On Generic Constructions of Circularly-Secure, Leakage-Resilient Public-Key Encryption Schemes

Mohammad Hajiabadi*, Bruce M. Kapron*, and Venkatesh Srinivasan *

Department of Computer Science, University of Victoria, Canada {mhaji,bmkapron,srinivas}@uvic.ca

Abstract. We propose generic constructions of public-key encryption schemes, satisfying *key-dependent message (KDM) security for projections* and different forms of *key-leakage resilience*, from CPA-secure private-key encryption schemes with two main abstract properties: (1) a form of (additive) homomorphism with respect to both plaintexts and randomness, and (2) *reproducibility*, providing a means for reusing encryption randomness across independent secret keys. More precisely, our construction transforms a private-key scheme with the stated properties (and one more mild condition) into a public-key one, providing:

- KDM-projection security, an extension of circular security, where the adversary may also ask for encryptions of negated secret key bits;
- a $(1 - o(1))$ resilience rate in the bounded-memory leakage model of Akavia et al. (TCC 2009); and
- *Auxiliary-input security* against subexponentially-hard functions.

We introduce *homomorphic weak pseudorandom functions*, a homomorphic version of the weak PRFs proposed by Naor and Reingold (FOCS '95) and use them to realize our base encryption scheme. We in turn obtain homomorphic weak PRFs from *homomorphic hash-proof systems* (*HHPS*). We also show how the base encryption scheme may be realized using subgroup indistinguishability (implied, in particular, by quadratic residuosity (QR) and decisional composite residuosity (DCR)). As corollaries of our results, we obtain (1) the first multiple-key projection-secure bit-encryption scheme (as well as the first scheme with a $(1 - o(1))$ resilience rate) based solely on the HHPS assumption, and (2) a unifying approach explaining the results of Boneh et al (CRYPTO '08) and Brakerski and Goldwasser (CRYPTO '10). Finally, by observing that Applebaum's KDM amplification method (EUROCRYPT '11) preserves both types of leakage resilience, we obtain schemes providing at the same time high leakage resilience and KDM security against any fixed polynomial-sized circuit family.

1 Introduction

A central goal in cryptography is to build a variety of cryptographic primitives with a high degree of versatility from assumptions that are as general as possible. Encryption in particular has been defined, starting with the seminal paper of Goldwasser and Micali [23], with respect to successively strong models of security. However, standard notions of encryption security (i.e., semantic (CPA) and different forms of chosen-ciphertext (CCA) security [23,36,38,17]) fall short in certain applications, in particular, where the adversary may obtain some side information about the internal secret parameters (e.g., the secret key) of the scheme. This leakage of side information may occur due to some unforeseen attacks on the scheme (*side-channel attacks*), or more fundamentally, when encryption is used as a primitive in a complex protocol which may inherently expose inside information. These observations have led to the definition and realization of stronger notions of encryption security, such as security against different forms of leakage [32,19,1,35,15,14,2,25,9], and key-dependent message (KDM) security [7,27,8,5,4,9,31,3,10]. Our goal is to construct schemes realizing these security properties from general assumptions. Our results concern a basic model of leakage, known as the *bounded-leakage model* [1] and a basic model of KDM security, known as *projection* security (which is slightly

* Work of the first two authors supported in part by an NSERC Discovery Grant, "Foundational studies in privacy and security", and the Simons Institute for the Theory of Computing program "Real analysis in computer science". Work of the second author was partly completed while a Visiting Member of the School of Mathematics at the Institute for Advanced Study. Part of this work was done while the third author was a Senior Visiting Research Fellow at the Centre for Quantum Technologies, NUS, Singapore.

stronger than *circular* security). We will also consider a model of *auxiliary-input security* [15,14]. We first provide some background on these models and then describe our results.

For all definitions below (unless otherwise stated) we assume we are encrypting the secret key (or functions thereof) bit-by-bit, i.e., the scheme is either bit encryption, or there is a mapping from bits to two fixed plaintext messages.

KDM security. KDM security is defined with respect to a function family F: informally, an encryption scheme (G, E, Dec) is F-KDM$^{(1)}$ secure if no adversary can distinguish between two oracles, where the first one, on input $f \in F$, returns $E_{pk}(f(sk))$ (for a random (pk, sk) chosen at the beginning), and the second one, regardless of the input, returns an encryption of a fixed message. A basic form of KDM$^{(1)}$ security is 1-circular security, allowing the adversary to obtain encryptions of any bit of the secret key. Another basic notion is projection security, which also allows the adversary to obtain encryptions of negations of secret key bits. KDM$^{(1)}$ security generalizes naturally to the case of multiple pairs of keys, giving rise to the notion of F-KDM$^{(n)}$-security, where in a system with the pairs of keys (pk_1, sk_1), $\ldots$, (pk_n, sk_n) a chosen function $f \in F$ comes with an index j, and as a result $f(sk_1, \ldots, sk_n)$ is encrypted under pk_j. For example, n-projection security allows the adversary to see encryptions of any bit of any secret key or its negation under (possibly) any other public key.

KDM security was originally defined by Black et al. [7], who built a *fully-KDM*-secure scheme (i.e., KDM-security with respect to all functions) in the random oracle model. In [8] Boneh et al. gave the first construction in the standard model, based on the DDH assumption, of a public-key scheme that was proved KDM$^{(n)}$ secure with respect to *affine functions*. This positive result led to a series of subsequent works, focusing on building affine-KDM$^{(n)}$ security under alternate specific assumptions (i.e., LPN/LWE [4], and QR/DCR and more generally *subgroup indistinguishability (SG)* assumptions [9]), and on developing *KDM-amplification* methods for transforming schemes with basic forms of KDM security into schemes with more sophisticated forms of KDM security [5,10,3]. These amplification methods in turn employ techniques such as garbled circuits [5], *randomized encoding of functions* [3] and *entropic-KDM* security [10] to enable KDM transformations. Most relevant to our work are the results of Applebaum [3], showing that, informally speaking, projection security is sufficient to obtain KDM security with respect to any fixed circuit family whose size is poly-bounded. Thus, a fundamental question regarding KDM security is to study general assumptions sufficient for realizing projection security, which is one of the main goals in our paper.

It turns out that realizing even 1-circular security for bit encryption is considerably more difficult than the case where the secret-key space is a subset of the plaintext space (so one can encrypt the whole key at once). In the latter case, through simple modifications to the encryption algorithm, one can make any CPA-secure scheme 1-circularly secure. Currently, the only constructions that provide bitwise 1-circular security are those of [8,4,10], which are based on specific assumptions. Also, it was shown in [41] that the implication that "any CPA-secure bit encryption scheme is also 1-circularly secure" is not provable using reductions that use both the adversary and the scheme in a blackbox way.[1] Moreover, under widely-believed assumptions, there exist CPA-secure bit-encryption schemes that are not 1-circularly secure [41,30].

Leakage resilience. Akavia et al. [1] introduce the notion of encryption security against bounded memory leakage, wherein an adversary (after seeing the public key) may obtain arbitrary information about the secret key, of the form $f(sk)$ for adaptively chosen f, as long as the total number of bits leaked does not exceed an a priori fixed quantity, ℓ. (We refer to the fraction $\ell/|sk|$ as the resilience rate.) They showed that Regev's scheme [39] and the identity based encryption scheme of [20], both under the LWE assumption, provide resilience rate $O(1/polylog(|sk|))$. Naor and Segev [35] showed how to obtain encryption schemes resilient to high leakage lengths (but with low resilience rates) from any hash-proof system [13] and how to obtain schemes with $(1 - o(1))$-resilience rates from d-linear assumptions; moreover, they showed that the circularly-secure scheme of [8] provides a $(1 - o(1))$ resilience rate. Brakerski and Goldwasser [9], under the subgroup indistinguishability assumption, implied in turn by the QR and DCR assumptions, showed how to obtain encryption schemes that are affine-KDM secure, with a $(1 - o(1))$ resilience rate.

[1] Note that this is different from asking whether CPA-secure bit encryption implies the existence of circularly-secure bit encryption.

Auxiliary-input security. In the auxiliary-input model [15,14] the adversary is given some side information of the form $h(pk, sk)$, and the goal is to guarantee security as long as recovering sk from $h(pk, sk)$ is sufficiently, computationally hard. For public-key encryption Dodis et al. [14] build schemes based on LWE and DDH (where their DDH-based scheme is a variant of [8]) secure against *subexponentially-hard-to-invert* functions. Brakerski and Goldwasser [9] present schemes with the same level of auxiliary-input security under the subgroup indistinguishability assumption.

1.1 Our results (assumptions and constructions)

As pointed our earlier, the only constructions of circularly-secure/projection-secure bit encryption (even 1-circular security) are based on specific assumptions [8,4,9]. Moreover, the schemes of [8,9], referred to as BHHO and BG henceforth, besides KDM security, also provide security against different forms of leakage (as shown in [14,35,9]). Therefore, a natural question is whether there exist more general constructions that encompass all these specific constructions.

We will try to answer these questions by building leakage-resilient, projection-secure encryption schemes from CPA-secure private-key schemes with some special properties, which we now informally describe. Then we will use this private-key encryption abstraction as a stepping stone toward obtaining our results under other primitives.

The first property is a generalized version of additive homomorphism, where homomorphism is required to hold also with respect to randomness (let Hom denote the associated function). The second property is what Bellare et al. [6] call *reproducibility*, requiring that given a message m_2, secret key sk_2 and ciphertext $c = E_{sk_1}(m_1; r)$, where sk_1, m_1 and r are unknown, one can efficiently obtain $E_{sk_2}(m_2; r)$, i.e., there is a way to efficiently transfer the randomness from one encryption to another, provided the secret key for the second encryption is known.[2] We denote this efficient computation by $Rep(c, m_2, sk_2)$. Note that if an encryption algorithm reveals its randomness in the clear, then reproducibility is trivially satisfied, e.g., the standard way of building CPA-secure private-key encryption from a pseudorandom function family F, defining encryption as $E_{sk}(m) = (r, F_{sk}(r) \oplus m)$, provides reproducibility. In fact, we will later use this idea to obtain our encryption primitive, based on the existence of *homomorphic weak pseudorandom functions*. Note that for homomorphism, we are assuming that the message and randomness spaces must form groups. For technical reasons, we will also require the following property: from any encryption $E_{sk}(b; r)$, for unknown sk, b, r, one can obtain $E_{sk}(1; 0)$, i.e., the encryption of bit 1 under key sk based on the identity element of the randomness group.[3] We see this as a form of *degenerate* homomorphism.

We introduce a construction C (formalized in Section 3 and sketched in Subsection 1.4) that transforms a private-key scheme with the stated properties into a public-key one and show the following result.

Theorem (informal). Assume that $\mathcal{E} = (G, E, Dec, Hom, Rep)$ is a CPA-secure private-key, bit-encryption scheme that is degenerate additively homomorphic and reproducible. Then the constructed scheme $\mathcal{E}' = C(\mathcal{E})$ is a public-key bit-encryption scheme that satisfies the following properties.

- For any integer n, by appropriately choosing the system parameters, $\mathcal{E}'$ is n-projection secure. (Formalized in Theorem 2)
- By appropriately choosing the system parameters, $\mathcal{E}'$ provides a $(1 - o(1))$-leakage resilience rate. (Formalized in Theorem 3)
- $\mathcal{E}'$ provides auxiliary-input leakage resilience against subexponentially-hard functions. (Formalized in Theorem 6 and Remark 2)

We will also discuss generalizations of the above construction to the case the base scheme is not bit-encryption.

[2] Both these conditions were used implicitly by Peikert and Waters as the main building blocks for their construction of lossy-trapdoor functions [37].

[3] The actual assumption we need is substantially weaker. However, we leave it this way for the sake of readability. In fact, under all concrete schemes we present, $E_{sk}(m; 0)$ depends only on m and is independent of sk.

1.2 Realizations

From homomorphic weak pseudorandom functions. Pseudorandom function families (PRFs) provide a convenient way of realizing reproducible CPA-secure private-key encryption via the standard PRF-based encryption construction. Towards providing homomorphism for a PRF-based scheme, we call a function family *homomorphic* if both the domain and range of the underlying functions form groups, and each function acts as a homomorphism. A standard PRF cannot, however, be homomorphic since with high probability a truly random function will not be homomorphic and an adversary with the power to (even) nonadaptively query a function oracle may easily exploit this fact. To prevent this type of attack, we work with *weak PRFs*, defined by Naor and Reingold [34], which allow an adversary to see values of the function only on a sequence of random inputs. Formally, f_k is *weakly pseudorandom* if no adversary can distinguish between $(d_1, f_k(d_1)), \ldots, (d_p, f_k(d_p))$ and $(d_1, r_1), \ldots, (d_p, r_p)$, where all d_i's and r_i's are chosen independently at random. As we see next, not only is the notion of homomorphic weak PRFs meaningful, it is naturally realizable under specific assumptions. We also note that the standard construction of private-key encryption from a PRF, when applied to homomorphic weak PRFs, results in a scheme that satisfies the properties we need from our base encryption primitive (Lemma 4).

For a DDH-hard group $\mathbb{G}$ with $o = |\mathbb{G}|$, define $F = \{f_k \colon \mathbb{G} \to \mathbb{G}\}_{k \in \mathbb{Z}_o}$ by $f_k(g) = g^k$. This function family was introduced and proved to be weakly pseudorandom by Naor, Pinkas and Reingold [33]; the proof of weak pseudorandomness uses standard techniques related to random-self-reducibility of DDH. The fact that f_k is homomorphic is clear. Interestingly, by plugging this PRF into our general construction, we obtain a scheme which is a close variant of the BHHO scheme. We also give a realization of weak homomorphic PRFs under *homomorphic hash-proof systems (HHPS)* [13]: here the PRF is simply the family of hash functions on *valid* points (Theorem 4). A corollary of our results is the following.

Corollary. Under the HHPS assumption and for any integer n, there exists a public-key encryption scheme that provides, at the same time, n-projection security and a $(1 - o(1))$-leakage resilience rate.

To the best of our knowledge, our results give the first HHPS-based encryption scheme that provides (even individually) n-projection security and a $(1 - o(1))$-leakage resilience rate. (See SubSection 1.4 for a comparison of our results with those of the recent work of [42].) Naor and Segev [35] show how to construct schemes with high tolerated leakage lengths (but low *rates* of leakage resilience) from any hash-proof system, and also how to obtain schemes with $(1 - o(1))$ leakage-resilience rates from *k-linear assumptions*. Our results can be thought of as complementing those of [35], by saying that if we add homomorphism to a HPS, we obtain schemes with high resilience rates. Hazay et al. [26] show how to obtain schemes withstanding high leakage lengths from any CPA-secure public-key encryption (which is the minimal assumption). Their construction, however, produces a scheme with low leakage-resilience rates, and does not imply our leakage resilience result based on HHPS.

From Subgroup indistinguishability. We show how to instantiate our encryption primitive under the the subgroup indistinguishability (SG) assumption [9], of which QR and DCR are special cases (Lemma 5). Our current formulation of homomorphic weak PRFs does not seem to be realizable under the SG assumption. It is, however, possible to formulate a more relaxed version of such PRFs, one that is still sufficient for realizing our encryption assumptions and is also realizable under the SG assumption. We choose not to pursue this direction since there is already an easy way to realize our encryption primitive under the SG assumption.

We provide a summary of our results in Figure 1.

1.3 KDM amplification and leakage resilience

We prove that Applebaum's KDM amplification method [3] for obtaining KDM-security for any fixed family of bounded circuits from projection security also preserves both types of leakage resilience (Theorem 9). We were not, however, able to show this for the KDM amplification methods of [5,10]. Applebaum's transformation has the key property that it only modifies the encryption and decryption algorithms of the base scheme, by applying *randomized encoding and decoding*, which are fixed mappings constructed based on the target

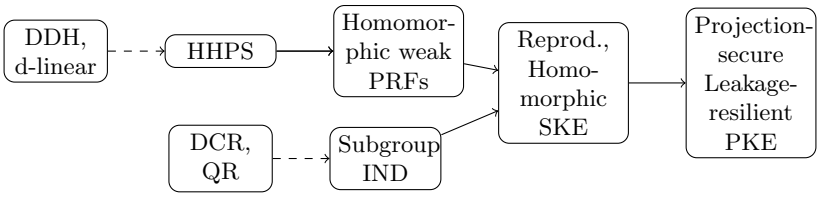

Fig. 1. Summary of results (dashed arrows indicate known implications)

function family, inside the encryption and decryption algorithms. This property facilitates reducing leakage resilience and auxiliary input security of the constructed scheme to the same requirements (i.e., with the same parameters) on the base scheme. As a corollary, for any fixed bounded function family F and any integer n, assuming the existence of private-key schemes with the stated properties, we obtain schemes that at the same time provide (1) F-KDM$^{(n)}$ security, (2) a $(1 - o(1))$-leakage resilience rate, and (3) auxiliary-input security against subexponentially-hard functions (Corollary 1).

1.4 Construction technique and further discussion

Construction and proof techniques. We now give a sketch of the construction, C, and proof techniques. Fix $\mathcal{E} = (G, E, D, Rep, Hom)$ to be a private-key bit-encryption scheme that provides reproducibility and the generalized homomorphism condition. The latter, using additive notation, states the following condition that $Hom(E_{sk}(b_1; r_1), E_{sk}(b_2; r_2)) = E_{sk}(b_1 + b_2; r_1 + r_2)$. (Note that because of our additive notation our message space is $\mathbb{Z}_2$, and 0 is the identity element of the randomness space.)

Under $\mathcal{E}' = C(\mathcal{E}) = (G', E', D')$, the secret key is a random string $\mathbf{s} \leftarrow \{0,1\}^l$ (for some poly l) and the public key is a tuple of ciphertexts

$$\mathbf{pk} = (E_{sk}(0; r_1), \ldots, E_{sk}(0; r_l), E_{sk}(0; \mathbf{s} \cdot \mathbf{r})),$$

where $sk, r_1, \ldots, r_l$ are generated randomly under $\mathcal{E}$ and and $(\cdot)$ denotes the inner product of $\mathbf{s}$ and $\mathbf{r} = (r_1, \ldots, r_l)$. In words, $\mathbf{pk}$ consists of $l + 1$ $\mathcal{E}$-encryptions of zero, where the first l encryptions are produced independently, while the randomness value used for the last encryption is a "subset-sum" of the previous ones based on $\mathbf{s}$. To encrypt a bit b we sample $sk' \leftarrow G(1^\lambda)$ and output $(E_{sk'}(0; r_1), \ldots, E_{sk'}(0; r_l), E_{sk'}(b; \mathbf{s} \cdot \mathbf{r}))$, which can be computed from $\mathbf{pk}$ by applying Rep component-wise. To decrypt $(c_1, \ldots, c_l, c_{l+1})$ under $\mathbf{s}$, we return 0 iff $c_{l+1} = Hom_{\mathbf{s}}(c_1, \ldots, c_l)$, where $Hom_{\mathbf{s}}(c_1, \ldots, c_l)$ "sums" those ciphertexts c_i where $\mathbf{sk}_i = 1$. The correctness of decryption follows.

Some notes are in order. Firstly, under G', the secret key of the old scheme, sk, is used *only* to compute the encryptions needed to form $\mathbf{pk}$. Roughly, the fact that $\mathbf{s}$ is independent of sk underlies the circular security of $\mathcal{E}'$. Secondly, E' has the somewhat unusual property that it calls G, with the returned values comprising all the randomness used in encryption.

As a warm-up we first discuss CPA security of $\mathcal{E}'$. Consider a malformed public key $\mathbf{pk}_{mal}$ with r_{l+1} chosen independently at random (instead of being $\mathbf{s} \cdot \mathbf{r}$). CPA security under $\mathbf{pk}_{mal}$ reduces to showing $(\mathbf{pk}_{mal}, \mathbf{c}_0) \equiv^c (\mathbf{pk}_{mal}, \mathbf{c}_1)$, where $\equiv^c$ denotes computational indistinguishability, and

$$\mathbf{c}_b = (E_{sk'}(0; r_1), \ldots, E_{sk'}(0; r_l), E_{sk'}(b; r_{l+1})$$

This in turn follows by appealing to the CPA security and reproducibility of $\mathcal{E}$. To complete the CPA-security proof, it would suffice to argue that a malformed public-key is indistinguishable from a valid one, which follows information theoretically (from the *leftover hash lemma*) if l is large enough. Below we extend the arguments given here to argue about KDM and leakage-resilience security of the scheme.

KDM security. A main idea used in the proof of 1-circular security (for simplicity) is that if one possesses $\mathbf{s}$, then the encryption of a bit b may be equivalently computed as $\mathbf{c} = (c_1, \ldots, c_l, Hom(c_{i_1}, \ldots, c_{i_w}, c'))$, where

$sk' \leftarrow G(1^\lambda)$, $c_j \leftarrow E_{sk'}(0)$ for $1 \leq j \leq l$, $(i_1, \ldots, i_w)$ are the indices of nonzero bits of $\mathbf{s}$ and $c' = E_{sk'}(b; 0)$ (i.e., c' is the encryption of b where the randomness value is fixed to the group identity 0.) Now we consider an intermediate hybrid, W_1, in which to encrypt the hth bit of $\mathbf{s}$, we return $(c_1, \ldots, c_l, Hom(c_{i_1}, \ldots, c_{i_w}, c'))$, where now c_h is an encryption of 1, but every other c_j is an encryption of 0 (and c' is an encryption of $\mathbf{s}_h$ under the identity randomness). We will show that W_1 provides a view computationally indistinguishable from the real view, W_0; the main idea is that any distinguisher between W_0 and W_1 can be reduced to an adversary $\mathcal{A}$ that wins in a special vector-encryption game (performed under $\mathcal{E}$), in which $\mathcal{A}$ may adaptively issue fixed-length vectors of bits (of a certain form), and in response to each vector query $\mathbf{v}$, either $\mathbf{v}$ or the all-zero vector (depending on the challenge bit) is component-wise encrypted under a fresh secret key, but by reusing randomness across each fixed component of vectors (that is the ith component of each vector is always encrypted under a fixed random r_i). In Lemma **??** we show any $\mathcal{A}$ has a negligible advantage under this game, and use this to prove the indistinguishability of W_0 and W_1. (It turns out this last step also requires us to use degenerate homomorphism to compute $E_{sk'}(1; 0)$ obliviously to sk'.) Having proved the indistinguishability of W_0 and W_1 we notice that under W_1 the reply to "encrypt the hth bit of $\mathbf{s}$" is indeed formed as $(E_{sk'}(0; r_1), \ldots, E_{sk'}(0; r_{h-1}), E_{sk'}(1; r_h), E_{sk'}(0; r_{h+1}), \ldots, E_{sk'}(0; r_l), E_{sk}(0; \mathbf{s} \cdot \mathbf{r}))$, and in particular is independent of $\mathbf{s}$ beyond $\mathbf{s} \cdot \mathbf{r}$, which makes the rest of the proof follow smoothly using ideas described for the CPA case.

The described techniques might be called *simulated KDM encryptions*, originally introduced in [8], used also in subsequent works [4,9], which show how to simulate KDM responses under public information. The main challenge in our setting is how to enable such properties under our general assumptions.

Leakage resilience. For simplicity, we first outline the idea of the proof for the case of nonadaptive leakage resilience (that is, the function f is queried before the public key being published). To argue about nonadaptive leakage resilience, one has to show $D_0 \equiv^c D_1$, where $D_b = (\mathbf{pk}, \mathbf{c}_b, f(\mathbf{s}))$, and

$$\mathbf{c}_b = (E_{sk'}(0; r_1), \ldots, E_{sk'}(0; r_l), E_{sk'}(b; \mathbf{s} \cdot \mathbf{r}))$$

Now since f is chosen independently of $\mathbf{pk}$, it is also independent of $\mathbf{r}$, which allows us to apply the average-case version of the leftover hash lemma [16] (considering the inner product acts as a universal hash function) to replace $\mathbf{s} \cdot \mathbf{r}$ with a totally random r_{l+1}; the rest of the proof follows from the fact that $\mathcal{E}$ allows secure reuse of randomness. For the adaptive case, to handle the issue that f depends on $\mathbf{pk}$ (and so we cannot apply random extraction directly), we use similar techniques to those used by [35]: we consider a hybrid D'_b, which is similar to D_b, but in which the first l bits encrypted under sk' are independently random bits $b_1, \ldots, b_l$ (as opposed to zeros) and that the last bit is $\mathbf{s} \cdot (b_1, \ldots, b_l) + b$. By proving $D'_b \equiv^c D_b$, for both $b \in \{0, 1\}$, (essentially using reproducibility and semantic security of $\mathcal{E}$) we can now apply the generalized leftover hash lemma by taking $(b_1, \ldots, b_l)$ as the seed, $\mathbf{s}$ as the source and considering that b_i's are chosen independently of f and $\mathbf{r}$; this allows us to replace $\mathbf{s} \cdot (b_1, \ldots, b_l)$ with a uniformly random bit, proving D'_0 is statistically close to D'_1. The leakage resilience proof follows. The proof for the auxiliary-input case essentially follows the same line of arguments, except for replacing randomness extraction with pseudorandomness extraction [22]. We refer the reader to the full proof.

Final remarks. Instantiating the above construction using homomorphic weak PRFs provides an improvement in efficiency, matching the same level of efficiency as [8] if the base PRF (in turn) is instantiated under the corresponding assumption. Technically, in this case, it would suffice to define the public key to be $(d_1, \ldots, d_l, \mathbf{s} \cdot (d_1, \ldots, d_l))$, i.e., instead of putting the whole ciphertext in each component, we only give the underlying randomness, which would have been given out by the ciphertext itself in the clear. Also, to encrypt m under $\mathbf{pk} = (d_1, \ldots, d_l, d_{l+1})$, we simply output $(F_{sk}(d_1), \ldots, F_{sk}(d_l), F_{sk}(d_{l+1}) + m)$, where sk is a fresh PRF key.

While our results enable us to explain those of [8,9,35], regarding KDM security and leakage resilience of the BHHO and BG schemes, they suffer from the same limitations as those of [9], in that, in order to achieve KDM$^{(n)}$ security, we must choose the parameters of our constructed scheme based on n. Boneh et al. [8] get around this dependency by using the *random self-reducibility* of DDH and strong key-homomorphism properties of DDH-based schemes. Similar dependencies for (even specific) non-DDH-based assumptions

occur in other settings as well, e.g., [11]. We leave it as an open problem to resolve this dependency. We should also mention that the BHHO and BG schemes were proved affine-KDM secure; under the current assumptions, we were not able to extend our results to the affine-KDM setting. Finally, we note that just the fact that we can build a CPA-secure (as opposed to KDM secure) public-key scheme from our private-key assumptions is not unheard of since even weaker forms of homomorphism are known to be sufficient to bridge this gap [40].

Comparison with [42]. Concurrently with our work, Wee [42] recently showed that the original HHPS-based encryption scheme of Cramer and Shoup [13] provides F-KDM$^{(1)}$ security, where F is a function class defined based on the underlying hash functions. (Specifically, following notation in Subsection 6.2, $F = \{f_{\mathsf{c},\mathsf{k}} \colon \mathsf{SK} \mapsto \mathsf{K}\}$, where $f_{\mathsf{c},\mathsf{k}}(sk) = \Lambda_{\mathsf{sk}}(\mathsf{c}) + \mathsf{k}$.) We note that the basic KDM setting of [42] is different from ours in that we are concerned with KDM-security with respect to bit-projections of the secret key. Nevertheless, by instantiating that framework under specific DDH/SG-based HHPSs, [42] obtains schemes that are close variants of BHHO and BG. Moreover, the results of [42] also explain the bit-affine-security of BHHO and BG, while our results only explain the projection security. On the other hand, we obtain HHPS-based schemes that are n-projection secure, while the results of [42] do not seem to extend to the multiple-key setting (as noted there). Moreover, by using an encryption-based primitive as our base assumptions, we are able to obtain generic constructions under homomorphic weak PRFs, that is a weaker abstraction than the HHPS, as we show.

Other related work. Choi and Wee [12] show how to construct *lossy trapdoor functions* from homomorphic reproducible encryption by abstracting the matrix-based construction of Peikert and Waters [37]. This shows one more application of homomorphic weak PRFs as a general primitive. We mention, however, that the main difference between our constructions and those of [37,12] is that in [37,12] the trapdoor key of the constructed schemes consists of secret keys produced under the base scheme, while in our setting, the main challenge (and novelty) is to come up with a construction whose encryption function still somehow calls that of the base scheme (in order to inherit its security), but in such a way that the secret keys of the base scheme are not included in the constructed secret key.

2 Definitions

2.1 Standard Notation and Definitions

For a finite set S we use $x \leftarrow S$ to denote sampling x uniformly at random from S and denote by U_S or $U(S)$ the uniform distribution on S. If D is a distribution then $x \leftarrow D$ denotes choosing x according to D. We denote the support set of a distribution D by $Sup(D)$, and write $x \in D$ to indicate $x \in Sup(D)$. The notions of computational indistinguishability and statistical indistinguishability are standard. We use $\equiv^c$ to refer to computational indistinguishability, $\equiv^s$ to statistical indistinguishability and $\equiv$ to *identity* of two distributions. We use the term PPT in this paper in the standard sense. We will often omit the adjective PPT/efficient when discussing functions – by default we assume all such functions are efficient.

We denote the length of $x \in \{0,1\}^*$ by $|x|$ and the ith bit of x, for $1 \leq i \leq |x|$, by x_i. We denote the n-th Cartesian power of a set S by S^n. We call $f : \mathbb{N} \to \mathbb{R}$ negligible if $f(\lambda) < 1/P(\lambda)$, for any poly P and sufficiently large λ.

All groups are assumed to admit efficient group operations, and to be commutative, but not necessarily cyclic, unless otherwise indicated.

2.2 Syntax of encryption schemes

We first start with some notation. We use $A(a_1, a_2, \ldots; r)$ to denote the deterministic output of randomized function A on inputs $a_1, a_2, \ldots$ and randomness r, and use $x \leftarrow A(a_1, a_2, \ldots)$ to denote the distribution formed by first choosing r uniformly at random and then outputting $A(a_1, a_2, \ldots; r)$.

We assume that all cryptographic primitives (encryption, PRFs, etc) discussed in this paper, besides their usual algorithms, have a *parameter-generation* algorithm that produces *public parameters* (e.g., a group) used

by all other algorithms. In situations where we talk about generating many keys it should be understood that all keys are sampled under the same public parameters, which were generated randomly at the beginning. We now give the syntax of encryption schemes.

A *public-key encryption scheme* $\mathcal{E}$ is given by algorithms $(Param, G, E, Dec)$, all taking as input a *security parameter* 1^λ (that we make it explicit for $Param$ and G and implicit for other algorithms.) $Param$ takes input 1^λ, and outputs a public parameter, par. The *key-generation* algorithm, takes 1^λ and par and outputs public/secret keys, $(pk, sk) \leftarrow G(1^\lambda, par)$. The *encryption* algorithm E, takes a public key pk, a plaintext $m \in \mathcal{M}_\lambda$ (where $\mathcal{M}_\lambda$ is the *plaintext space*) and randomness $r \in \mathcal{R}_\lambda$ (where $\mathcal{R}_\lambda$ is the *randomness space*), and deterministically produces ciphertext $c = E_{pk}(b; r)$. Finally, the *decryption* algorithm takes a secret key sk and ciphertext c, and deterministically outputs $m = Dec_{sk}(c)$. For correctness, we require, for every $par \in Param(1^\lambda)$, $(pk, sk) \in G(1^\lambda, param)$, every m and $c \in E_{pk}(m)$, that $Dec_{sk}(E_{pk}(m)) = m$. We typically use $\mathcal{PK}_\lambda$ and $\mathcal{SK}_\lambda$ to refer to the public-key and secret-key spaces. Formally, $(\mathcal{PK}_\lambda, \mathcal{SK}_\lambda) = Sup(G(1^\lambda))$. We make the inclusion of $Param$ implicit henceforth.

2.3 Key-dependent-message security

In this paper we consider encryption schemes, whose generated secret keys are always bitstrings, but whose plaintext space may or may not be the single-bit space, e.g., it may be a group space. For the latter case, in order to make the notion of bitwise encryption of the secret key meaningful, we assume that a fixed mapping $(\{0,1\} \to \mathcal{M}_\lambda)$ is already in place. In the following, when we say $E_{pk}(b)$, where b is a bit, if E is a bit encryption algorithm, then we are encrypting the actual bit b, and otherwise, we are encrypting the element that b is mapped to. We now proceed to describe the notion of $\text{KDM}^{(n)}$ security for an arbitrary encryption scheme $\mathcal{E} = (G, E, Dec)$ (bit encryption or otherwise).

Assume that $F = \{F_\lambda\}_{\lambda \in \mathbb{N}}$ is an ensemble of sets of functions, where for each $f \in F_\lambda$, it holds that $f : \mathcal{SK}_\lambda^n \to \{0,1\}$.

We define $F\text{-KDM}^{(n)}$ security through the following $F\text{-KDM}^{(n)}$ game, played between a challenger and an adversary. The challenger first chooses $b \leftarrow \{0,1\}$, generates $(pk_1, sk_1), \ldots, (pk_n, sk_n) \leftarrow G(1^\lambda)$, and gives $pk_1, \ldots, pk_n$ to the adversary. The adversary $\mathcal{A}$, given pk_i's, can repeatedly and adaptively, for $1 \le i \le n$, make queries of the form (i, f), where $f \in F_\lambda$, or of the form (i, m), where $m \in \mathcal{M}_\lambda$, and in return,

- If $b = 0$, the challenger returns $E_{pk_i}(f(sk_1, \ldots, sk_n))$ in response to (i, f) and $E_{pk_i}(m)$ in response to (i, m); and
- If $b = 1$, the challenger returns $E_{pk_i}(0)$.

$\mathcal{A}$ finally outputs a bit b'. We define the $F\text{-KDM}^{(n)}$ *advantage of* $\mathcal{A}$ as

$$Adv^{F\text{-KDM}^{(n)}}(\mathcal{A}) = \left|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]\right|,$$

where the probabilities are computed over the coins of $\mathcal{A}$ and of the challenger.

We say that $\mathcal{E}$ is $F\text{-KDM}^{(n)}$-secure if for any $\mathcal{A}$ in the above game, it holds that $Adv^{F\text{-KDM}^{(n)}}(\mathcal{A}) = negl$.

Assume $\mathcal{SK}_\lambda = \{0,1\}^{l(\lambda)}$ and let $l = l(\lambda)$. For $1 \le i \le n$ and $1 \le j \le l$, define $Sel_{i,j} \colon \mathcal{SK}_\lambda^n \mapsto \{0,1\}$ to be the function that on input $(sk_1, \ldots, sk_n)$ returns the jth bit of sk_i. Similarly, define $NSel_{i,j}$ to be the function that on input $(sk_1, \ldots, sk_n)$ returns the negation of the jth bit of sk_i. Finally, define $S_\lambda = \{Sel_{i,j} \colon 1 \le i \le n, 1 \le j \le l\}$ and $\hat{S}_\lambda = \{NSel_{i,j} \colon 1 \le i \le n, 1 \le j \le l\}$. We now give the following definitions.

- We call $\mathcal{E}$ *n-circularly secure* if $\mathcal{E}$ is $F\text{-KDM}^{(n)}$ secure, where $F_\lambda = S_\lambda$.
- We call $\mathcal{E}$ *n-projection secure* if $\mathcal{E}$ is $F\text{-KDM}^{(n)}$ secure for $F_\lambda = S_\lambda \cup \hat{S}_\lambda$.

Semantic security for private-key encryption. For a private-key encryption scheme (G, E, Dec) it is convenient to work with the following definition of CPA security. (1) The challenger chooses $b \leftarrow \{0,1\}$ and private key $sk \leftarrow G(1^\lambda)$. (2) The adversary submits a sequence of messages $(m_1, \ldots, m_p)$, where $p = p(\lambda)$ is

an arbitrary function. (3) The challenger returns $(E_{sk}(m_1), \ldots, E_{sk}(m_p))$ if $b = 0$, and $(E_{sk}(0), \ldots, E_{sk}(0))$, otherwise. (4) The adversary returns a bit b'. We define the CPA-security advantage of the adversary as

$$\left|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]\right|,$$

and call the scheme CPA secure if all adversaries have negligible advantage.

2.4 Leakage resilience

We define the notion of leakage resilience. For $\mathcal{L} = \mathcal{L}(\lambda)$, we say that the public-key encryption scheme $\mathcal{E} = (G, E, Dec)$ is *$\mathcal{L}$-length leakage resilient* if, for any adversary $\mathcal{A}$, the $\mathcal{L}$-leakage-advantage of $\mathcal{A}$, $Adv^{\mathcal{L}\text{-}leak}(\mathcal{A})$, defined via the following game, is negligible.

- Setup: The challenger generates $(pk, sk) \leftarrow G(1^\lambda)$ and gives pk to $\mathcal{A}$.
- Leakage queries: $\mathcal{A}$ sends function $f : \mathcal{SK}_\lambda \to \{0,1\}^*$ to the challenger, where $|f(sk)| \leq \mathcal{L}$, and receives, in response, $f(sk)$.
- Challenge: $\mathcal{A}$ submits $(m_0, m_1) \in \mathcal{M}_\lambda^2$, and the challenger, samples $b \leftarrow \{0,1\}$, and returns $E_{pk}(m_b)$ to $\mathcal{A}$. Finally, $\mathcal{A}$ returns an output bit b'.

We define $Adv^{\mathcal{L}\text{-}leak}(\mathcal{A}) = \left|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]\right|$. We say that $\mathcal{E}$ is *r-rate leakage resilient* (or has *resilience rate* r) if $\mathcal{E}$ is $r \cdot \log|\mathcal{SK}|$-length leakage resilient.

Finally, we note that restricting $\mathcal{A}$ in the above game to a single leakage query is without loss of generality. In particular, the security definition does not become stronger if $\mathcal{A}$ is allowed to adaptively make multiple leakage queries provided that the total length of the bits leaked is bounded by $\mathcal{L}(\lambda)$. The proof of this fact is straightforward; see [1] for a proof.

2.5 Properties of the base scheme

We give the definitions of the main properties that we need from the base private-key encryption scheme.

Definition 1. *A private-key encryption scheme $\mathcal{E} = (G, E, Dec)$ provides* reproducibility *(or is* reproducible*) if there is an efficient function Rep such that for any $sk, sk' \in G(1^\lambda)$, $r \in \mathcal{R}_\lambda$ and $m_1, m_2 \in \mathcal{M}_\lambda$,*

$$Rep(E_{sk}(m_1; r), m_2, sk') = E_{sk'}(m_2; r).$$

Definition 2. *Let $\mathcal{E} = (G, E, Dec)$ be a private key encryption scheme where both $(\mathcal{R}_\lambda, +)$ and $(\mathcal{M}_\lambda, +)$ form groups. Then $\mathcal{E}$ is* additively homomorphic with respect to plaintexts and randomness (PR-additively homomorphic) *if there is an efficient function Hom such that for every $sk \in G(1^\lambda)$, $m_1, m_2 \in \mathcal{M}_\lambda$, and $r_1, r_2 \in \mathcal{R}_\lambda$,*

$$Hom\left(E_{sk}(m_1; r_1), E_{sk}(m_2; r_2)\right) = E_{sk}(m_1 + m_2; r_1 + r_2).$$

We extend the notation of $Hom(\cdot)$ to define $Hom(c_1, \ldots, c_m)$ in the straightforward way. For technical reasons, we also need the following condition: for any sk, m, r and m', given only m' and $E_{sk}(m; r)$, we can form the ciphertext $E_{sk}(m', 0)$, where 0 denotes the identity element of $\mathcal{R}_\lambda$. We sometimes refer to this property as the degenerate condition.

Henceforth, when discussing encryption schemes, we will use "homomorphic" as shorthand for "PR-additively homomorphic."

3 Construction

We first fix some notation. Throughout this section we will be working with additive notation for groups with 0 denoting the identity element. For $\mathbf{g} = (g_1, \ldots, g_p) \in \mathbb{G}^p$ and $\mathbf{b} = (b_1, \ldots, b_p) \in \{0,1\}^p$ we define $\mathbf{b} \cdot \mathbf{g} = b_1 \cdot g_1 + \cdots + b_p \cdot g_p \in \mathbb{G}$, where, $0 \cdot g = 0$, and for $n \in \mathbb{N}$, we define $n \cdot g = g + (n-1) \cdot g$.

We present a generic construction that transforms a reproducible, homomorphic private-key encryption scheme into a public-key bit-encryption scheme. This always produces a bit-encryption scheme even if the base scheme is not. In the full version we show how to adjust the construction, to maintain the plaintext space, at the cost of additional syntactic assumptions (which are satisfied by our specific instantiations).

We then show (Section A.1) how to adjust the construction, to maintain the plaintext space, at the cost of additional syntactic assumptions (which are satisfied by our specific instantiations). For simplicity, we present (and prove the security of) the bit-encryption construction for the case where the base scheme is also bit encryption. We then discuss adaptations to the general case in the appendix, Remark 1.

Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure private-key bit-encryption scheme providing reproducibility (with the associated function *Rep*) and homomorphism (with the associated function Hom). Recall for homomorphism, both the message space, $\{0,1\}$, and the randomness space, $\mathcal{R}_\lambda$, form groups, which implies the plaintext group is just $\mathbb{Z}_2$. We now present the construction.

Construction 1 *(Single bit encryption): Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be as above and let $l = l(\lambda)$ be a value that we instantiate later.*

- *Key generation G': Choose the secret key as $\mathbf{s} \leftarrow \{0,1\}^l$ and the public key as $(E_{sk}(0; r_1), \ldots, E_{sk}(0; r_l), E_{sk}(0; \mathbf{s} \cdot \mathbf{r}))$, where $sk \leftarrow G(1^\lambda)$, $r_1, \ldots, r_l \leftarrow \mathcal{R}_\lambda$ and $\mathbf{r} = (r_1, \ldots, r_l)$.*
- *Encryption E': To encrypt bit b under public key $(c_1, \ldots, c_l, c_{l+1})$, do the following: choose $sk' \leftarrow G(1^\lambda)$ and return $(c'_1, \ldots, c'_l, c'_{l+1})$, where $c'_i = Rep(c_i, 0, sk')$, for $1 \le i \le l$, and $c'_{l+1} = Rep(c_{l+1}, b, sk')$.*
- *Decryption Dec': To decrypt $(c'_1, \ldots, c'_l, c'_{l+1})$ under secret key $\mathbf{s}$, letting $(i_1, \ldots, i_w)$ be the indices of non-zero bits of s, output 0 if $c'_{l+1} = Hom\left(c'_{i_1}, \ldots, c'_{i_w}\right)$, and 1 otherwise.*

The completeness of the scheme follows immediately. A few comments are in order. First, the encryption algorithm of the constructed scheme uses that of the base scheme, but by reusing the randomness values of the ciphertexts given in the public key. Second, the constructed decryption function does not need any secret keys of the base scheme, e.g., sk, for its computation. Roughly, this is why proving circular security for the constructed scheme should not be much harder than proving CPA security. In our security proofs, we will rely on the fact that we may use the homomorphism properties of the base primitive to form public keys and encryptions in alternate, equivalent ways as described below.

Proposition 1 *1. The public key may be computed as $(c_1, \ldots, c_l, c_{l+1})$, where $c_i \leftarrow E_{sk}(0)$, for $1 \le i \le l$, and $c_{l+1} = Hom\,(c_{i_1}, \ldots, c_{i_w})$, where $(i_1, \ldots, i_w)$ are the indices of non-zero bits of $\mathbf{s}$.*

2. Let $\mathbf{s}$, sk' and $c'_1, \ldots, c'_l$ be as in the definition of encryption in Construction 1. Then, c'_{l+1} may be computed as $c'_{l+1} = Hom(c_{i_1}, \ldots, c_{i_w}, E_{sk'}(b; 0))$, where $(i_1, \ldots, i_w)$ are the indices of non-zero bits of $\mathbf{s}$.

4 Proof of projection security

In this section we give the proof of projection security of our constructed scheme. For clarity of exposition, we present the results with respect to asymptotic security, without specifying the exact advantage functions. This section is organized as follows. In Subsection 4.1 we reviews some facts related to entropy which are needed by our proofs. In Subsection 4.2 we introduce an intermediate lemma that will be used in the proofs of our main theorems. Finally, in Subsection 4.3 we give the proof for projection security.

4.1 Information-theoretic tools

We denote the *min-entropy* of a distribution D by $H_\infty(D)$, defined as $H_\infty(D) = \min_{d \in D} \left[\log(\frac{1}{\Pr[D=d]})\right]$. We also need to work with the notion of *average min entropy*, formalized by Dodis et al. [16], which measures the expected unpredictability of X given a random value y of Y. Formally,

$$\tilde{H}_\infty(X|Y) = -\log\left(E_{y \leftarrow Y}(2^{-H_\infty(X|Y=y)})\right) = -\log\left(E_{y \leftarrow Y}(\max_x \Pr[X=x|Y=y])\right).$$

A well-known fact about average-min entropy is a special form of the *chain rule*, saying that conditioning on a random variable Y, the average min entropy decreases by at most the logarithm of the support size of Y.

Lemma 1. *([16]) For any (X, Y, Z) it holds that $\tilde{H}_\infty(X|Y,Z) \geq \tilde{H}_\infty(X|Z) - \log|Sup(Y)|$.*

A family of functions $\{h : D \to R\}_{h \in H}$ is called *universal* if for all $x_1, x_2 \in D$, with $x_1 \neq x_2$, it holds that

$$\Pr_{h \leftarrow H}[h(x_1) = h(x_2)] \leq \frac{1}{|R|}.$$

We typically denote a family of functions $\{h : D \to R\}_{h \in H}$ as a single function $\mathcal{H} : D \times H \to R$, where $\mathcal{H}(d, h) = h(d)$. We have the following fact, showing that universal hash functions are good *average-case extractors*.

Lemma 2. *([16]) If $Ext : \{0,1\}^n \times W \to W'$ is a family of universal hash functions, then for any pair of random variables (D, X), where D takes values in $\{0,1\}^n$, it holds that*

$$\Delta\left((Ext(D,S), S, X), (R, S, X)\right) \leq 1/2\sqrt{2^{-\tilde{H}_\infty(D|X)}|W'|},$$

where S is uniform over W, R is uniform over W' and Δ denotes statistical distance. We stress that S is independent of (D, X).

4.2 A Useful lemma

We begin by introducing a game that will be used in proving our main results. Intuitively, the following experiment corresponds to a vector-encryption game, in which an adversary may interactively issue vectors of bits (of certain forms) to be encrypted, and each vector is component-wise encrypted under a fresh secret key while reusing randomness across each fixed component of vectors.

The randomness-sharing (RS) Game. Let (G, E, Dec) be a private-key bit-encryption scheme. As some notation, for $l \in \mathbb{N}$, we let $\mathbf{e}_i^l$, for $1 \leq i \leq l$, be the the vector of size l which has 1 in the ith position and 0 everywhere else, and $\mathbf{e'}_i^l$, for $1 \leq i \leq l$, be the vector of size l which has 1 in both its ith position and last position, and 0 everywhere else. We let $\mathbf{0}^l$ be the all-0 vector of size l. Finally, for $\mathbf{b} = (b_1, \ldots, b_l)$ and $\mathbf{r} = (r_1, \ldots, r_l)$, we define $E_{sk}(\mathbf{b}; \mathbf{r}) = (E_{sk}(b_1; r_1), \ldots, E_{sk}(b_l; r_l))$.

The game is parameterized over $l = l(\lambda)$ and $n = n(\lambda)$ and is played as follows.

The challenger chooses $b \leftarrow \{0,1\}$ and for each $1 \leq h \leq n$ it samples $\mathbf{r}_h = (r_{h1}, \ldots, r_{hl}) \leftarrow \mathcal{R}_\lambda^l$. Then the game proceeds as follows: the adversary repeatedly and adaptively makes queries of the form $(h, \mathbf{e})$, where $1 \leq h \leq n$ and $\mathbf{e} \in \{\mathbf{0}^l\} \cup \{\mathbf{e}_1^l, \ldots, \mathbf{e}_l^l\} \cup \{\mathbf{e'}_1^l, \ldots, \mathbf{e'}_l^l\}$, and in response to each such query, the challenger samples $sk \leftarrow G(1^\lambda)$ (using fresh coins for each query) and returns $E_{sk}(\mathbf{e}; \mathbf{r}_h)$ if $b = 0$, and $E_{sk}(\mathbf{0}^l; \mathbf{r}_h)$, otherwise. Finally, the adversary outputs a bit b' and its advantage is defined as:

$$Adv^{\text{p-rs}}(\mathcal{A}) = \Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1].$$

We now give the following lemma.

Lemma 3. *Assume $\mathcal{E} = (G, E, Dec, Rep)$ is a CPA-secure, private-key bit-encryption scheme that provides reproducibility. For any polynomial functions $l(\cdot)$ and $n(\cdot)$, and any adversary $\mathcal{A}$ in the (l, n)-RS game has a negligible advantage.*

Proof. For convenience we drop the security parameter from all sets throughout the proof. Using a simple hybrid argument we can show that any adversary that has advantage ϵ against the (l, n)-RS game can be reduced to an adversary with advantage ϵ/n against the $(l, 1)$-RS game. So we assume $n = 1$.

First, we introduce the following notation. For $\mathbf{b} = (b_1, \ldots, b_l)$ and $\mathbf{c} = (c_1, \ldots, c_l)$, define

$$Rep(\mathbf{c}, \mathbf{b}, sk) = (Rep(c_1, b_1, sk), \ldots, Rep(c_l, b_l, sk)) .$$

Assuming that $\mathcal{A}$ makes $t = t(\lambda)$ queries $\mathbf{q}_1, \ldots, \mathbf{q}_t$ we define the hybrid W_i, for $1 \leq i \leq t + 1$, as follows: first generate randomness vector $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}^l$ and respond to queries as follows: in response to the j'th query, for $1 \leq j < i$, generate $sk_j \leftarrow G(1^\lambda)$ and return $E_{sk_j}(\mathbf{q}_j; \mathbf{r})$ (i.e., encryption of the actual vector); and in response to the w'th query, for $w \geq i$, generate $sk_w \leftarrow G(1^\lambda)$ and return $E_{sk_w}(\mathbf{0}^l; \mathbf{r})$ (i.e., encryption of the all-zero vector). Note that W_1 and W_{t+1} match exactly the view of the adversary produced under the the RS game when $b = 1$ and $b = 0$, respectively. Thus, for the rest of the proof, we show how to reduce an adversary that can distinguish between W_i and W_{i+1}, for some $1 \leq i \leq t$, to an adversary against the CPA security game; the whole proof then follows using a standard hybrid argument.

Assume that $\mathcal{A}'$ can distinguish between W_i and W_{i+1} with a non-negligible advantage. Noting that W_i and W_{i+1} only differ in the way that the answer to the ith query is made, and that each query vector can take at most $2l + 1$ different values, we guess the ith query vector (that is going to be issued by $\mathcal{A}'$), call the LOR-CPA oracle, which is parameterized over an unknown secret key, on the guessed vector to receive $\mathbf{c} = (c_1, \ldots, c_l)$, and start simulating $\mathcal{A}'$ as follows: in response to the j'th query, $\mathbf{q}_j$, for $1 \leq j < i$, we generate $sk_j \leftarrow G(1^\lambda)$ and return $Rep(\mathbf{c}, \mathbf{q}_j, sk_j)$; in response to the ith query we return $\mathbf{c}$ (if our guess for $\mathbf{q}_i$ was incorrect, we stop and return a random bit); and in response to the w'th query, $\mathbf{q}_w$, for $w > i$, we generate $sk_w \leftarrow G(1^\lambda)$ and return $Rep(\mathbf{c}, \mathbf{0}^l, sk_w)$. Now it is easy to see that, if our guessing for the ith query was correct, depending on whether the CPA-challenge bit was zero or one, the resulting experiment matches exactly either W_i or W_{i+1}. This completes the proof. □

4.3 Proof of projection security

We first give the proof of 1-projection security of our scheme and then present a proof for n-projection security. Our proofs build on techniques from [9], which in turn generalize the DDH-based techniques of [8].

Theorem 1. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure private-key bit-encryption scheme providing degenerate homomorphism and reproducibility.Then, by taking $l = l(\lambda) = \omega(\log \lambda) + \log(|\mathcal{R}_\lambda|)$, the scheme built in Construction 1 is 1-projection secure.*

Proof. To represent the 1-projection game more concisely, we denote:
- *enc-secret*(i) encrypt the ith bit of the secret key; and
- *enc-secret*$(\bar{i})$ encrypt the negation of the ith bit of the secret key.

We introduce a series of hybrid games and show no adversary can distinguish between any two adjacent games. The first game corresponds to the real-encryption circular-security game, while the last game is the one where we always encrypt 0. Letting x_i be the adversary's output in *Game-i*, we write *Game-i* $\equiv^G$ *Game-j* to indicate $|\Pr[x_i = 1] - \Pr[x_j = 1]| = negl$. In all these games, whenever we write, say, $sk' \leftarrow G(1^\lambda)$ we mean that sk' is chosen freshly, so we may keep using the same variable sk' inside each game whenever we are producing a new key. Let $\mathcal{R} = \mathcal{R}_\lambda$ for the following discussion. Also, recall the notation $E_{sk}(\mathbf{b}, \mathbf{r})$ introduced in Subsection 4.2. Below we write $\mathbf{e}_i$ as shorthand for $\mathbf{e}_i^l$.

Game-0: real encryption. This game provides the adversary with a view that is identical to that under the projection security game in which the challenge bit is zero. The identical view is produced by using the algorithm Hom to produce the public key and to reply to encryption queries. (See Proposition 1.)

Generate $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}^l$ and $\mathbf{s} \leftarrow \{0, 1\}^l$ and let $(i_1, \ldots, i_w)$ be the indices of nonzero bits of $\mathbf{s}$. Then,

– the adversary is given $(c_1, \ldots, c_l, Hom(c_{i_1}, \ldots, c_{i_w}))$ as the public key, where

$$(c_1, \ldots, c_l) = E_{sk}(\mathbf{0}^l; \mathbf{r})$$

and $sk \leftarrow G(1^\lambda)$.
– In response to *enc-secret*(i) we return $(c'_1, \ldots, c'_l, Hom(c'_{i_1}, \ldots, c'_{i_w}, E_{sk'}(\mathbf{s}_i; 0)))$, where

$$(c'_1, \ldots, c'_l) = E_{sk'}(\mathbf{0}^l; \mathbf{r})$$

and $sk' \leftarrow G(1^\lambda)$. Again we emphasize sk' is chosen freshly for each query.
– In response to *enc-secret*$(\bar{i})$ we return $(c''_1, \ldots, c''_l, Hom(c''_{i_1}, \ldots, c''_{i_w}, E_{sk''}(\bar{\mathbf{s}}_i; 0)))$, where

$$(c''_1, \ldots, c''_l) = E_{sk''}(\mathbf{0}^l; \mathbf{r})$$

and $sk'' \leftarrow G(1^\lambda)$.

Game-1: In this game we handle key generation exactly as in *Game*-0, but we reply to *enc-secret* queries in a special way. Formally, generate $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}^l$ and $\mathbf{s} \leftarrow \{0,1\}^l$ and let $(i_1, \ldots, i_w)$ be the indices of nonzero bits of $\mathbf{s}$. Then,

– the adversary is given $(c_1, \ldots, c_l, Hom(c_{i_1}, \ldots, c_{i_w}))$ as the public key, where $(c_1, \ldots, c_l) = E_{sk}(\mathbf{0}^l; \mathbf{r})$, for $sk \leftarrow G(1^\lambda)$.
– In response to *enc-secret*(i) we return $(c'_1, \ldots, c'_l, Hom(c'_{i_1}, \ldots, c'_{i_w}, E_{sk'}(\mathbf{s}_i; 0)))$, where $(c'_1, \ldots, c'_l) = E_{sk'}(\mathbf{e}_i; \mathbf{r})$ and $sk' \leftarrow G(1^\lambda)$.
– In response to *enc-secret*$(\bar{i})$ we return $(c''_1, \ldots, c''_l, Hom(c''_{i_1}, \ldots, c''_{i_w}, E_{sk''}(\bar{\mathbf{s}}_i; 0)))$, where $(c''_1, \ldots, c''_l) = E_{sk''}(\mathbf{e}_i; \mathbf{r})$ and $sk'' \leftarrow G(1^\lambda)$.

We claim that the difference between *Game*-0 and *Game*-1 can be simulated through the l-RS game. The reason is if we know $\mathbf{s}$, then we can compute $Hom(c'_{i_1}, \ldots, c'_{i_w}, E_{sk'}(\mathbf{s}_i; 0))$ from $(c'_1, \ldots, c'_l)$ even if we do not have sk': note that here we are using the degenerate condition of the homomorphism property. A similar argument holds with respect to c and c''. Moreover, for every $1 \le j \le l$, the ciphertexts c_j, c'_j and c''_j were formed under the same randomness. Thus, we can reduce any distinguisher between *Game*-0 and *Game*-1 to an l-RS game adversary $\mathcal{A}$ as follows: $\mathcal{A}$ samples $\mathbf{s} \leftarrow \{0,1\}^l$ and lets $(i_1, \ldots, i_w)$ be the indices of nonzero bits of $\mathbf{s}$; it calls its RS-oracle on $\mathbf{0}^l$ to receive $(c_1, \ldots, c_l)$ and then returns $(c_1, \ldots, c_l, Hom(c_{i_1}, \ldots, c_{i_w}))$ as the public key; it responds to *enc-secret*(i) by first calling its oracle on $\mathbf{e}_i$ to get $(c'_1, \ldots, c'_l)$ and then returning $(c'_1, \ldots, c'_l, Hom(c'_{i_1}, \ldots, c'_{i_w}, E_{sk'}(\mathbf{s}_i; 0)))$; it responds to *enc-secret*$(\bar{i})$ in a similar way. Thus, by Lemma 3 we obtain that *Game*-0 $\equiv^G$ *Game*-1.

Finally, note that under this game, the distribution of the public key and the distributions of responses to *enc-secret*(i)'s and to *enc-secret*$(\bar{i})$'s are:

$$\begin{array}{ll} \big(E_{sk}(\mathbf{0}^l; \mathbf{r}), E_{sk}(0; r_{l+1})\big) & \text{public key} \\ (E_{sk}(\mathbf{e}_i; \mathbf{r}), E_{sk'}(0; r_{l+1})) & \textit{enc-secret}(i) \\ (E_{sk}(\mathbf{e}_i; \mathbf{r}), E_{sk''}(1; r_{l+1})) & \textit{enc-secret}(\bar{i}), \end{array} \tag{1}$$

where $sk, sk', sk'' \leftarrow G(1^\lambda)$, $\mathbf{s} \leftarrow \{0,1\}^l$ and $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}^l$ and $r_{l+1} = \mathbf{s} \cdot \mathbf{r}$. In particular, note that the bits of $\mathbf{s}$ never appear as a plaintext (under E) in Equation 1, and the only place we use $\mathbf{s}$ is to form r_{l+1}.

Game-2: This game proceeds exactly as in *Game*-1, except we now sample r_{l+1} independently of all other r_i's. Namely, we sample $(r_1, \ldots, r_l, r_{l+1}) \leftarrow \mathcal{R}^{l+1}$ and run the game by forming the public key and responses to the adversary's queries exactly as in Equation 1. Notice that the entire game can be simulated by only knowing $(r_1, \ldots, r_l, r_{l+1})$: we generate the public key and we answer to *enc-secret* queries by sampling sk, sk'

and sk'' on our own and forming the outputs as spelled out by Equation 1. (Here we are exploiting the fact that the bits of $\mathbf{s}$ never appear as a plaintext under E in Equation 1.) Thus, since $l = \omega(\log\lambda) + \log(|\mathcal{R}|)$ and the inner product is a family of universal hash functions, by Lemma 2 (indeed by the Leftover Hash Lemma, which is a special case of Lemma 2) we obtain that the statistical distance between $(\mathbf{r}, \mathbf{s}\cdot\mathbf{r})$ and a tuple chosen uniformly at random from $\mathcal{R}_\lambda^l$ is at most $\sqrt{1/2^{\omega(\log\lambda)}} = negl(\lambda)$, and thus *Game*-1 $\equiv^G$ *Game*-2.

Game-3: In this game we again sample r_{l+1} independently of other r_i's, but reply to all queries as "encryptions" of zero. That is, we generate $(r_1, \ldots, r_l, r_{l+1}) \leftarrow \mathcal{R}^{l+1}$ and form the public key and responses to the adversary's queries as follows:

$$\begin{array}{ll} (E_{sk}(0;r_1), \ldots, E_{sk}(0;r_l), E_{sk}(0;r_{l+1})) & \text{public key} \\ (E_{sk'}(0;r_1), \ldots, E_{sk'}(0;r_l), E_{sk'}(0;r_{l+1})) & \text{response to all queries} \end{array} \quad (2)$$

where, again, sk' is sampled freshly for each query. Now using the fact that all r_i's are sampled independently, and also that sk' is generated using fresh coins each time, we obtain that any adversary that can distinguish between *Game*-2 and *Game*-3 can be reduced to break the $(l+1)$-RS security of $\mathcal{E}$ (which is a contradiction by Lemma (3)). Thus, *Game*-2 $\equiv^G$ *Game*-3.

Game-4: In this game we change back the distributions of r_i's to the original, but answer to all the adversary's queries as encryptions of zero. That is, we generate $\mathbf{s} \leftarrow \{0,1\}^l$, $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}^l$, let $r_{l+1} = \mathbf{s}\cdot\mathbf{r}$, and form the public key and responses to the adversary's queries as follows:

$$\begin{array}{ll} (E_{sk}(0;r_1), \ldots, E_{sk}(0;r_l), E_{sk}(0;r_{l+1})) & \text{public key} \\ (E_{sk'}(0;r_1), \ldots, E_{sk'}(0;r_l), E_{sk'}(0;r_{l+1})) & \text{responses to all queries} \end{array} \quad (3)$$

Now, similarly to our proof of *Game*-1 $\equiv^G$ *Game*-2, since *Game*-3 and *Game*-4 differ only in the way that $(r_1, \ldots, r_l, r_{l+1})$ is generated, and again using the fact that $l = \omega(\log\lambda) + \log(|\mathcal{R}|)$, by applying Lemma 2, we conclude that *Game*-3 $\equiv^G$ *Game*-4. This completes the proof. □

We now give the statement and proof for n-projection security.

Theorem 2. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure private-key bit-encryption scheme providing degenerate homomorphism and reproducibility. For any constant $c > 1$, by taking $l = n\log(|\mathcal{R}_\lambda|) + \omega(\log\lambda)$, the scheme built in Construction 1 is n-projection secure.*

Proof. To represent the n-projection game more concisely, we use the following notation for denoting the adversary's queries.

- $Enc(pub_h, secret_i, j)$ encrypt the jth bit of the ith secret key under the hth public key; and
- $Enc(pub_h, secret_i, \bar{j})$ encrypt the negation of the jth bit of the ith secret key under the hth public key.

We also need the following notation. In all the games below we denote the ith secret key as $\mathbf{s}_i$ and define $\mathbf{s}_{ij}$ to be the jth-bit of the ith secret key. If $\mathbf{v}$ is a vector of size l, and $\mathbf{s} \in \{0,1\}^l$ we define $\mathbf{v}[\mathbf{s}]$ to be $(\mathbf{v}_{i_1}, \ldots, \mathbf{v}_{i_w})$, where $i_1 < \cdots < i_w$ are the indices of nonzero bits of $\mathbf{s}$. We also recall the notation $\mathbf{e}_i^l$, $\mathbf{0}^l$, and $E_{sk}(\mathbf{b};\mathbf{r})$, defined at the beginning of Section 4. To ease notation though, we shall write $\mathbf{e}_i$ to mean $\mathbf{e}_i^l$.

We introduce a series of hybrid games and show no adversary can distinguish between any two adjacent games. The first game corresponds to the real-encryption n-circular-security game, while the last game is the one where we always encrypt 0. Letting x_i be the adversary's output in *Game-i*, we write *Game-i* $\equiv^c$ *Game-j* to indicate $|\Pr[x_i = 1] - \Pr[x_j = 1]| = negl$. In all these games, whenever we write $sk \leftarrow G(1^\lambda)$ we mean that sk is chosen freshly, so we keep using the same variable sk inside each game whenever we are producing a new key.

Game-0: real encryption. This game provides the adversary with a view that is identical to that under the projection security game in which the challenge bit is zero. The identical view is produced by using the algorithm Hom to produce the public keys and to reply to encryption queries. (See the first paragraph following Construction 1.) Generate $\mathbf{s}_1, \ldots, \mathbf{s}_n \leftarrow \{0,1\}^l$ and for $1 \le i \le n$, $\mathbf{r}_i = (r_{i1}, \ldots, r_{il}) \leftarrow \mathcal{R}_\lambda^l$. (Here $\mathbf{s}_i$ will be the secret key of the ith "user" and $\mathbf{r}_i$ will be used to produce the public key for the ith user.) For each h, where $1 \le h \le n$, do:

- the adversary is given public key $(\mathbf{c}, Hom(\mathbf{c}[\mathbf{s}_h]))$, where $\mathbf{c} = E_{sk}(\mathbf{0}^l, \mathbf{r}_h)$, for $sk \leftarrow G(1^\lambda)$. Recall that $E_{sk}(\mathbf{0}^l, \mathbf{r}_h) = (E_{sk}(0; r_{h1}), \ldots, E_{sk}(0; r_{hl}))$.
- In response to $Enc(pub_h, secret_i, j)$, we return $(\mathbf{c}', Hom(\mathbf{c}'[\mathbf{s}_h], E_{sk'}(\mathbf{s}_{ij}; 0)))$, where $sk' \leftarrow G(1^\lambda)$, $\mathbf{c}' = E_{sk'}(\mathbf{0}^l, \mathbf{r}_h) = (E_{sk'}(0; r_{h1}), \ldots, E_{sk'}(0; r_{hl}))$. Recall by Proposition 1 that this provides an identical encryption view. Again we stress that sk' is chosen freshly for each query.
- In response to $Enc(pub_h, secret_i, \bar{j})$, we return $(\mathbf{c}'', Hom(\mathbf{c}''[\mathbf{s}_h], E_{sk''}(\bar{\mathbf{s}}_{ij}; 0)))$, where $sk'' \leftarrow G(1^\lambda)$, $\mathbf{c}'' = E_{sk''}(\mathbf{0}^l, \mathbf{r}_h) = (E_{sk''}(0; r_{h1}), \ldots, E_{sk'}(0; r_{hl}))$.

Game-1: In this game we handle key generation exactly as in *Game*-0, but we reply to Enc queries in the following way. For each h, where $1 \le h \le n$,

- in response to $Enc(pub_h, secret_i, j)$ we return $(\mathbf{c}', Hom(\mathbf{c}'[\mathbf{s}_h], E_{sk'}(\mathbf{s}_{ij}; 0)))$, where $sk' \leftarrow G(1^\lambda)$ and $\mathbf{c}' = E_{sk'}(\mathbf{e}_j; \mathbf{r}_h)$.
- in response to $Enc(pub_h, secret_i, \bar{j})$ we return $(\mathbf{c}'', Hom(\mathbf{c}''[\mathbf{s}_h], E_{sk''}(\bar{\mathbf{s}}_{ij}; 0)))$, where $sk'' \leftarrow G(1^\lambda)$ and $\mathbf{c}'' = E_{sk''}(\mathbf{e}_j; \mathbf{r}_h)$.

We now wish to use Lemma 3 to prove *Game*-0 $\equiv^c$ *Game*-1. First, note that in both games the hth public key is sampled as $(\mathbf{c}, Hom(\mathbf{c}[\mathbf{s}_h]))$, for $\mathbf{c} = E_{sk}(\mathbf{0}^l; \mathbf{r}_h)$. Moreover, in both games, the response to $Enc(pub_h, secret_i, j)$ is formed as $(\mathbf{c}', Hom(\mathbf{c}'[\mathbf{s}_h], E_{sk'}(\mathbf{s}_{ij}; 0)))$ with the only difference that in *Game*-0, $\mathbf{c}'$ are bitwise encryptions of $\mathbf{0}^l$ under the randomness values $(r_{h1}, \ldots, r_{hl})$ and under a fresh sk', while in *Game*-1, $\mathbf{c}'$ are bitwise encryptions of $\mathbf{e}_j$ under the randomness values $(r_{h1}, \ldots, r_{hl})$ and under a fresh sk'. Similarly, in both games, the response to $Enc(pub_h, secret_i, \bar{j})$ is formed as $(\mathbf{c}'', Hom(\mathbf{c}''[\mathbf{s}_h], E_{sk''}(\bar{\mathbf{s}}_{ij}; 0)))$ with the only difference that in *Game*-0, $\mathbf{c}''$ are bitwise encryptions of $\mathbf{0}^l$ under the randomness values $(r_{h1}, \ldots, r_{hl})$ and under a fresh sk'', while in *Game*-1, $\mathbf{c}''$ are bitwise encryptions of $\mathbf{e}_j$ under the randomness values $(r_{h1}, \ldots, r_{hl})$ and under a fresh sk''. Finally, note that if we have all of $\mathbf{s}_1, \ldots, \mathbf{s}_n$ and if we get $\mathbf{c}'$ from an oracle, knowing that $\mathbf{c}'$ corresponds either to $E_{sk'}(\mathbf{0}^l, \mathbf{r}_h)$ or to $E_{sk'}(\mathbf{e}_j, \mathbf{r}_h)$, then we can compute $Hom(\mathbf{c}'[\mathbf{s}_h], E_{sk'}(\mathbf{s}_{ij}; 0))$ even without knowing the underlying sk'; this follows by the degenerate PR-additive homomorphism of the private-key scheme. A similar argument holds for $\mathbf{c}''$.

From the discussion above we can see how to reduce a distinguisher between *Game*-0 and *Game*-1 to an adversary $\mathcal{A}$ against the (l, n)-RS security of the private-key scheme $\mathcal{E}$: $\mathcal{A}$ samples $\mathbf{s}_h$, for each $1 \le h \le n$ by itself; it generates the hth public key by querying its RS-oracle on $(h, \mathbf{0}^l)$ to get $\mathbf{c}$ and then outputting $(\mathbf{c}, Hom(\mathbf{c}[\mathbf{s}_h]))$; it responds to an $Enc(pub_h, secret_i, j)$ query by calling its RS-oracle on $(h, \mathbf{e}_j)$ to get $\mathbf{c}'$ and forming the output as explained above; and it responds to an $Enc(pub_h, secret_i, \bar{j})$ in a similar manner.

Finally, notice that under *Game*-1, for each $1 \le h \le n$, the distributions of the public key, the response to $Enc(pub_h, secret_i, j)$ and the response to $Enc(pub_h, secret_i, \bar{j})$ are indeed identical to:

$$\begin{aligned}
&(E_{sk}(\mathbf{0}^l; \mathbf{r}_h), E_{sk}(0; r_{h,l+1})) \qquad \text{public key} \\
&(E_{sk'}(\mathbf{e}_j^l; \mathbf{r}_h), E_{sk'}(\mathbf{s}_{ij} + \mathbf{s}_{hj}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, j) \\
&(E_{sk''}(\mathbf{e}_j^l; \mathbf{r}_h), E_{sk''}(\bar{\mathbf{s}}_{ij} + \mathbf{s}_{hj}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, \bar{j}),
\end{aligned} \tag{4}$$

where $r_{h,l+1} = \mathbf{s}_h \cdot \mathbf{r}_h$.

Game-2: In this game we generate the distributions as in Equation 4, but with a small deviation. For $1 \le h \le n$ we generate $\mathbf{s}_h \leftarrow \{0,1\}^l$ and $\mathbf{r}_h \leftarrow \mathcal{R}_\lambda^l$, and set $r_{h,l+1} = \mathbf{s}_h \cdot \mathbf{r}_h$. Moreover, for $1 \le i \le n$ we set

$\mathbf{d}_i = \mathbf{s}_1 + \mathbf{s}_i$, i.e., bitwise binary addition. We form the hth public key and responses to $Enc(pub_h, secret_i, j)$ and to $Enc(pub_h, secret_i, \bar{j})$ queries as follows:

$$\begin{aligned}
&(E_{sk}(\mathbf{0}^l; \mathbf{r}_h), E_{sk}(0; r_{h,l+1})) \qquad \text{public key} \\
&(E_{sk'}(\mathbf{e}^l_j; \mathbf{r}_h), E_{sk'}(\mathbf{d}_{ij} + \mathbf{d}_{hj}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, j) \\
&(E_{sk''}(\mathbf{e}^l_j; \mathbf{r}_h), E_{sk''}(\overline{\mathbf{d}_{ij} + \mathbf{d}_{hj}}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, \bar{j}),
\end{aligned} \tag{5}$$

Now since $\mathbf{s}_{ij}+\mathbf{s}_{hj} = \mathbf{s}_{1j}+\mathbf{s}_{ij}+\mathbf{s}_{1j}+\mathbf{s}_{hj} = \mathbf{d}_{ij}+\mathbf{d}_{hj}$ and $\bar{\mathbf{s}}_{ij}+\mathbf{s}_{hj} = \mathbf{s}_{1j}+\bar{\mathbf{s}}_{ij}+s_{1j}+s_{hj} = \bar{\mathbf{d}}_{ij}+\mathbf{d}_{hj} = \overline{\mathbf{d}_{ij} + \mathbf{d}_{hj}}$, and since the adversarial view under *Game*-1 is as in Equation 4, we get that *Game*-1 and *Game*-2 provide identically-distributed adversarial views. Finally, note that by having only the tuple

$$D = (\mathbf{r}_1, \ldots, \mathbf{r}_n, r_{1,l+1}, \ldots, r_{n,l+1}, \mathbf{d}_2, \ldots, \mathbf{d}_n), \tag{6}$$

(and in particular without having the individual $\mathbf{s}_1, \ldots, \mathbf{s}_n$) we can perfectly simulate all the distributions given in Equation 5, by sampling sk, sk' and sk'' by ourselves. Thus, the whole view of an adversary in this game can be simulated by only having D, through a simulation algorithm $Sim()$.

Game-3: This game proceeds exactly as in *Game*-2, to form the view as $Sim(D)$, except that we generate D (given in Equation 6) differently from *Game*-2 by sampling $r_{h,l+1}$, for all $1 \leq h \leq n$, independently of $\mathbf{r}_h$; the rest of D remains unchanged. Now we show that the following two distributions are statistically indistinguishable, which by the last statement of *Game*-2, it implies that *Game*-2 $\equiv^s$ *Game*-3;

$$\begin{aligned}
D &= (\mathbf{r}_1, \ldots, \mathbf{r}_n, \mathbf{s}_1 \cdot \mathbf{r}_1, \ldots, \mathbf{s}_n \cdot \mathbf{r}_n, \mathbf{d}_2, \ldots, \mathbf{d}_n), \\
D' &= (\mathbf{r}_1, \ldots, \mathbf{r}_n, r_1, \ldots, r_n, \mathbf{d}_2, \ldots, \mathbf{d}_n),
\end{aligned} \tag{7}$$

where $\mathbf{r}_1, \ldots, \mathbf{r}_n \leftarrow \mathcal{R}^l_\lambda$, $\mathbf{s}_1, \ldots, \mathbf{s}_n \leftarrow \{0,1\}^l$, and, for $2 \leq i \leq n$, $\mathbf{d}_i = \mathbf{s}_1 + \mathbf{s}_i$. To show Equation 7, observe that

$$\begin{aligned}
\tilde{H}_\infty(\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n \mid \mathbf{d}_2, \ldots, \mathbf{d}_n) &\geq H_\infty(\mathbf{s}_1, \mathbf{s}_2 \ldots \mathbf{s}_n) - (n-1)l \\
&= l = n \log |\mathcal{R}_\lambda| + \omega(\log \lambda).
\end{aligned}$$

Now considering the fact that

$$Ext((\mathbf{s}_1, \ldots, \mathbf{s}_n), (\mathbf{r}_1, \ldots, \mathbf{r}_n)) =^{\text{def}} (\mathbf{s}_1 \cdot \mathbf{r}_1, \ldots, \mathbf{s}_n \cdot \mathbf{r}_n)$$

is a family of universal hash functions, by applying Lemma (2) we obtain that the statistical distance between D and D' is at most $\sqrt{1/2^{\omega(\log \lambda)}}$, which is negligible.

Note that under this game for $1 \leq h \leq n$, the distributions of the public key, the response to $Enc(pub_h, secret_i, j)$ and the response to $Enc(pub_h, secret_i, \bar{j})$ are indeed identical to:

$$\begin{aligned}
&(E_{sk}(\mathbf{0}^l; \mathbf{r}_h), E_{sk}(0; r_{h,l+1})) \qquad \text{public key} \\
&(E_{sk'}(\mathbf{e}^l_j; \mathbf{r}_h), E_{sk'}(\mathbf{d}_{ij} + \mathbf{d}_{hj}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, j) \\
&(E_{sk''}(\mathbf{e}^l_j; \mathbf{r}_h), E_{sk''}(\overline{\mathbf{d}_{ij} + \mathbf{d}_{hj}}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, \bar{j}),
\end{aligned} \tag{8}$$

where $\mathbf{r}_1, \ldots, \mathbf{r}_h \leftarrow \mathcal{R}^l_\lambda$, $r_{1,l+1}, \ldots, r_{n,l+1} \leftarrow \mathcal{R}_\lambda$, $\mathbf{s}_1, \ldots, \mathbf{s}_n \leftarrow \{0,1\}^l$ and $\mathbf{d}_i = \mathbf{s}_i + \mathbf{s}_1$, for $1 \leq i \leq l$. Note that $\mathbf{s}_i$'s, except in forming $\mathbf{d}_i$'s, play no other role in the distributions above.

Game-4: This game proceeds as in *Game*-3 (i.e., Equation 8), but we reply to all queries as "encryptions" of zero. That is, generate $\mathbf{r}_1, \ldots, \mathbf{r}_n \leftarrow \mathcal{R}^n_\lambda$, and for $1 \leq h \leq n$, form the hth public key as

$(E_{sk}(\mathbf{0}^l;\mathbf{r}_h), E_{sk}(0;r_{h,l+1}))$, where $r_{h,l+1}$ is chosen independently of $\mathbf{r}_h$, and respond to every query under this public key as $(E_{sk'}(\mathbf{0}^l;\mathbf{r}_h), E_{sk'}(0;r_{h,l+1}))$. (Note that sk''s is chosen freshly for each query.)

Since $r_{h,l+1}$ for every h is sampled independently of $\mathbf{r}_h$, again we can easily see that any adversary that can distinguish between *Game*-3 and *Game*-4 can be reduced to break the $(l+1,n)$-RS security of $\mathcal{E}$ (which is a contradiction by Lemma (3)), implying *Game*-3 $\equiv^c$ *Game*-4. Finally, note that the entire view of an adversary under this game can be simulated by having only $(\mathbf{r}_1, r_{1,l+1}, \ldots, \mathbf{r}_n, r_{n,l+1})$.

Game-5: This game runs exactly like *Game*-4, except that for $1 \le h \le n$ we now define $r_{h,l+1} = \mathbf{s}_h \cdot \mathbf{r}_h$. For every h by the Leftover Hash Lemma (in fact a special case of Lemma 2) we have that $(\mathbf{r}_h, \mathbf{s}_h \cdot \mathbf{r}_h)$ is statistically indistinguishable from a tuple chosen uniformly at random from $\mathcal{R}_\lambda^{l+1}$. Since n is poly-bounded, we obtain that *Game*-4 $\equiv^s$ *Game*-5. The view of the adversary under this game is exactly that under the n-projection security game when the challenge bit is one. This concludes the proof. □

Remark 1. For simplicity, we presented Construction 1, which always produces a bit-encryption scheme, only for the case that the base scheme is also a bit-encryption one. It is easy to modify Construction 1, so it works (i.e., produces a bit-encryption scheme) even if the original scheme is not bit-encryption. To do this, we just need to fix a nonzero $m \in \mathcal{M}_\lambda$ and change the encryption algorithm of Construction 1 by replacing b with $b \cdot m$. As for security, we need to change the RS game, so that the queried vectors can now be of the form $(0,\ldots,0,-m,0,\ldots,0)$, $(0,\ldots,0,m,0,\ldots,0)$ or $(0,\ldots,m,0,\ldots,0,m)$. Now the proof of Lemma 3 follows with straightforward changes. As for the the proof of Theorem 1 (for simplicity we discuss changes with respect to the proof of Theorem 1, rather than the more complex proof of Theorem 2), we change Game-1 (i.e., the simulated encryption game), so that (i) in response to *enc-secret*(u) we return $(c'_1,\ldots,c'_l,c'_{l+1})$, where $c'_{l+1} = Hom(c'_{h_1},\ldots,c'_{h_w}, E_{sk}(\mathbf{s}_u \cdot m;0))$, in which $c'_i = E_{sk}(0;r_i)$, for any $1 \le i \ne u \le l$ and $c'_u = E_{sk}(-m;r_u)$; also, in response to *enc-secret*$(\bar{u})$, we return $(c'_1,\ldots,c'_l,c'_{l+1})$, where $c'_{l+1} = Hom(c'_{h_1},\ldots,c'_{h_w}, E_{sk}(\bar{\mathbf{s}_u} \cdot m;0))$, where $c'_i = E_{sk}(0;r_i)$, for any $1 \le i \ne u \le l$ and $c'_u = E_{sk}(m;r_u)$. Now Game-0 $\equiv^c$ Game-1 follows by the new version of Lemma 3. Moreover, in Game-1, the view of an adversary against *enc-secret*(u) and *enc-secret*$(\bar{u})$ would be, respectively,

$$(E_{sk'}(0;r_1),\ldots,E_{sk'}(0;r_{i-1}),E_{sk'}(-m;r_i),E_{sk'}(0;r_{i+1}),\ldots,E_{sk'}(0;r_l),E_{sk'}(0;r_{l+1}))\,,$$

$$(E_{sk'}(0;r_1),\ldots,E_{sk'}(0;r_{i-1}),E_{sk'}(m;r_i),E_{sk'}(0;r_{i+1}),\ldots,E_{sk'}(0;r_l),E_{sk'}(m;r_{l+1}))\,,$$

and since all the plaintexts encrypted under sk''s are independent of $\mathbf{s}$, the rest of the proof follows as that of Theorem 1.

5 Proof of leakage resilience

The following theorem shows the leakage resilience property of our scheme.

Theorem 3. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure private-key bit-encryption scheme providing degenerate homomorphism and reproducibility. Then, the scheme built in Construction 1 is $(l - \log|\mathcal{R}_\lambda| - u)$-length leakage resilient, for any $u \in \omega(\log\lambda)$. Moreover, by taking $l = \omega(\log|\mathcal{R}_\lambda| + u)$, the constructed scheme achieves a $(1 - o(1))$ resilience rate.*

Proof. We first show the second statement of the theorem, assuming the first statement is true. Fix $u \in \omega(\log\lambda)$. We know that the scheme provides $(l - \log|\mathcal{R}_\lambda| - u)$-length leakage resilience, and so its resilience rate is

$$\frac{\omega(\log|\mathcal{R}_\lambda| + u) - \log|\mathcal{R}_\lambda| - u}{\omega(\log|\mathcal{R}_\lambda| + u)} = 1 - \frac{\log|\mathcal{R}_\lambda| + u}{\omega(\log|\mathcal{R}_\lambda| + u)} = 1 - o(1). \tag{9}$$

To prove the first statement, first we assume without loss of generality that the adversary always outputs $(0,1)$ as its challenge query, since otherwise the challenge ciphertext can be simulated by the adversary itself. We prove the first statement through a series of games, where the first game matches the actual leakage game (under a fixed challenge bit b), and in the last game the view of the adversary is independent of the challenge bit b. We conclude the proof by showing that the views of the adversary under any two adjacent

games under the same $b \in \{0,1\}$ are computationally indistinguishable. Thus, fix $b \in \{0,1\}$ for the rest of the proof. In all game below, we let f be the leakage query of the adversary.

Game-0: In this game we reply to the adversary's queries exactly as in the actual leakage game, where the challenge bit is b. Thus, at the end of the game, the view of the adversary is $(c_1, \ldots, c_l, c_{l+1}, f(\mathbf{s}), c'_1, \ldots, c'_l, c'_{l+1})$, produced as follows: $\mathbf{s} \leftarrow \{0,1\}^l$, $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}_\lambda^l$, $r_{l+1} = \mathbf{s} \cdot \mathbf{r}$, $sk \leftarrow G(1^\lambda)$, $sk' \leftarrow G(1^\lambda)$, $c_i = E_{sk}(0; r_i)$, for $1 \le i \le l+1$, $c'_j = E_{sk'}(0; r_j)$, for $1 \le j \le l$, and $c'_{l+1} = E_{sk'}(b; r_{l+1})$.

Notice that the view of the adversary may identically be produced as

$$(c_1, \ldots, c_l, c''_{l+1}, f(\mathbf{s}), c'_1, \ldots, c'_l, c'''_{l+1}), \tag{10}$$

where all c_i's and c'_i's are produced as above, and $c''_{l+1} = Hom(c_{h_1}, \ldots, c_{h_w})$ and $c'''_{l+1} = Hom(c'_{h_1}, \ldots, c'_{h_w}, E_{sk'}(b; 0))$ with $(h_1, \ldots, h_w)$ being the indices of non-zero bits of $\mathbf{s}$.

Game-1. In this game we generate the secret key, the public key and the response to the leakage query exactly as in *Game*-0, but we reply to the encryption challenge query in a special way. Formally, choose $\mathbf{s} \leftarrow \{0,1\}^l$, $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}_\lambda^l$, let $(h_1, \ldots, h_w)$ be the indices of non-zero bits of $\mathbf{s}$, and

- form the public key as $(c_1, \ldots, c_l, c''_{l+1})$, where $sk \leftarrow G(1^\lambda)$, $c_i = E_{sk}(0; r_i)$, for $1 \le i \le l$, and $c''_{l+1} = Hom(c_{h_1}, \ldots, c_{h_w})$;
- reply to the the leakage query f with $f(\mathbf{s})$;
- return $(c'_1, \ldots, c'_l, c'''_{l+1})$ as the challenge ciphertext, where $\mathbf{b} = (b_1, \ldots, b_l) \leftarrow \{0,1\}^l$, $sk' \leftarrow G(1^\lambda)$, $c'_j = E_{sk'}(b_j; r_j)$, for $1 \le j \le l$, and

$$c'''_{l+1} = Hom(c'_{h_1}, \ldots, c'_{h_w}, E_{sk'}(b; 0)).$$

To show *Game*-0 $\equiv^G$ *Game*-1, note that both games can be simulated in exactly the same way by only having $\mathcal{D} = (c_1, \ldots, c_l, c'_1, \ldots, c'_l)$ (see Equation 10); this can be done by sampling $\mathbf{s}$ by ourselves and forming c''_{l+1} and c'''_{l+1} from, respectively, $(c_1, \ldots, c_l)$ and $(c'_1, \ldots, c'_l)$ by using the degenerate homomorphic property of $\mathcal{E}$. Further, since $\mathcal{E}$ is reproducible, in both games the distribution of $(c_1, \ldots, c_l)$ can be generated from $(c'_1, \ldots, c'_l)$ alone. Now since the distributions produced for $(c'_1, \ldots, c'_l)$ under the two games are computationally indistinguishable, which is followed by semantic security (recall that in *Game*-0, c'_i's are encryptions of zeros and in *Game*-1, they are encryptions of the bits of $\mathbf{b}$), we get that the distributions produced for $\mathcal{D}$ under the two games are computationally indistinguishable. Thus, we conclude *Game*-0 $\equiv^G$ *Game*-1. Notice that, under Game-1, the view of the adversary is

$$\big(E_{sk}(0; r_1), \ldots, E_{sk}(0; r_l), E_{sk}(0; \mathbf{s} \cdot \mathbf{r}), f(\mathbf{s}), E_{sk'}(b_1; r_1), \ldots, E_{sk'}(b_l; r_l), E_{sk'}(b_{l+1} + b; \mathbf{s} \cdot \mathbf{r})\big), \tag{11}$$

where $b_{l+1} = \mathbf{s} \cdot \mathbf{b}$.

Game-2. This game runs exactly as in *Game*-1 (Equation 11), except that now we generate $b_{l+1} \leftarrow \{0,1\}$, i.e., independent of $\mathbf{b} = (b_1, \ldots, b_l)$. First, notice that both *Game*-1 and *Game*-2 can be simulated in exactly the same manner by only having

$$Dis = (\mathbf{r}, \mathbf{s} \cdot \mathbf{r}, \mathbf{b}, b_{l+1}, f(\mathbf{s})). \tag{12}$$

The only difference between Dis from *Game*-1 to *Game*-2 is that under *Game*-1 we set $b_{l+1} = \mathbf{s} \cdot \mathbf{b}$, while in *Game*-2 we sample b_{l+1} freshly; the other parts of Dis are generated in the same way under both games: that is, $\mathbf{s} \leftarrow \{0,1\}^l$ and $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}_\lambda^l$. Thus, to show the indistinguishability between these two games, it suffices to show that the distributions of Dis under the two games are indistinguishable. We have,

$$\begin{aligned}
\tilde{H}_\infty(\mathbf{s} | \mathbf{r}, \mathbf{s} \cdot \mathbf{r}, f(\mathbf{s})) &\ge \tilde{H}_\infty(\mathbf{s} | \mathbf{r}, f(\mathbf{s})) - log|\mathcal{R}_\lambda| \\
&= \tilde{H}_\infty(\mathbf{s} | f(\mathbf{s})) - log|\mathcal{R}_\lambda| \\
&\ge H_\infty(\mathbf{s}) - l + \log|\mathcal{R}_\lambda| + u - log|\mathcal{R}_\lambda| \\
&= u = \omega(\log \lambda).
\end{aligned}$$

Now, since $\mathbf{r}$ is independent of $\mathbf{b}$, and also that f is independent of $\mathbf{b}$ (since f is queried before seeing the challenge ciphertext) we may use Lemma 2 to deduce that the distribution of Dis under *Game*-1 and *Game*-2 are statistically indistinguishable. To apply Lemma 2, take $D = \mathbf{s}$, $S = \mathbf{b}$ and $X = (\mathbf{r}, \mathbf{s} \cdot \mathbf{r}, f(\mathbf{s}))$. Notice that *Game*-2 produces the same views for the adversary under $b = 0$ and $b = 1$ (since b_{l+1} is chosen uniformly at random and hides the value of b), and hence the proof is complete. □

6 Realizations

We show how to realize our base encryption primitive under various number-theoretic assumptions. In Subsection 6.1 we formulate an abstraction, called *homomorphic weak pseudorandom functions*, and use them to realize our encryption primitive. Then in Subsection 6.2 we give realizations of such pseudorandom functions using *homomorphic hash-proof systems*. Finally, in Subsection 6.3 we show how to realize our encryption primitive under *subgroup indistinguishably*.

6.1 Realizations from homomorphic weak PRFs

We introduce the notion of *homomorphic weak pseudorandom functions (PRFs)*, which is a homomorphic version of the notion of weak PRFs, introduced by Naor and Reingold [34].

Let $K = \{K_\lambda\}_{\lambda \in \mathbb{N}}$, $D = \{D_\lambda\}_{\lambda \in \mathbb{N}}$ and $R = \{R_\lambda\}_{\lambda \in \mathbb{N}}$ be ensembles of sets. For each security parameter λ and each $k \in K_\lambda$ we have an associated function $f_k : D_\lambda \to R_\lambda$. We let $F_\lambda = \{f_k \mid k \in K_\lambda\}$ and $F = \{F_\lambda\}_{\lambda \in \mathbb{N}}$. The following is the definition of weak pseudorandomness for a function family.

Definition 3. *[34] We call F a* weak pseudorandom function family *if for any polynomial function $p = p(\lambda)$, it holds that $\mathcal{DS}_1 \equiv^c \mathcal{DS}_2$, where*

$$\mathcal{DS}_1 \equiv (d_1, r_1), \ldots, (d_p, r_p)$$
$$\mathcal{DS}_2 \equiv (d_1, f_k(d_1)), \ldots, (d_p, f_k(d_p)),$$

for $k \leftarrow K_\lambda$, $d_1, \ldots, d_p \leftarrow D_\lambda$ and $r_1, \ldots, r_p \leftarrow R_\lambda$.[4]

Note that a PRF in the standard sense is trivially a weak PRF.

Let F be as above. We call F *homomorphic* if for every $\lambda \in \mathbb{N}$, both D_λ and R_λ are groups, and that for every $k \in K_\lambda$, the function f_k is a homomorphism from D_λ to R_λ.

Now we show that the standard method of constructing CPA-secure private-key encryption from a PRF, when applied to a homomorphic weak PRF, results in the kind of encryption primitive we need.

Lemma 4. *Assuming the existence of a homomorphic weak pseudorandom function family, there exists a CPA-secure private-key encryption scheme which is degenerately homomorphic and reproducible.*

Proof. Let F be a homomorphic weak PRF with the associated set parameters given above (i.e., K_λ, etc.). Construct $\mathcal{E} = (G, E, Dec)$, with plaintext space R_λ and randomness spaces D_λ as follows: $G(1^\lambda)$ returns $k \leftarrow K_\lambda$; $E_k(p_1; d_1)$ returns $(d_1, f_k(d_1) + p_1)$; and $Dec_k(d, r)$ returns $r - f_k(d)$. CPA-security, homomorphism and reproducibility of $\mathcal{E}$ are clear. Finally, note that since $f_k(0) = 0$, we have $E_k(p; 0) = (0, p)$, which verifies the degenerate case of homomorphism. □

[4] The domain and the key spaces may themselves come with an associated distribution, but we leave this point implicit for simplicity.

6.2 Homomorphic hash-proof systems to homomorphic weak PRFs

We first review the notion of a *homomorphic hash-proof system* (HHPS), originally defined in [13]. Then we realize homomorphic weak PRFs using an HHPS.

A HHPS $\mathsf{HHPS} = (\mathsf{Param}, \mathsf{Priv}, \mathsf{Pub})$ is described as follows. The randomized *setup* algorithm $\mathsf{Param}(\cdot)$ takes as input a security parameter 1^λ and outputs *public parameters* $\mathsf{HP} = (\mathsf{C}, \mathsf{C}_v, \mathsf{W}, \mathsf{K}, \mathsf{SK}, \mathsf{PK}, \mu : \mathsf{SK} \to \mathsf{PK}, \Lambda : \mathsf{SK} \times \mathsf{C} \to \mathsf{K})$, where, C is called the set of *ciphertexts*, $\mathsf{C}_v \subseteq \mathsf{C}$ the set of *valid ciphertexts*, W the set of *witnesses*, K the set of *plaintexts*, SK the set of *secret keys* and PK the set of *public keys*. We should point out that all these aforementioned sets are indeed descriptions of their actual sets. Each $\mathsf{c} \in \mathsf{C}_v$ admits a witness $\mathsf{w} \in \mathsf{W}$ of its membership in C_v, meaning that there exists a PPT relation R such that

$$\mathsf{c} \in \mathsf{C}_v \Leftrightarrow \exists \mathsf{w} \in \mathsf{W} \text{ s.t. } \mathsf{R}(\mathsf{c}, \mathsf{w}) = 1.$$

We assume it is efficiently possible to generate a uniform element from C_v along with a corresponding witness, and also to sample uniformly from SK and K.

The efficient *private evaluation* algorithm Priv takes as input $\mathsf{sk} \in \mathsf{SK}$ and $\mathsf{c} \in \mathsf{C}$, and deterministically computes $\mathsf{Priv}_{\mathsf{sk}}(\mathsf{c}) = \Lambda(\mathsf{sk}, \mathsf{c})$. The efficient *public evaluation* algorithm Pub, takes as input $\mathsf{pk} = \mu(\mathsf{sk})$, $\mathsf{c} \in \mathsf{C}_v$ and a witness w for c, and deterministically computes $\mathsf{Pub}_{\mathsf{pk}}(\mathsf{c}, \mathsf{w}_\mathsf{c}) = \Lambda(\mathsf{sk}, \mathsf{c})$. Finally, we require HHPS to satisfy the following properties.

Subset membership: For every adversary $\mathcal{A}$, given all the public parameters of the scheme, it holds that

$$|\Pr[\mathcal{A}(\mathsf{c_v}) = 1] - \Pr[\mathcal{A}(\mathsf{c_{inv}}) = 1]| = negl(\lambda),$$

where, $\mathsf{c_v} \leftarrow \mathsf{C}_v$, $\mathsf{c_{inv}} \leftarrow \mathsf{C} \setminus \mathsf{C}_v$ and the probabilities are computed over the random coins of the adversary and over the choices of $\mathsf{c_v}$ and $\mathsf{c_{inv}}$, and also over the choices of C and C_v, which are taken from the output of $\mathsf{Param}(1^\lambda)$.

Smoothness: It holds that $\Delta[(\mathsf{pk}, \mathsf{Priv}_{\mathsf{sk}}(\mathsf{c}), \mathsf{c}), (\mathsf{pk}, \mathsf{k}, \mathsf{c})] = negl(\lambda)$, where $\mathsf{c} \leftarrow \mathsf{C} \setminus \mathsf{C}_v$, $\mathsf{k} \leftarrow \mathsf{K}$, $\mathsf{sk} \leftarrow \mathsf{SK}$ and $\mathsf{pk} = \mu(\mathsf{sk})$.

Homomorphism: $(\mathsf{C}, +)$, $(\mathsf{C}_v, +)$ and $(\mathsf{K}, +)$ admit groups (with efficient group operations), and, for every sk, it holds that $\Lambda(sk, \cdot)$ constitutes a homomorphism, i.e., for every $\mathsf{sk} \in \mathsf{SK}$ and $\mathsf{c_1}, \mathsf{c_2} \in \mathsf{C}$, it holds that,

$$\Lambda(\mathsf{sk}, \mathsf{c_1}) + \Lambda(\mathsf{sk}, \mathsf{c_2}) = \Lambda(\mathsf{sk}, \mathsf{c_1} + \mathsf{c_2}).^5$$

We now show how to construct a homomorphic weak PRF from a HHPS.

Theorem 4. *Assuming the existence of a HHPS, there exists a homomorphic weak PRF.*

Proof. Assume that $\mathsf{HHPS} = (\mathsf{Param}, \mathsf{Priv}, \mathsf{Pub})$ is a HHPS. Let

$$\mathsf{HP} = (\mathsf{C}, \mathsf{C}_v, \mathsf{W}, \mathsf{K}, \mathsf{SK}, \mathsf{PK}, \mu : \mathsf{SK} \to \mathsf{PK}, \Lambda : \mathsf{SK} \times \mathsf{C} \to \mathsf{K})$$

be the public parameters of HHPS produced by running Param. The tuple HP will also be the public parameters of our PRF, F, constructed as follows. We set $K_\lambda = \mathsf{SK}$, $D_\lambda = \mathsf{C}_v$ and $R_\lambda = \mathsf{K}$, and define $f_{\mathsf{sk}}(c) = \Lambda_{\mathsf{sk}}(\mathsf{c})$. We have that both C_v and K admit groups and that $f_{\mathsf{sk}}(c_1) + f_{\mathsf{sk}}(c_2) = f_{\mathsf{sk}}(\mathsf{c_1} + \mathsf{c_2})$, which implies homomorphism for PRF F. To prove weak pseudorandomness for F we need to show that, for any $p = p(\lambda)$, it holds that $\mathcal{DS} \equiv^c \mathcal{DS}'$, where

$$\mathcal{DS} = (\mathsf{c_1}, \Lambda_{\mathsf{sk}}(\mathsf{c_1})), \ldots, (\mathsf{c_p}, \Lambda_{\mathsf{sk}}(\mathsf{c_p}))$$
$$\mathcal{DS}' = (\mathsf{c_1}, \mathsf{k_1}), \ldots, (\mathsf{c_p}, \mathsf{k_p}),$$

[5] We remark that in many settings the homomorphism of C_v is implied by that of C: Especially in the standard setting, where the set of valid ciphertexts is defined as those, for which the value of $\Lambda(\mathsf{sk}, \cdot)$, for any sk is determined solely from the ciphertexts itself and $\mu(\mathsf{sk})$. However, we put it as a separate condition just to be as general as possible.

for $\mathsf{c}_1, \ldots, \mathsf{c}_p \leftarrow \mathsf{C}_v$, $\mathsf{k}_1, \ldots, \mathsf{k}_p \leftarrow \mathsf{K}$ and $\mathsf{sk} \leftarrow \mathsf{SK}$. To this end, for $0 \leq i \leq p$, we define the hybrid $\mathcal{DS}_i$ as follows.

$$\mathcal{DS}_i = ((\mathsf{c}_1, \Lambda_{\mathsf{sk}}(\mathsf{c}_1)), \ldots, (\mathsf{c}_i, \Lambda_{\mathsf{sk}}(\mathsf{c}_i)), (\mathsf{c}_{i+1}, \mathsf{k}_{i+1}), \ldots, (\mathsf{c}_p, \mathsf{k}_p)), \tag{13}$$

where $\mathsf{c}_1, \ldots, \mathsf{c}_p$, $\mathsf{k}_1, \ldots, \mathsf{k}_p$ and sk are sampled as above. Note that $\mathcal{DS}' = \mathcal{DS}_0$ and $\mathcal{DS} = \mathcal{DS}_p$. Now to conclude the proof for each $0i$ we show $\mathcal{DS}_i \equiv^c \mathcal{DS}_{i+1}$.

Note that we have $(\mathsf{pk}, \mathsf{c}_{i+1}, \Lambda_{\mathsf{sk}}(\mathsf{c}_{i+1})) \equiv^c (\mathsf{pk}, \mathsf{c}_{i+1}, \mathsf{k}_{i+1})$. This follows by combining the subset membership and smoothness properties of HHPS. Now we claim that $\mathcal{DS}_i = \mathcal{DS}_{i+1}$ follows from the fact that was just given: to see this, given $(\mathsf{pk}, \mathsf{c}_{i+1}, *)$, where $*$ either corresponds to $\Lambda_{\mathsf{sk}}(\mathsf{c}_{i+1})$ or to k_{i+1}, we form

$$((\mathsf{c}_1, \mathsf{Pub}_{\mathsf{pk}}(\mathsf{c}_1, \mathsf{w}_1)), \ldots, (\mathsf{c}_i, \mathsf{Pub}_{\mathsf{pk}}(\mathsf{c}_i, \mathsf{w}_i)), (\mathsf{c}_{i+1}, *), (\mathsf{c}_{i+2}, \mathsf{k}_{i+2}), \ldots, (\mathsf{c}_p, \mathsf{k}_p)), \tag{14}$$

where, for $1 \leq j \leq i$, we sample $\mathsf{c}_j \leftarrow \mathsf{C}_v$ along with a witness w_j, and sample $\mathsf{c}_{i+2}, \ldots, \mathsf{c}_p \leftarrow \mathsf{C}_v$ and $\mathsf{k}_{i+2}, \ldots, \mathsf{k}_p \leftarrow \mathsf{K}$. The distribution given in Equation 14 would either correspond to $\mathcal{DS}_i$ or to $\mathcal{DS}_{i+1}$. □

6.3 Realization under subgroup indistinguishability assumptions

For the sake of clarity, in this section we give an instantiation of our encryption primitive based only on the *quadratic residuosity* assumption, which is a special case of the subgroup indistinguishability (SG) assumption. We leave the general SG-based instantiation to the full version [24].

We first start by reviewing the *quadratic residuosity* assumption. For an *RSA number* N (i.e., $N = pq$, where p and q are distinct odd primes) we use $\mathcal{QR}_N$ to denote the subset of $\mathbb{Z}_{\mathbb{N}}^*$ consisting of quadratic residues modulo N, and let $\mathcal{J}_N$ denote the set of elements in $\mathbb{Z}_{\mathbb{N}}^*$ with Jacobi symbol one. Finally, we define $\mathcal{QNR}_N = \mathcal{J}_N \setminus \mathcal{QR}_N$.

Assume that $\mathsf{RSAGen}(1^\lambda)$ is a PPT algorithm that on input 1^λ generates a *Blum integer* N, i.e., $N = pq$ with p and q being distinct primes satisfying $p, q \equiv 3 \pmod 4$. We stress here that we do not need $\mathsf{RSAGen}(1^\lambda)$ to output the factorization of N as well. We say that the quadratic residuosity (QR) problem is hard under RSAGen if $\{N, U(\mathcal{QR}_N)\}_{\lambda \in \mathbb{N}}$ is computationally indistinguishable from $\{N, U(\mathcal{QNR}_N)\}_{\lambda \in \mathbb{N}}$, where N is generated according to $RSAGen(1^\lambda)$.

Theorem 5. *Assuming the quadratic residuosity assumption holds for* RSAGen *there exists a CPA-secure private-key bit encrypiton scheme that is both reproducible and homomorphic.*

Proof. We construct the private-key bit encryption scheme (G, E, Dec) as follows. The public parameter of the scheme is $N \leftarrow \mathsf{RSAGen}(1^\lambda)$, and the plaintext group and the randomness group of the scheme are, respectively, $\mathbb{Z}_2$ and $\mathcal{QR}_N$. The components of the encryption scheme are defined as follows. (All computations, if not otherwise stated, are done modulo N.)

- $G(1^\lambda)$: Choose the secret key as $x \leftarrow \mathbb{Z}_{N^2}$;
- $E_x(b; g)$: return $(g, (-1)^b g^x)$;
- $Dec_x(g_1, g_2)$: return $b \in \{0, 1\}$ if $g_2 = (-1)^b g_1^x$.

We first verify the syntactic properties required of the scheme. Notice that given an encryption $(g, (-1)^b g^{x_1})$ (of an arbitrary bit b) under x_1, we can efficiently obtain the encryption of an arbitrary bit b_1 under the same randomness, g, relative to a secret key x_2 by simply outputting $(g, (-1)^{b_1} g^{x_2})$. This verifies the reproducibility property. As for homomorphism, from $(g_1, (-1)^{b_1} g_1^x)$ and $(g_2, (-1)^{b_2} g_2^x)$, we can easily derive $(g_1 g_2, (-1)^{b_1 + b_2} (g_1 g_2)^x)$, which is the encryption of $b_1 + b_2$ under randomness $g_1 g_2$ (relative to the same unknown secret key x). Note that as the randomness group here is multiplicative, we will denote the identity element by 1. We then have that $E_x(b; 1) = (1, (-1)^b)$, independently of x. This verifies the degenerate case of homomorphism.

To show that the above scheme is CPA-secure, we need to show that for any $p = p(\lambda)$ and any sequence of bits $(b_1, \ldots, b_p)$, it holds that $\mathcal{DS}_0 \equiv^c \mathcal{DS}_1$, where

$$\mathcal{DS}_0 = \begin{bmatrix} g_1 & g_2 & \cdots & g_p \\ (-1)^{b_1} g_1^x & (-1)^{b_2} g_2^x & \ldots & (-1)^{b_p} g_p^x \end{bmatrix}, \text{ and} \tag{15}$$

$$\mathcal{DS}_1 = \begin{bmatrix} g_1 & g_2 & \cdots & g_p \\ g_1^x & g_2^x & \cdots & g_p^x \end{bmatrix}, \tag{16}$$

for $g_1, \ldots, g_p \leftarrow \mathcal{QR}_N$ and $x \leftarrow \mathbb{Z}_{N^2}$. The proof of the above indistinguishability is standard. (See, e.g., [13,29] for a simple proof and also [9, Lemma 5.1] for a stronger statement.) □

7 Extensions

In this section we discuss some extensions and complementary results. In Subsection 7.1 we show that our constructed scheme provides *auxiliary-input security*. In Subsection 7.2 we show that an existing *KDM-amplification* construction preserves leakage resilience.

7.1 Auxiliary-input security

We first give the definitions related to auxiliary-input security.

Background. Let $\mathcal{E} = (G, E, Dec)$ be an encryption scheme with public-key, secret-key and message spaces, respectively, $\mathcal{PK}_\lambda$, $\mathcal{SK}_\lambda$ and $\mathcal{M}_\lambda$. Throughout this Section we use f to refer to a function with domain $(\mathcal{PK}_\lambda, \mathcal{SK}_\lambda)$ and range $\mathcal{SK}_\lambda$. We follow the notation of [9]. For $\mathcal{E} = (G, E, Dec)$ we define *f-weak inversion* and *f-strong inversion* as follows. We say that f is *ϵ-strongly-uninvertible* under $\mathcal{E}$ if for any adversary $\mathcal{A}$, the probability that $\mathcal{A}$ outputs sk when given $(f(pk, sk), pk)$ is at most $\epsilon(\lambda)$, where the probability is taken over $\mathcal{A}$'s random coins and $(pk, sk) \leftarrow G(1^\lambda)$. Also, we say that f is *ϵ-weakly-uninvertible* under $\mathcal{E}$ if for any adversary $\mathcal{A}$, the probability that $\mathcal{A}$ outputs sk when given $f(pk, sk)$ is at most $\epsilon(\lambda)$, where the probability is taken over $\mathcal{A}$'s random coins and $(pk, sk) \leftarrow G(1^\lambda)$. Let Aux_ϵ^{st} be the class of all ϵ-strongly-uninvertible functions and Aux_ϵ^{wk} be the class of all ϵ-weakly-uninvertible functions. Note that $Aux_\epsilon^{st} \subseteq Aux_\epsilon^{wk}$.

We say that $\mathcal{E}$ is *f-auxiliary-input secure* if any adversary $\mathcal{A}$ has a negligible advantage in the following game: $\mathcal{A}$ is given $(pk, f(pk, sk))$, where $(pk, sk) \leftarrow G(1^\lambda)$; $\mathcal{A}$ submits $(m_0, m_1) \in \mathcal{M}_\lambda^2$; $\mathcal{A}$ receives $E_{pk}(m_b)$, for $b \leftarrow \{0, 1\}$; finally, $\mathcal{A}$ outputs bit b', and achieves the following advantage

$$\left|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]\right|.$$

We say that $\mathcal{E}$ is *ϵ-weakly-auxiliary-input secure* (resp., ϵ-strongly-auxiliary-input secure) if $\mathcal{E}$ is *f-auxiliary-input secure* for any $f \in Aux_\epsilon^{st}$ (resp., Aux_ϵ^{wk}). We say $\mathcal{E}$ is auxiliary-input secure against *subexponentially-hard functions* if for some $c > 0$, $\mathcal{E}$ is $1/(2^{\lambda^c})$-strongly-auxiliary-input secure.

We now show that the encryption scheme produced by Construction 1 provides auxiliary-input security. We first consider weak-auxiliary-input security and then discuss the extension to the strong-auxiliary case.

Theorem 6. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure private-key bit-encryption scheme providing degenerate homomorphism and reproducibility. Let $\mathcal{E}'$ be the scheme constructed from $\mathcal{E}$ using Construction 1. For any poly-bounded $l = l(\lambda)$ and negligible function $\epsilon = \epsilon(\lambda)$, it holds that $\mathcal{E}'$ is ϵ-weakly-auxiliary-input secure.*[6]

The proof of Theorem 6 follows similarly to that of Theorem 3, except for one step, where we replace real-randomness extraction with *pseudorandomness extraction*. We first give the following theorem, due to Goldreich and Levin [22], where we follow the presentation of [14], adapted to the binary field.

Theorem 7. *([22]) Assume that $l = l(\lambda)$ and $h\colon \{0,1\}^l \to \{0,1\}^*$ is a (possibly randomized) function and $\mathcal{D}$ is a distinguisher, where*

$$\left|\Pr[D(\mathbf{b}, b, h(\mathbf{s})) = 1] - \Pr[D(\mathbf{b}, b', h(\mathbf{s})) = 1]\right| = \delta(l), \tag{17}$$

[6] In order for statement to be useful, it should hold that $\frac{1}{2^l} \leq \epsilon$, because otherwise the statement will be vacuously true, as $Aux_\epsilon^{st} = Aux_\epsilon^{wk} = \emptyset$.

where $\mathbf{s}, \mathbf{b} \leftarrow \{0,1\}^l$, $b \leftarrow \{0,1\}$ *and* $b' = \mathbf{s} \cdot \mathbf{b}$. *Then there exists an inverter* $\mathcal{A}$, *for which it holds that*

$$\Pr[\mathcal{A}(y) = \mathbf{s}] \in \Omega(\frac{\delta^3}{l}), \tag{18}$$

where $\mathbf{s} \leftarrow \{0,1\}^l$ *and* $y \leftarrow h(\mathbf{s})$.

We now give the proof of Theorem 6, using ideas from [14].

Proof. The proof follows by introducing *Game*-0, *Game*-1 and *Game*-2 exactly as in the proof of Theorem 3 (except that now the function f is applied to both the secret key and the public key), and deriving *Game*-$0 \equiv^G$ *Game*-1 exactly as in there. To prove *Game*-$1 \equiv^G$ *Game*-2, however, we proceed as below. To prove *Game*-$1 \equiv^G$ *Game*-2, it suffices to show that

$$\begin{aligned}(b_1, \ldots, b_l, b_{l+1}, f(PK, \mathbf{s}), \overbrace{E_{sk}(0; r_1), \ldots, E_{sk}(0; r_l), E_{sk}(0; r_{l+1})}^{PK}) \equiv^c \\ (b_1, \ldots, b_l, b'_{l+1}, f(PK, \mathbf{s}), \underbrace{E_{sk}(0; r_1), \ldots, E_{sk}(0; r_l), E_{sk}(0; r_{l+1})}_{PK}),\end{aligned} \tag{19}$$

where $\mathbf{s} \leftarrow \{0,1\}^l$, $b_1, \ldots, b_l, b_{l+1} \leftarrow \{0,1\}$, $b'_{l+1} = \mathbf{s} \cdot (b_1, \ldots, b_l)$, $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathcal{R}_\lambda^l$, $r_{l+1} = \mathbf{s} \cdot \mathbf{r}$ and $sk \leftarrow G(1^\lambda)$. The fact that proving Equation 19 suffices to conclude *Game*-$1 \equiv^G$ *Game*-2 can easily be verified by considering the descriptions of *Game*-1 and *Game*-2, taking into account the fact that the private-key scheme is reproducible.

By the assumption of the theorem, we know that it is ϵ-hard to recover $\mathbf{s}$ from $(PK, f(PK, \mathbf{s}))$. Now Equation 19 follows from Theorem 7, by defining the randomized function $h(\mathbf{s}) = (PK, f(PK, \mathbf{s}))$, where all the variables are sampled as above. Formally, if there is an adversary that can distinguish between the distributions in Equation 19 with a non-negligible probability, then there exists an adversary that, with a non-negligible probability, recovers $\mathbf{s}$ from $h(\mathbf{s}) = (PK, f(PK, \mathbf{s}))$, which is a contradiction to the first sentence of this paragraph.

Remark 2. As in previous work [14,9] we can prove strong auxiliary-input security for $\mathcal{E}'$ with respect to subexponentially-hard functions by working with a modification of Construction 1, letting $(c_1, \ldots, c_l) = (E_{sk}(0; r_1), \ldots, E_{sk}(0; r_l))$ be the public parameters of the scheme, and letting the public key be computed, under secret key $\mathbf{s}$, as $Hom(c_{i_1}, \ldots, c_{i_w})$, where $(i_1, \ldots, i_w)$ are the indices of non-zero bits of $\mathbf{s}$. Now since a public key under the new scheme has at most $l' = |\mathcal{R}_\lambda|$ different values we can obtain $\frac{\epsilon}{l'}$-strong auxiliary-input security from ϵ-weak-auxiliary-input security. This last step follows since, for any scheme with l' different public keys, if recovering sk from $f(pk, sk)$ is ϵ/l'-hard (i.e., succeeds with a probability at most ϵ/l'), recovering sk from $(f(pk, sk), pk)$ is ϵ-hard. Finally, we mention that the proof of multiple-key circular security (Theorem 2) extends to the setting above which contains public parameters.

7.2 KDM amplification

We show that Applebaum's KDM-amplification method [3], which, informally speaking, shows that projection security is sufficient for obtaining "rich-KDM" security, preserves both types of leakage resilience. For simplicity, we focus on the case of bit encryption and 1-KDM security.

As notation, we identify an efficiently computable function $f = \{f_\lambda \colon \{0,1\}^{l(\lambda)} \mapsto \{0,1\}\}_{\lambda \in \mathbb{N}}$ with an ensemble of circuits $\{c_\lambda\}_{\lambda \in \mathbb{N}}$, and say that f has size $p = p(\lambda)$ if, for any λ, the circuit c_λ has size at most p. We say an ensemble of sets of functions $F = \{F_\lambda\}_{\lambda \in \mathbb{N}}$ is p-bounded if for every λ and every $f \in F_\lambda$, f has size p. The following theorem is a special case of the results of [3].

Theorem 8. *([3]) Assume that* $F = \{F_\lambda\}_\lambda$ *is a fixed* p*-bounded ensemble of sets of functions and* $\mathcal{E} = (G, E, Dec)$ *is a* 1*-projection-secure public-key encryption scheme. The scheme* $\mathcal{E}' = (G, E', D')$, *constructed below, is* F*-KDM*$^{(1)}$ *secure:* $E'_{pk}(b) = E_{pk}(Sim(b))$ *and* $D'_{sk}(C) = Rec(D_{sk}(C))$. *Here* Sim *is a randomized function and* Rec *is a deterministic function, both of which are constructed based on* F, *through the procedure of* randomized encoding of functions. *The details of* Sim *and* Rec *are not important for our analysis, bu we refer the reader to [3] for further details.*

Theorem 9. *Let $\mathcal{E}$ and $\mathcal{E}'$ be as in Theorem 8. Then assuming that $\mathcal{E}$ is r-rate leakage resilient (resp., ϵ-auxiliary input secure) then $\mathcal{E}'$ is r-rate leakage resilient (resp., ϵ-auxiliary input secure).*

Proof. This follows by noting that the constructed scheme $\mathcal{E}'$ has the same key generation algorithm as that $\mathcal{E}$. We consider the leakage resilience case; the proof for the auxiliary-input case is entirely the same. Assume $\mathcal{A}'$ wins against ℓ-length leakage resilience of $\mathcal{E}'$; we build $\mathcal{A}$ that breaks the ℓ-length leakage resilience of $\mathcal{E}'$ by simulating $\mathcal{A}'$ as follows: $\mathcal{A}$ runs $\mathcal{A}'(pk)$, where pk is the public key that $\mathcal{A}$ receives; when $\mathcal{A}'$ sub,its the leakage query f, $\mathcal{A}$ makes the same query from its oracle and gives $f(sk)$ to $\mathcal{A}'$; finally, when $\mathcal{A}'$ submits (b_0, b_1), $\mathcal{A}$ submits $(Sim(b_0), Sim(b_1))$ to its oracle and gives the returned ciphertext to $\mathcal{A}'$. Thus, $\mathcal{A}$ achieves the same advantage as $\mathcal{A}'$ does, and the proof is complete. □

We now obtain the following corollary, by combining Theorems 2, 3, 6 and 9.

Corollary 1. *Assuming the existence of a CPA-secure private-key scheme with reproducibility and degenerate homomorphism, for any poly p and any fixed p-bounded function family F, there exists a scheme $\mathcal{E}'$ which (at the same time) (1) is F-KDM secure, (2) achieves a $(1 - o(1))$ resilience rate, and (3) is auxiliary-input secure against subexponentially-hard functions.*

8 Acknowledgments

We would like to thank Josh Benaloh and Dan Boneh for helpful discussions.

References

1. Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *TCC*, volume 5444 of *LNCS*, pages 474–495. Springer, 2009.
2. Joël Alwen, Yevgeniy Dodis, Moni Naor, Gil Segev, Shabsi Walfish, and Daniel Wichs. Public-key encryption in the bounded-retrieval model. In Gilbert [21], pages 113–134.
3. Benny Applebaum. Key-dependent message security: Generic amplification and completeness. *J. Cryptology*, 27(3):429–451, 2014.
4. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, 2009.
5. Boaz Barak, Iftach Haitner, Dennis Hofheinz, and Yuval Ishai. Bounded key-dependent message security. In Gilbert [21], pages 423–444.
6. Mihir Bellare, Alexandra Boldyreva, and Jessica Staddon. Randomness re-use in multi-recipient encryption schemeas. In *Public Key Cryptography - PKC 2003*, pages 85–99, 2003.
7. John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In *SAC 2002, Selected Areas in Cryptography*, volume 2595 of *LNCS*, pages 62–75. Springer, 2002.
8. Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. In *CRYPTO 2008*, pages 108–125, 2008.
9. Zvika Brakerski and Shafi Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). In *Advances in Cryptology - CRYPTO 2010*, pages 1–20, 2010.
10. Zvika Brakerski, Shafi Goldwasser, and Yael Tauman Kalai. Black-box circular-secure encryption beyond affine functions. In Ishai [28], pages 201–218.
11. Zvika Brakerski and Gil Segev. Better security for deterministic public-key encryption: The auxiliary-input setting. In *CRYPTO 2011, volume 6841 of LNCS*, pages 543–560. Springer, 2011.
12. Seung Geol Choi and Hoeteck Wee. Lossy trapdoor functions from homomorphic reproducible encryption. *Inf. Process. Lett.*, 112(20):794–798, 2012.
13. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, 2002.

14. Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *Theory of Cryptography, TCC 2010*, volume 5978 of *LNCS*, pages 361–381. Springer, 2010.
15. Yevgeniy Dodis, Yael Tauman Kalai, and Shachar Lovett. On cryptography with auxiliary input. In *STOC 2009*, pages 621–630, 2009.
16. Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.
17. Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *STOC 1991*, pages 542–552, 1991.
18. Cynthia Dwork, editor. *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. ACM, 2008.
19. Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS 2008*, pages 293–302, 2008.
20. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Dwork [18], pages 197–206.
21. Henri Gilbert, editor. *Advances in Cryptology - EUROCRYPT 2010,*, volume 6110 of *LNCS*. Springer, 2010.
22. Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32. ACM, 1989.
23. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.
24. Mohammad Hajiabadi, Bruce M. Kapron, and Venkatesh Srinivasan. On generic constructions of circularly-secure, leakage-resilient public-key encryption schemes. *IACR Cryptology ePrint Archive*, 2015:741, 2015.
25. Shai Halevi and Huijia Lin. After-the-fact leakage in public-key encryption. In Ishai [28], pages 107–124.
26. Carmit Hazay, Adriana López-Alt, Hoeteck Wee, and Daniel Wichs. Leakage-resilient cryptography from minimal assumptions. In *Advances in Cryptology - EUROCRYPT 2013*.
27. Dennis Hofheinz and Dominique Unruh. Towards key-dependent message security in the standard model. In *Advances in Cryptology - EUROCRYPT 2008*.
28. Yuval Ishai, editor. *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, volume 6597 of *LNCS*. Springer, 2011.
29. Eike Kiltz, Krzysztof Pietrzak, Martijn Stam, and Moti Yung. A new randomness extraction paradigm for hybrid encryption. In *Advances in Cryptology - EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 590–609. Springer, 2009.
30. Venkata Koppula, Kim Ramchen, and Brent Waters. Separations in circular security for arbitrary length key cycles. In *TCC 2015*.
31. Tal Malkin, Isamu Teranishi, and Moti Yung. Efficient circuit-size independent public key encryption with kdm security. In *Advances in Cryptology – EUROCRYPT 2011*, pages 507–526, 2011.
32. Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In *Theory of Cryptography, TCC 2004*, volume 2951 of *LNCS*, pages 278–296. Springer, 2004.
33. Moni Naor, Benny Pinkas, and Omer Reingold. Distributed pseudo-random functions and kdcs. EUROCRYPT'99, pages 327–346, 1999.
34. Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *J. Comput. Syst. Sci.*, 58(2), 1999.
35. Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. *SIAM J. Comput.*, 41(4):772–814, 2012.
36. Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC 1990*, pages 427–437, 1990.
37. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Dwork [18], pages 187–196.
38. Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology - CRYPTO 1991*, volume 576 of *LNCS*, pages 433–444. Springer, 1991.
39. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC 2005*, pages 84–93, 2005.
40. Ron Rothblum. Homomorphic encryption: From private-key to public-key. In Ishai [28], pages 219–234.
41. Ron Rothblum. On the circular security of bit-encryption. In *TCC*, pages 579–598, 2013.
42. Hoeteck Wee. KDM-security via homomorphic smooth projective hashing. *IACR Cryptology ePrint Archive*, 2015:721, 2015.

A Appendix

A.1 A scheme that preserves the plaintext space

Construction 1 yields a bit-encryption scheme, even when the scheme we start with has a larger plaintext space. We now describe a scheme that maintains the plaintext space of the original scheme, at the cost of making two additional syntactic assumption, aimed at making the decryption algorithm efficient. We stress here that these new conditions are required solely for the efficiency of decryption, and are not used anywhere in security proofs. We show that all these assumptions are satisfied for encryptions schemes built based on homomorphic weak PRFs.

Definition 4. *Assume $\mathcal{E} = (G, E, Dec, Hom, Rep)$ is a private-key scheme that provides homomorphism and reproducibility. We define two new conditions for the scheme.*

1. *There exists an efficient algorithm $RandInv$, such that for every $sk \in G(1^\lambda)$ and $r \in \mathcal{R}_\lambda$, it holds that*

$$RandInv(E_{sk}(0;r)) = E_{sk}(0;-r)).$$

2. *For every $sk \in G(1^\lambda)$ and $m \in \mathcal{M}_\lambda$, one can efficiently compute m from the encryption of m under the identity element, $E_{sk}(m;0)$.*

We now give the plaintext-preserving construction.

Construction 2 *(Plaintext-preserving encryption)*
Assume that $\mathcal{E} = (G, E, Dec, Hom, Rep)$ satisfies the conditions in Definition 4.

- *Key generation G': Choose the secret key as $\mathbf{s} \leftarrow \{0,1\}^l$, and the public key as*

$$(E_{sk}(0;r_1), \ldots, E_{sk}(0;r_l), E_{sk}(0;\mathbf{s}\cdot\mathbf{r}))\,,$$

  where $sk \leftarrow G(1^\lambda)$, $r_1, \ldots, r_l \leftarrow \mathcal{R}_\lambda$ and $\mathbf{r} = (r_1, \ldots, r_l)$.
- *Encryption E': To encrypt plaintext $m \in \mathcal{M}_\lambda$ under public key $(c_1, \ldots, c_l, c_{l+1})$, do the following: choose $sk' \leftarrow G(1^\lambda)$ and return $(c'_1, \ldots, c'_l, c'_{l+1})$, where $c'_i = Rep(c_i, 0, sk')$, for $1 \le i \le l$, and $c'_{l+1} = Rep(c_{l+1}, m, sk')$.*
- *Decryption D': To decrypt $(c'_1, \ldots, c'_l, c'_{l+1})$ under secret key s, letting $(i_1, \ldots, i_w)$ be the indices of non-zero bits of s, compute $c' = Hom(c_{i_1}, \ldots, c_{i_w}, c''_{l+1})$, where $c''_{l+1} = RandInv(c'_{l+1})$. Now recover m from c', using Condition 2 of Definition 4.*

In the following theorem we state the modified version of Theorems 2, 3 for the case where the initial scheme is not bit encryption. The proofs are entirely similar (except for a few simple changes) to the bit-encryption case, and so we omit them here.

Theorem 10. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure private-key encryption scheme providing degenerate-homomorphism and reproducibility. Moreover, assume that $\mathcal{E}$ provides the conditions in Definition 4 (aimed at having efficient decryption). Then, the scheme $\mathcal{E}' = (G', E', D')$ built in Construction 2 has the following properties.*

- *for any constant $c > 1$, by taking $l = cn \log(|\mathcal{R}_\lambda|) + \omega(\log\lambda)$, $\mathcal{E}'$ is n-projection secure; and*
- *$\mathcal{E}'$ is $(l - \log|\mathcal{R}_\lambda| - \log|\mathcal{M}_\lambda| - u)$-length leakage resilient, for any $u \in \omega(\log\lambda)$. Moreover, by taking $l = \omega(\log|\mathcal{R}_\lambda|) + \omega(\log|\mathcal{M}_\lambda|) + \omega(\log\lambda)$, the constructed scheme achieves a leakage rate of $(1 - o(1))$.*

A.2 From subgroup indistinguishability to homomorphic reproducible encryption

We refer the reader to [9] for the definition of subgroup indistinguishability assumptions. We use the same notation as in [9] and refer the reader to [9] for a detailed definition. .

Theorem 11. *Assuming the subgroup indistinguishability assumption holds there exists a CPA-secure private-key encryption scheme that is both reproducible and degenerate-homomorphic.*

Proof. We construct the scheme as follows. The plaintext group and the randomness group of the scheme are, respectively, G_M and G_L. The components of the encryption scheme are defined as follows.

- $G(1^\lambda)$: Choose the secret key as $x \leftarrow \mathbb{Z}_{T^2}$;
- $E_x(m; r)$: return $(r, m \cdot r^x)$;
- $D_x(g_1, g_2)$: return $g_2 \cdot (g_1^x)^{-1}$.

The proof of CPA security of the scheme follows from the facts proved in [9] (specifically Lemma B.1.). Moreover, it is not hard to show that the scheme provides all the properties needed by Theorem 10; we omit the details. □