Namely, we could use linear encryption in the exponent, i.e., instead of encrypting $\mathsf{amsg} \in \mathbb{G}$ we encrypt g^{amsg} for amsg coming from a polynomially bounded subset of $\mathbb{Z}_p$, and follow the exact same idea.

For the 4-PKASE Π_{aBBS} in 6, although using ElGamal encryption and thus suggesting the use of the same strategy as above, it is not obvious how to achieve this. More precisely, when we consider a vector of μ normal signatures, then elements s_i, s_j, $i \neq j$ from any such two signatures when decoded, need to yield a random element in $\mathbb{G}_1$. However, this depends on the concrete encoding and is something that would need to be satisfied by the Elligator Squared encoding in our specific case or in general by the algorithm Decode of the pseudo-random $\mathbb{Z}_p^k$-encoding.

We leave a thorough study of robustness in context of PKASE and anamorphism-friendly signatures with compliant encryption for future work.