IBE with Incompressible Master Secret and Small Identity Secrets

Nico Döttling[1], Sanjam Garg[2], Sruthi Sekar[3], and Mingyuan Wang[3]

[1]Helmholtz Center for Information Security (CISPA) nico.doettling@gmail.com
[2]University of California, Berkeley and NTT Research sanjamg@berkeley.edu
[3]University of California, Berkeley {sruthi,mingyuan}@berkeley.edu

May 25, 2022

Abstract

Side-stepping the protection provided by cryptography, exfiltration attacks are becoming a considerable real-world threat. With the goal of mitigating the exfiltration of cryptographic keys, big-key cryptosystems have been developed over the past few years. These systems come with very large secret keys which are thus hard to exfiltrate. Typically, in such systems, the setup time must be large as it generates the large secret key. However, subsequently, the encryption and decryption operations, that must be performed repeatedly, are required to be efficient. Specifically, the encryption uses only a small public key and the decryption only accesses small ciphertext-dependent parts of the full secret key. Nonetheless, these schemes require decryption to have access to the entire secret key. Thus, using such big-key cryptosystems necessitate that users carry around large secret-keys on their devices, which can be a hassle and in some cases might also render exfiltration easy.

With the goal of removing this problem, in this work, we initiate the study of big-key identity-based encryption (bk-IBE). In such a system, the master secret-key is allowed to be large but we require that the identity-based secret keys are short. This allows users to use the identity-based short keys as the ephemeral secret keys that can be more easily carried around and allow for decrypting ciphertexts matching a particular identity, e.g. messages that were encrypted on a particular date. In particular:

- We build a new definitional framework for bk-IBE capturing a range of applications. In the case when the exfiltration is small our definition promises stronger security — namely, an adversary can break semantic security for only a few identities, proportional to the amount of leakage it gets. In contrast, in the catastrophic case where a large fraction of the master secret key has been ex-filtrated, we can still resort to a guarantee that the ciphertexts generated for a randomly chosen identity (or, an identity with enough entropy) remain protected. We demonstrate how this framework captures the best possible security guarantees.
- We show the first construction of such a bk-IBE offering strong security properties. Our construction is based on standard assumptions on groups with bilinear pairings and brings together techniques from seemingly different contexts such as leakage resilient cryptography, reusable two-round MPC, and laconic oblivious transfer. We expect our techniques to be of independent interest.

Contents

1 Introduction

Compromises of deployed cryptographic schemes by means of cryptanalysis are becoming increasingly rare. Instead, real-world adversaries try to circumvent the protection offered by cryptography via side-channel attacks. The most high-value targets for such side-channel attacks are cryptographic secret keys, which, if somehow exfiltrated, give the adversary unrestrained access to its victim's confidential communication. For advanced notions of public-key encryption such as identity-based encryption (IBE), exfiltration of the long-term master secret key is the single biggest risk coming with the adoption of such a system. This risk can be somewhat mitigated by distributing the master secret across several servers [BF01, Goy07, KG10, Cha12], but this comes with an additional overhead of maintaining multiple servers with shares of the master key.

Big-Key Cryptography in Bounded-Retrieval Model. The pervasiveness of side-channel attacks has motivated the development of cryptosystems that remain secure even when the adversary may have the ability to leak secrets of honest parties. One line of defense against such attacks, is to develop cryptosystems that have very large secret keys, or what is called *big-key* cryptography (see e.g., [Dzi06a, DLW06, CDD$^+$07, ADW09, ADN$^+$10, BKR16, MW20]). Big-key cryptosystems are developed with huge secret keys with the intent of making it hard to exfiltrate or leak on such keys. Furthermore, leakage of large amounts of data from a device can often be easier to detect and mitigate, or the bandwidth of any residual side-channels of such a device can be bounded conservatively[1]. Such cryptosystems aim to provide appropriate security even when a large amount of *arbitrary* leakage occurs on the big secret key. Prior works have focused on constructing various big-key primitives in the bounded-retrieval model, including symmetric-key encryption [BKR16], public-key encryption [ADN$^+$10, MW20] and authenticated key agreement [Dzi06b, CDD$^+$07, ADW09].

In the symmetric key setting [BKR16], the big-key setup involves a procedure to bound the adversary's probability of predicting an optimal length sub-key of the original exfiltrated big key, and using this to design an encapsulation mechanism that can extract a random key (such a key encapsulation mechanism directly gives an encryption scheme). Here, the encapsulation and decapsulation procedures only make local access to the big-key, thus ensuring efficiency. The key technique leveraged here is a primitive called "reusable locally-computable computational extractors" [Dzi06b, CDD$^+$07, BKR16].

On the other hand, in the public-key setting, only the secret key is big and prone to exfiltration, while the public key is still short. The efficiency goals are that the encryption and decryption running times do not grow with the size of the big secret key. This naturally leads to the decryption procedure only making a few local ciphertext-dependent access to the big secret key. The security goal in this setting is typically to achieve semantic security, even when the adversary can obtain arbitrary leakage on the big secret key. The security is only required for fresh ciphertexts that are generated after the leakage by the adversary. In contrast, no meaningful security can be offered for the old ciphertexts based on which the leakage can be performed, e.g., the adversary might obtain leakage corresponding to a few bits of the plaintext for a given ciphertext.

The use of big-keys in a big-key public-key encryption scheme limits their usability and principal practicality. In particular, a user does not a priori know what parts of a secret key it will need to decrypt a ciphertext on a particular device. Thus, the user must carry around the entire large secret keys on all her devices. This poses two challenges: (1) including large secret keys on a number of

[1] *Screaming Channels* [CPM$^+$18] are one such example, which optimistically transfers at most 1 bit per second.

devices can be a significant burden, e.g., wastage of limited storage space on a mobile device; and (2) the replication of a large secret key across multiple devices makes the user once again more susceptible to leakage based attacks, e.g., the loss of a mobile device could leak the entire big key.

1.1 Leakage-Resilient Identity-Based Encryption: Our Approach and Challenges

Motivated by these concerns, in this work, we will focus on the notion of identity-based encryption (IBE) as a natural proxy for encryption schemes that allow the delegation of decryption tokens. Recall that in an IBE scheme [BF01] a setup algorithm generates a pair $(\mathsf{mpk}, \mathsf{msk})$ of master public and master secret keys. The identity key generation algorithm takes the master secret key msk and an identity string id and outputs an identity secret key $\mathsf{sk}_{\mathsf{id}}$. To encrypt a message m, the encryption algorithm takes a master public key mpk and an identity string id and produces a ciphertext c. Finally, the decryption algorithm takes an identity secret key $\mathsf{sk}_{\mathsf{id}}$ and a ciphertext c and returns a message m. In terms of correctness, we require that if $\mathsf{sk}_{\mathsf{id}}$ is a user secret key corresponding to an identity id and a ciphertext c was encrypted to this same identity, then decrypting c with $\mathsf{sk}_{\mathsf{id}}$ returns the message that was encrypted.

Mapping our goal of designing a system with large long-term secrets but succinct public keys, ephemeral keys, and ciphertexts to the notion of IBE, we obtain the requirement that all system parameters except the master secret key should be succinct. We refer to this notion as big-key identity-based encryption (or bk-IBE for short).

Defining Security. In terms of security, the standard security notion for IBE requires that a ciphertext c^* encrypted to an identity id^* should remain secure, even if that adversary has access to any (polynomial number of) other secret keys $\mathsf{sk}_{\mathsf{id}}$ for $\mathsf{id} \neq \mathsf{id}^*$. Depending on whether the adversary has to specify the challenge identity id^* at the start of the experiment or is allowed to choose it adaptively depending on the master public key and some identity secret keys, we refer to selective or full security, respectively.

Now, when we consider (selective or full) security under leakage, the adversary additionally gets a leakage, $L(\mathsf{msk})$, on the master secret key. In the bounded retrieval model [Dzi06a, DLW06], we only limit the number of bits that the function L outputs, but otherwise allow L to perform any (efficient) computation on msk, i.e., L may try to somehow compress L first before producing its output. However, how does this notion of leakage resilience go along with our goal of making all system parameters small *except* the master secret key?

A moment of reflection points to the following dilemma, even in the setting of selective security: if the bit-length of the leakage function's output is allowed to be larger than the bit-length of an identity secret key, then the leakage function may just compute the key generation algorithm for the challenge identity id^* on msk and output the identity secret key $\mathsf{sk}_{\mathsf{id}^*}$ thus obtained. This makes the adversary's task of breaking the security of the challenge ciphertext c^* essentially trivial: The leakage $\mathsf{sk}_{\mathsf{id}^*}$ allows to recover the challenge message via the legitimate decryption algorithm!

For this reason, all prior works which studied the notion of leakage resilient IBE thus restricted themselves to a setting where the identity secret keys are large, and the master secret key is either large or permits no leakage [ADN+10, CDRW10, LRW11, HLWW13, CZLC16, NY19]. This brings us to the following question:

How can we meaningfully reconcile our design goal of short public parameters, identity *secret keys, and ciphertexts with security against large amounts of master secret key leakage?*

1.2 A New Security Notion and Construction for bk-IBE

From the above discussion, it is clear that we have to depart from the standard security notion of IBE. One way of relaxing the IBE security to circumvent the problem of exfiltration of the challenge identity key, described above, could consist of choosing the challenge identity at random or from a distribution of sufficiently high entropy, *after the adversary has obtained his leakage.*

While this indeed leads to a meaningful notion sufficient for certain use cases, the requirement of the challenge identity to be *entropic* puts restrictions on most of the use cases we envision. As an example, if the identities correspond to calendar dates, then choosing the challenge identity from a high entropy distribution would imply that the point in time corresponding to the challenge message necessarily needs to be highly uncertain — something that may not always be true.

However, we do expect exfiltration of a large portion of the already pretty big master secret key to be hard, particularly while also avoiding detection. Note that detection of leakage allows for alternative remedies such as revoking old keys and replacing them with new ones. Thus, a natural way to think of the leakage obtained by the adversary is as a *budget* of information about the master secret key, which we expect to be relatively smaller than the size of the master secret key. Of course, in a catastrophic event, a large fraction of the master secret key may be leaked, in which case, we would like to revert to the weaker entropic security guarantees.

The main intuition behind our new definition is as follows: the adversary may spend his exfiltration budget arbitrarily, and yet, he should not obtain more information than what he could get via a *trivial exfiltration attack*− leaking the identity secret keys of a number of challenge identities. Further, as mentioned above, catastrophic leakage of a large fraction of the master secret key still preserves entropic security.

In light of this, our security definition aims to capture how many identities the adversary could break. In particular, suppose the adversary obtains an ℓ-bit leakage from the master secret key. We define our big-key IBE to be secure if the adversary cannot break the security of $\geq \ell + 1$ number of identities. This is essentially the optimal security one could hope for as the adversary could potentially break $\Theta(\ell)$ identities by leaking a few bits for each identity.

Observe that this security notion is sufficiently strong for our applications. For instance, if the identities are the calendar dates, our security guarantees that an adversary leaking ℓ bits cannot break the security for more than ℓ days. Moreover, a random identity with sufficiently high entropy will also be secure since an adversary can break at most polynomially many identities.

Our Construction. Given this new security definition, we construct the first bk-IBE that achieves selective security based on the hardness of standard assumptions on groups with bilinear pairing. Our construction builds on seemingly very different tools such as leakage-resilient encryption scheme [HLWW13], reusable two-round MPC [BL20], and laconic OT [CDG+17].

Potential Extensions to ABE/HIBE. In the context of IBE, it is usual to also consider stronger encryption systems such as attribute-based encryption and hierarchical identity-based encryption, which typically offer a single small secret key that can be used to decrypt large families of ciphertexts. This is at odds with the goals of this paper, where we aim to not have a single short key that can decrypt large families of circuits, as such a key could end up getting leaked.

1.3 Technical Outline

bk-PKE via random selection. We will start by discussing the existing paradigms to construct bk-PKE and the challenges that arise when trying to adapt these techniques to the bk-IBE setting. One of the core ideas in the construction of bk-PKE [ADN+10, MW20] is *random selection.* For the sake of simplicity, let us drop the requirement of a short public key for a moment. Then there is a natural idea to construct bk-PKE via the following approach, as detailed in [ADN+10]. Let $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be any public key encryption scheme, and consider the following transformed scheme $(KEYGEN, ENC, DEC)$. The $KEYGEN$ algorithm produces a pair of public key $PK = (\mathsf{pk}_1, \ldots, \mathsf{pk}_\ell)$ and a secret key $SK = (\mathsf{sk}_1, \ldots, \mathsf{sk}_\ell)$ for a *largeness parameter* ℓ, where each key-pair $(\mathsf{pk}_i, \mathsf{sk}_i)$ has been independently generated. The encryption algorithm ENC takes the public key PK and a message m and selects a random subset $I = \{i_1, \ldots, i_\lambda\} \subseteq [\ell]$ of size (say) λ. Next, it computes a λ-out-of-λ secret sharing of $s_1, \ldots, s_\lambda$ of m (e.g. via additive secret sharing), computes ciphertexts $c_1 = \mathsf{Enc}(\mathsf{pk}_{i_1}, s_1), \ldots, c_\lambda = \mathsf{Enc}(\mathsf{pk}_{i_\lambda}, s_\lambda)$ and outputs the ciphertext $C = (I, c_1, \ldots, c_\lambda)$. To decrypt such a ciphertext C, the decryption algorithm DEC retrieves the secret keys sk_i (for $i \in I$) from SK, decrypts the c_i and reconstructs the message m.

Note first, that the ciphertext C is small (i.e., of size $\mathsf{poly}(\lambda, \log(\ell))$) and that both the encryption algorithm ENC and the decryption algorithm DEC are *local*, in the sense that ENC only accesses PK in λ location and DEC accesses SK in λ locations respectively.

Somewhat oversimplified, security is argued as follows, making critical use of the random selection of the set I: Given any leakage $L(SK)$ of size sufficiently smaller than ℓ bits, many of the individual secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_\ell$ will be information-theoretically hidden from the adversary. As the set I is chosen randomly *after* the leak $L(SK)$ has been computed, with very high probability over the choice of I, there is an index $i \in I$ for which $L(SK)$ contains essentially no information about sk_i. Thus, one can argue that the ciphertext component c_i hides the share s_i, and therefore the message m is hidden.

Returning to the issue of large public keys, compressing the public key PK while keeping the secret key SK incompressible was, in fact, the main technical challenge in the original construction of [ADN+10]. This was achieved via the notion of *identity-based hash-proof-systems.*

With more recently developed tools, namely laconic oblivious transfer, hash functions with encryption or registration-based encryption [CDG+17, DG17b, DGHM18, DGGM19, GHMR18, GHM+19, MW20], there is a significant shortcut to compress the public key PK. Instead of providing the public key PK in its entirety, only a short hash $H(PK)$ of PK is provided. This hash $H(PK)$ then allows the encrypter to *delegate* the computation of the ciphertexts $c_1, \ldots, c_\lambda$ to the decrypter in a secure way. As a matter of fact, looking ahead, our construction will rely on the same tools to compress the master public keys.

Challenges for extending to bk-IBE. To adapt this high level idea to the IBE setting, one encounters several bottlenecks, which we highlight below.

Firstly, recall that in the case of bk-PKE, the random selection of the set I, containing the indices of secret keys that will be accessed by the decryption, needs to be crucially made at the encryption time. This leads to a critical problem in the bk-IBE setting: since our target is to keep the identity secret keys (decryption keys) short, this information pertaining to selection of the identity keys must be fixed independent of the random coins of the encryption.

Secondly, one might think the above issue is no longer relevant if the challenge identity is picked randomly. For example, suppose every identity id implicitly defines some subset S_{id}, and its identity secret-key corresponds to $\{\mathsf{sk}_i : i \in S_{\mathsf{id}}\}$. Then, one might hope a similar argument will prove the security of a randomly-selected identity. However, recall that the adversary is given unbounded access to $KEYGEN$ in IBE schemes, through which he could potentially learn all the sk_i's, thus breaking the security. This challenge posed by an unbounded access to $KEYGEN$ queries does not exist in the bk-PKE schemes.

Thus, one might wonder if we could handle the $KEYGEN$ queries by starting with a leakage-resilient IBE scheme and amplifying the leakage tolerance on the master secret key through the above parallel repetition idea. For such an amplification, we must start with an IBE scheme that tolerates some bounded leakage, say m-bits, on the master secret key, and the only known prior scheme allowing that is [LRW11] (other schemes only tolerate bounded leakage on large identity secret keys, and not on the master key itself). The new scheme is obtained by generating ℓ independent instances of this underlying IBE scheme. Now, every identity id is associated with a subset $S_{\mathsf{id}} \subseteq [\ell]$ and its identity secret key is the identity secret keys for all the instances $i \in S_{\mathsf{id}}$. It is plausible to conjecture that a random identity is secure in this new scheme tolerating (approximately) $m \cdot \ell$-bit leakage. However, the only known techniques of proving such leakage amplification (using parallel repetition) are based on information-theoretic arguments [ADW09, ADN$^+$10, BK12, HLWW13]. In particular, the security proof requires that the ciphertext is indistinguishable from some simulated ciphertext, which contains information-theoretic entropy in the adversary's view.[2] However, no known leakage-resilience IBE supports such a proof structure (as no entropy is left, given all the unbounded identity queries), and hence the parallel repetition does not give an amplification. In fact, there are works (e.g., [LW10, JP11]) which show that in general, parallel repetition of a leakage-resilient encryption scheme *does not* amplify the leakage-resilience.

Our work precisely circumvents the problems listed above, and builds a leakage-resilient IBE scheme from scratch, such that it supports such an information-theoretic argument. In particular, we show that there is a way to simulate the entire view of the adversary including all the secret key queries such that (1) the adversary cannot distinguish the simulated view from the real view and (2) in this simulated view, the challenge ciphertext retains information-theoretic entropy, given the leakage. The key primitive that helps us achieve this is a *big-key pseudo-entropy function.*

Our Ideas. We construct our bk-IBE scheme by anchoring the leakage resilience properties from the corresponding properties of a simpler primitive, namely a big-key pseudo-entropy function. A pseudo-entropy function (PEF) [BHK11] has the property that its output at certain inputs are still *unpredictable*, even if the distinguisher has obtained leakage about the PEF key (in addition to the output of the PEF elsewhere). While ideally we would want to rely on pseudo-random functions (PRFs), they cannot even tolerate a single bit of leakage.

In this work, we will focus on the selective security notion, both for IBE and for PEFs. A pseudo-entropy function PEF is selectively secure for t inputs against ℓ bits of leakage, if for any inputs $x_1, \ldots, x_t$ it holds that $PEF(K, x_1), \ldots,\ PEF(K, x_t)$ is unpredictable given $L(K)$, where $L(\cdot)$ is an ℓ-bit leakage function. For our construction, we will need a *locally computable* PEF, i.e., $PEF(K, x)$ will access the key K only in a few locations.[3]

[2] Given such a proof structure, parallel repetition amplifies the total entropy of the simulated ciphertexts and, hence, naturally amplifies the leakage-resilience of the system as well.

[3] For technical reasons, we need that the locations in which K is queried do not depend on K itself. For this reason, our actual PEF construction relies on an additional common reference string.

Leakage Resilient Public-Key Encryption. Our big-key IBE scheme is conceptually built on the *weak hash proof system* framework of Hazay et al. [HLWW13]. This work constructs a leakage resilient key-encapsulation mechanism from any (non-leakage resilient) public key encryption scheme. The main ideas of their construction can roughly be summarized as follows. The public key PK of their scheme consists of $2n$ pairs $(\mathsf{pk}_{1,0}, \mathsf{pk}_{1,1}), \ldots, (\mathsf{pk}_{n,0}, \mathsf{pk}_{n,1})$ of public keys for an underlying public key encryption scheme. The secret key SK on the other hand, contains a random vector $b = (b_1, \ldots, b_n)$ and only contains one secret key sk_{i,b_i} for every index i. Key encapsulation proceeds as follows: To encapsulate a randomly chosen key $k = (k_1, \ldots, k_n) \leftarrow \{0,1\}^n$, compute ciphertexts $c_{i,0}$ and $c_{i,1}$ (for $i = 1, \ldots, n$), where $c_{i,0}$ encrypts k_i under $\mathsf{pk}_{i,0}$ and $c_{i,1}$ encrypts k_i under $\mathsf{pk}_{i,1}$. To decapsulate such a ciphertext, compute $k_i = \mathsf{Dec}(\mathsf{sk}_{i,b_i}, c_{i,b_i})$ for each index i.

Leakage resilience of this encapsulation mechanism is established as follows: Let $c = ((c_{1,0}, c_{1,1}), \ldots, (c_{1,n}, c_{1,n}))$ be a challenge ciphertext. In the real CPA experiment, both $c_{i,0}$ and $c_{i,1}$ encrypt the same bit k_i for all i. Since for each i the secret key corresponding to $pk_{i,1-b_i}$ is *not part* of the secret key SK, by relying on the IND-CPA security of the underlying encryption scheme we can switch each $c_{i,1-b_i}$ to encrypt $1 - k_i$ instead of k_i. Note that even an adversary in possession of SK would not notice this switch. Now, since the b_i are chosen uniformly at random, the encapsulated key depends on the entropy of b (which is part of the secret key). Specifically, decapsulating such a malformed ciphertext produces a key $k' = k \oplus b$. But this means that unless the adversary knows the vector b entirely, k' has entropy from the adversary's view. In other words, as long as the adversary's leakage is sufficiently shorter than n, the key encapsulated in such a malformed ciphertext will be unpredictable from the adversary's point of view. Establishing a uniform key follows via standard randomness extraction techniques in a post-processing step.

Towards Identity-Based Encryption. Alas, this idea does not translate directly to the setting of identity-based encryption. For each identity secret key $\mathsf{sk}_{\mathsf{id}}$ we would need to argue that some part of $\mathsf{sk}_{\mathsf{id}}$, similar to the vector b above, must retain entropy in the adversary's view, even given leakage about the *master secret key* msk. However, since msk is a compact representation of *all* identity secret keys, msk will be used to compute both $\mathsf{sk}_{\mathsf{id},i,0}$ and $\mathsf{sk}_{\mathsf{id},i,1}$ (to stay with the above notation). In other words, msk cannot just *forget* half of the secret keys for each identity.

Anchoring Leakage resilience in PEFs. Our approach is to adapt the [HLWW13] technique so as to push the entire entropy of the master secret key into the key K of a pseudo-entropy function. Furthermore, we will not rely on pairs of public keys $\mathsf{pk}_{i,0}, \mathsf{pk}_{i,1}$ as the construction of [HLWW13], but instead rely on a special type of witness encryption scheme [BL20] which allows us to use information relating to the PEF key K to decrypt. Looking ahead, for each identity id the role of the random vector b in the construction of [HLWW13] will be played by a function value $PEF(K, \mathsf{id})$. We first describe a version of our construction with non-succinct public parameters and later show how these can be compressed into succinct public parameters via a laconic OT-based non-interactive secure computation (NISC) [CDG$^+$17, DG17b, DG17a]. The master secret key msk of our scheme is simply the key K for a leakage resilient local big-key PEF. Assume that $K = (K_1, \ldots, K_n)$, where the K_i are "short" blocks of size $\mathsf{poly}(\lambda)$ (independent of the leakage bound ℓ).

The public parameters pp consist of commitments to the blocks K_i of K, as well as a common reference string crs for a special NIZK proof system. Both the commitment scheme and the NIZK proof system need to be compatible with the special witness encryption scheme of [BL20].

Identity secret keys in our scheme are generated as follows. First, the KeyGen algorithm computes $s_{\mathsf{id}} = PEF(K, \mathsf{id})$. Since PEF is local, this will only access a small number of the blocks K_i. Further recall that the indices of these blocks do not depend on K itself. The KeyGen algorithm now computes NIZK proofs Π_i, for each $i = 1, \cdots, \lambda$, corresponding to the statements x_i ="the i-th bit of $PEF(K, \mathsf{id})$ is $s_{\mathsf{id},i}$" (where K relates to the commitments in the public parameters pp). We stress that since $PEF(K, \mathsf{id})$ only accesses a small number of blocks of K, both the statements x_i and the proofs Π_i are succinct, i.e. independent of ℓ. The identity secret key $\mathsf{sk}_{\mathsf{id}}$ now consists of s_{id}, the statements x_i and the NIZK proofs Π_i.

We will now describe the encapsulation and decapsulation algorithms. For an identity id, we encapsulate a random key $u = (u_1, \ldots, u_\lambda) \leftarrow \{0,1\}^\lambda$ as follows: for each index i we compute two ciphertexts $c_{i,0}$ and $c_{i,1}$ using the special witness encryption scheme, both encrypting u_i. The statement under which we encrypt $c_{i,0}$ is $x_{i,0}$ ="the i-th bit of $PEF(K, \mathsf{id})$ is 0", whereas the corresponding statement for $c_{i,1}$ is $x_{i,1}$ ="the i-th bit of $PEF(K, \mathsf{id})$ is 1". The ciphertext C consists of $(c_{1,0}, c_{1,1}), \ldots, (c_{\lambda,0}, c_{\lambda,1})$. To decapsulate such a ciphertext C using an identity secret key $\mathsf{sk}_{\mathsf{id}}$, for each $i \in \{1, \ldots, \lambda\}$ we decrypt $c_{i,s_{\mathsf{id},i}}$ using Π_i as a witness. Correctness follows routinely from the correctness of the components.

Security. We will establish security roughly following the blueprint of [HLWW13]. Specifically, assume we have challenge identities $\mathsf{id}_1, \ldots, \mathsf{id}_t$ and challenge ciphertexts $C_1, \ldots, C_t$. Our first step of modification relies on the fact that, for each pair of ciphertexts $c_{i,0}, c_{i,1}$, one of the statements $x_{i,0}$ or $x_{i,1}$ must be false. Consequently, by the security of the witness encryption scheme we can flip one of the encrypted bits, effectively pushing entropy from $s_{\mathsf{id}} = PEF(K, \mathsf{id})$ into the corresponding challenge ciphertext.

In the second step, we use the simulation property of the NIZK to remove the dependence of the proofs Π_i's (in the identity secret keys) on the PEF key K. Likewise, we can replace the commitments in the public parameters with fake commitments, which are generated independently of the PEF key K.

Now observe that, the only part of the identity secret key that still depends on the key K is $PEF(K, \mathsf{id})$. To handle this, our PEF comes with a puncture mode, where, given a set of challenge identities $\mathsf{id}_1, \ldots, \mathsf{id}_t$, the PEF samples a punctured key $K^{\odot}$, such that: (A) it satisfies correctness for all non-challenge identities, i.e., $PEF(K, \mathsf{id}) = PEF(K^{\odot}, \mathsf{id})$ for all $\mathsf{id} \notin \{\in \mathsf{id}_1, \ldots, \mathsf{id}_t\}$. This ensures that we can answer all KeyGen queries using $K^{\odot}$; (B) the PEF outputs $(PEF(K, \mathsf{id}_1), \ldots, PEF(K, \mathsf{id}_t))$ contain "high-enough" entropy, given $K^{\odot}$. This property ensures that the challenge ciphertexts are unpredictable, given the adversary's view (which now does not depend on K, but only on $K^{\odot}$).

Finally, we reduce the selective security to the security of the underlying PEF. The above arguments help us to push all the entropy of the $PEF(K, \mathsf{id}_i)$ to the corresponding challenge ciphertexts. Hence, we now invoke the selective leakage resilience of the PEF to information-theoretically show that for some identity id_i the adversary cannot have a non-trivial advantage in distinguishing the corresponding challenge ciphertext.

1.4 Future Directions

Our work leaves open several exciting problems. We discuss a few of them below.

As in IBE schemes, there are two flavors of security one could imagine, namely, selective and adaptive/full security. In this work, we achieve selective security, where the adversary must select the challenge set, $\mathcal{J}$, of $\ell + 1$ identities before the setup of the system, and succeeds only if she breaks all the identities in $\mathcal{J}$. In contrast, full security allows the adversary to adaptively pick this set, i.e., she succeeds as long as she breaks the security of all the identities in any set $\mathcal{J}$ of size $\ell+1$. We leave the problem of building a fully secure big-key IBE as a fascinating open problem.

Secondly, having initiated the study of big-key IBE, the next natural step towards making it truly practical would be to build it with only black-box use of the underlying primitives. Another practically useful feature to add to our big-key IBE would be to incorporate the updatability of the keys.

The third interesting problem that we leave open stems from the recent technique [MW20] of making the secret keys "catalytic", i.e., the large secret key is no longer needed to be a completely random string (which the user doesn't utilize elsewhere), but is generated as a (randomized) encoding of some public data (e.g., music library) that cannot be compressed further by the adversary. Extending the study of such public-key encryption schemes with catalytic keys to our big-key IBE setup would be another exciting problem to explore.

Finally, we note that typically, in IBE security definitions, the adversary is given access to a KeyGen oracle, which outputs identity secret keys. The only restriction is that the adversary cannot query the challenge identity id^*. In our security definition, we do not allow the adversary to query KeyGen on any identity in the set of challenge identities $\mathcal{J}$ accordingly. While such a restriction seems natural, one may wonder whether it is necessary. Consider a relaxation of this assumption where the adversary is allowed to make KeyGen queries with keys in the set $\mathcal{J}$ of challenge identities, which are subsequently removed from $\mathcal{J}$. We claim that any scheme with a deterministic KeyGen algorithm (as is the case for most IBE constructions) would be immediately insecure. The reason is that the leakage function L may leak a succinct *parity information* about the keys of the challenge identities, e.g. $\mathsf{leak} = \bigoplus_{\mathsf{id} \in \mathcal{J}} \mathsf{sk}_{\mathsf{id}}$. Given this leakage leak, the adversary could query the KeyGen oracle on all but one of the identities in $\mathcal{J}$, say id^*, and then reconstruct $\mathsf{sk}_{\mathsf{id}^*}$ via $\mathsf{sk}_{\mathsf{id}^*} = \mathsf{leak} \oplus \bigoplus_{\mathsf{id} \in \mathcal{J} \setminus \{\mathsf{id}^*\}} \mathsf{sk}_{\mathsf{id}}$. As the question of achieving such a stronger security notion by relying on additional randomization of the KeyGen procedure seems quite challenging and is beyond the scope of this work, we leave it open for future work.

2 Preliminaries

Notations. We use λ to denote the security parameter. $\mathsf{negl}(\cdot)$ denotes a negligible function. For $n \in \mathbb{Z}$, $[n]$ denotes the set $[n] = \{1, \cdots, n\}$. For a distribution X, we use $x \leftarrow X$ to denote the process of sampling x from X. For a set $\mathcal{X}$, we use $x \leftarrow \mathcal{X}$ to denote sampling x from $\mathcal{X}$ uniformly at random. We also use $U_{\mathcal{X}}$ for the uniform distribution over $\mathcal{X}$. We define statistical difference as $\Delta(X; Y) = 1/2 \sum_a |\Pr[X = a] - \Pr[Y = a]|$, and say that X and Y are statistically close if their statistical difference is bounded by a negligible function of the security parameter. We say that X and Y are computationally indistinguishable if for any PPT adversary D, $|\Pr[D(X) = 1] - \Pr[D(Y) = 1]| \leq \mathsf{negl}(\lambda)$.

2.1 Min-Entropy

Let X be a random variable supported on a finite set $\mathcal{X}$ and let $Z \in \mathcal{Z}$ be another random variable (possibly correlated with X). The min-entropy of X is defined as $H_\infty(X) = -\log(\max_x \Pr[X = x])$.

The *average conditional min-entropy* [DORS08] of X given Z is defined by

$$\tilde{H}_\infty(X|Z) = -\log\left(\mathsf{E}_{z\sim Z}\left[\max_{x\in\mathcal{X}} \Pr[X = x|Z = z]\right]\right).$$

We use the following weak chain rule about average conditional min-entropy.

Lemma 1 (Weak Min-Entropy Chain Rule [DORS08])**.** *Let $X \in \mathcal{X}$ and $Z \in \mathcal{Z}$ be random variables. Then it holds that*

$$\tilde{H}_\infty(X|Z) \geq H_\infty(X) - \log(|\mathcal{Z}|).$$

Additionally, for any $\delta > 0$, with probability at least $1 - \delta$ over $z \leftarrow Z$, we have

$$H_\infty(X|Z = z) \geq \tilde{H}_\infty(X|Z) - \log(1/\delta).$$

Further, our proof requires the following min-entropy splitting lemma, the proof of which, essentially follows from recursively invoking [DFR$^+$07, Lemma 4.2].

Lemma 2 (Min-Entropy Splitting Lemma)**.** *Let $X_1, \ldots, X_\kappa$ be a sequence of random variables such that $H_\infty(X_1, \ldots, X_\kappa) \geq \alpha$. There exists a random variable C over $[\kappa]$ s.t.*

$$H_\infty(X_C|C) \geq \alpha/\kappa - \log\kappa.$$

Proof. We use the following min-entropy splitting lemma from [DFR$^+$07]. Lemma 4.2 of [DFR$^+$07] proves the general statement for *ε-smooth min-entropy.* Below, we use the special case of that lemma with $\varepsilon = 0$, i.e., min-entropy.

Lemma 3. *[DFR$^+$07, Lemma 4.2] Let $\varepsilon \geq 0$, and let X_0, X_1 be random variables with $H_\infty(X_0, X_1) \geq \alpha$. Then, there exists a binary random variable C over $\{0, 1\}$ such that $H_\infty(X_{1-C}, C) \geq \alpha/2$.*

To prove our lemma, we recursively invoke Lemma 3 to bi-partition the coordinates for $\log\kappa$ times. This gives us that

$$H_\infty(X_C, C) \geq \alpha/\kappa,$$

where C is some random variable over $\{1, 2, \ldots, \kappa\}$. Now, by Lemma 1,

$$\tilde{H}_\infty(X_C|C) \geq \alpha/\kappa - \log\kappa.$$

□

Our construction also relies on a randomness extractor, which we recall below.

Definition 1 (Randomness Extractor)**.** *A function* $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is called a* (k, ε)-strong randomness extractor *if, for all distributions X over $\{0,1\}^n$ such that $H_\infty(X) \geq k$, we have*

$$\Delta\Big(\ (s, \mathsf{Ext}(X, s))\ ;\ (U_{\{0,1\}^d}, U_{\{0,1\}^m})\ \Big) \leq \varepsilon,$$

where the seed s is chosen uniformly at random from $\{0, 1\}^d$.

3 Puncturable Local Pseudo-Entropy Functions

In this Section we will provide definitions and construction of local pseudo-entropy functions. Our target security notion is selective security, i.e., before receiving leakage and getting access to the function, the adversary has to announce his challenge inputs.

Definition 2. *Given a parameter ℓ, a puncturable local pseudo-entropy function is specified by a pair of PPT algorithms* $(\mathsf{Gen}, \mathsf{PEF})$ *with the following syntax.*

- $\mathsf{Gen}(1^\lambda, \ell)$*: Outputs a pair* (CRS, K)*, where* CRS *is a common reference string of size* $\mathsf{poly}(\lambda)$*, and* $K = (K_1, \ldots, K_n)$ *is a key consisting of* $K_i \in \{0,1\}^{\mathsf{poly}(\lambda)}$*.*[4]
- $\mathsf{PEF}(\mathsf{CRS}, K, x)$*: Takes as input* CRS *and* x *and gets RAM access to* K*, and outputs a* $Y \in \{0,1\}^{\mathsf{poly}(\lambda)}$*.*

We also require the existence of $(\mathsf{Gen}_2, \mathsf{PEF}_2)$ *with the following syntax.*

- $\mathsf{Gen}_2(1^\lambda, \ell, x_1, \ldots, x_\kappa)$*: Outputs a tuple* $(\mathsf{CRS}, K, K^{\odot})$*.*
- $\mathsf{PEF}_2(\mathsf{CRS}, K^{\odot}, x)$*: Takes as input* CRS*,* $K^{\odot}$*,* x*, and outputs a* Y*.*

We require the following properties to hold.

- *Locality:* $\mathsf{PEF}(\mathsf{CRS}, K, \cdot)$ *makes at most* $\mathsf{poly}(\lambda)$ *(independent of* ℓ*) RAM access to* $K = (K_1, \ldots, K_n)$*.*
- *Mode-Indistinguishability: Fix* $x_1, \ldots, x_\kappa \in \{0,1\}^\lambda$ *and let* $(\mathsf{CRS}', K', K^{\odot}) \leftarrow \mathsf{Gen}_2(1^\lambda, \ell, x_1, \ldots, x_\kappa)$*. Then* (CRS', K') *is computationally indistinguishable from* $(\mathsf{CRS}, K) \leftarrow \mathsf{Gen}(1^\lambda, \ell)$*.*
- *Punctured correctness: Fix* $x_1, \ldots, x_\kappa \in \{0,1\}^\lambda$ *and let* $(\mathsf{CRS}, K, K^{\odot}) \leftarrow \mathsf{Gen}_2(1^\lambda, \ell, x_1, \ldots, x_\kappa)$*. Then it holds for all* $x \notin \{x_1, \ldots, x_\kappa\}$ *that* $\mathsf{PEF}(\mathsf{CRS}, K, x) = \mathsf{PEF}_2(\mathsf{CRS}, K^{\odot}, x)$*, except with negligible probability over the coins of* Gen_2*.*
- *k-Selective* β*-Pseudo-Entropy Security: Fix* $x_1, \ldots, x_\kappa \in \{0,1\}^\lambda$ *and let* $(\mathsf{CRS}, K, K^{\odot}) \leftarrow \mathsf{Gen}_2(1^\lambda, \ell, x_1, \ldots, x_\kappa)$*. Then it holds that*

$$\tilde{H}_\infty\Big(\mathsf{PEF}(\mathsf{CRS}, K, x_1), \ldots, \mathsf{PEF}(\mathsf{CRS}, K, x_\kappa) \;\Big|\; \mathsf{CRS}, K^{\odot}\Big) \geq \beta.$$

  Observe that, by the punctured correctness, one could use $K^{\odot}$ *to correctly evaluate* PEF *at all inputs* $x \notin \{x_1, \ldots, x_\kappa\}$*. Therefore, this property implicitly states that, even if the adversary obtains* $\mathsf{PEF}(x)$ *at all inputs* $x \notin \{x_1, \ldots, x_\kappa\}$*,* $\mathsf{PEF}(\mathsf{CRS}, K, x_1), \ldots, \mathsf{PEF}(\mathsf{CRS}, K, x_\kappa)$ *is still (information -theoretically) unpredictable.*

The notion of pseudo-entropy functions is first proposed by Braverman, Hassidim, and Kalai [BHK11]. Their definition supports puncturing at one input and does not require locality. Let us recall their result.[5]

[4] The length of CRS and every K_i do not depend on ℓ, but n shall depend on ℓ.

[5] Their work predates the first mention of punctured PRFs [BGI14]. While they do not use puncturing formalism, they implicitly define a punctured generation and evaluation algorithm in their proof.

Theorem 1 ([BHK11] Thm. 4.1)**.** *Let $\delta > 0$ be an arbitrary constant. Under the decisional Diffie Hellman assumption, there exists a family of 1-selective $\gamma = (1-\delta)\alpha$-pseudo-entropy functions, where α is the length of the secret key.*

In other words, [BHK11] constructed a PEF such that, after puncturing at one input x, $\mathsf{PEF}(\mathsf{CRS}, K, x)$ preserves almost the entire entropy of the key K.

Remark 1. *We make a few remarks about our definition.*

- ***Leakage-resilience.*** *The leakage-resilience of the PEF simply follows from the min-entropy chain rule (Lemma 1). That is, given an m-bit leakage $L(K)$ of the key K, the entropy guarantee in the definition*

$$\tilde{H}_\infty\Big(\mathsf{PEF}(\mathsf{CRS}, K, x_1), \ldots, \mathsf{PEF}(\mathsf{CRS}, K, x_\kappa) \;\Big|\; \mathsf{CRS}, K^{\odot}\Big) \geq \gamma$$

  implies

$$\tilde{H}_\infty\Big(\mathsf{PEF}(\mathsf{CRS}, K, x_1), \ldots, \mathsf{PEF}(\mathsf{CRS}, K, x_\kappa) \;\Big|\; \mathsf{CRS}, K^{\odot}, L(K)\Big) \geq \gamma - m.$$

  Braverman et. al. [BHK11] choose to incorporate the leakage resilience in their definition. Here, our definition simply states the min-entropy guarantee, and we shall handle the leakage within corresponding proofs directly.

- ***Parameters Setting.*** *Looking ahead, we shall use our PEF to construct our big-key IBE scheme. The big-key scheme first specifies a leakage parameter ℓ that it aims to achieve, which, in turn, determines the number κ of inputs our PEF needs to puncture in order to obtain sufficiently high (e.g., $\geq \ell$) min-entropy guarantee. Finally, the number of inputs to be punctured determines the number n of blocks we need to have in the key $K = (K_1, \ldots, K_n)$.*

- ***CRS.*** *We note that our definition includes a CRS. Intuitively, the locations in K that one needs to access in order to evaluate $\mathsf{PEF}(\mathsf{CRS}, K, x)$ must be fixed and public, given the CRS and x. As it will become clear in our big-key IBE construction, this ensures that the encryption algorithm is also local (i.e., independent of ℓ). We shall elaborate more on this later.*

  Finally, we note that the construction of [BHK11] does not have a CRS. Hence, we omit the CRS when we use their PEF as an underlying building block.

Finally, the following simple Lemma about random bipartite graphs shall be useful to us, whose proof follows by a simple probabilistic argument.

Lemma 4. *Let $N, M > 0$ be integers with $N \leq (1-\varepsilon)M$ for a constant $\varepsilon > 0$ and $d > 0$ be an integer. Let $L = [N]$ and $R = [M]$. Let $\Gamma \subseteq L \times R$ be a random graph which is chosen as follows: For every vertex $v \in L$ the neighborhood $\Gamma(v)$ is sampled by choosing $w_1, \ldots, w_d \leftarrow R$ uniformly at random and setting $\Gamma(v) = \{w_1, \ldots, w_d\}$. Let MATCH be the event that every vertex $v \in L$ can be matched with a unique vertext $w \in R$, i.e. for each $v \in L$ there exists a $W(v) \in \Gamma(v)$ such that for $v \neq v'$ it holds that $W(v) \neq W(v')$. Then it holds that*

$$\Pr[\mathsf{MATCH}] \geq 1 - N \cdot (1-\varepsilon)^d \geq 1 - N \cdot e^{-\varepsilon \cdot d}.$$

Furthermore, such matching can be found efficiently, except with probability $N \cdot (1-\varepsilon)^d$

Proof. Consider the following process. Initialize a set $R_{\mathsf{good},0} = R$. For $i = 1, \ldots, N$ do the following: Choose a $w_i \in \Gamma(i) \cap R_{\mathsf{good},i-1}$ and set $R_{\mathsf{good},i} \leftarrow R_{\mathsf{good},i-1} \backslash \{w_i\}$. If $\Gamma(i) \cap R_{\mathsf{good},i-1} = \emptyset$ abort and output $\perp$.

We will now show that the probability of failure of this procedure is at most $N \cdot (1 - \varepsilon)^d$, establishing the statement of the lemma. For each $i \in L$, let BAD_i be the event that $\Gamma(i) \cap R_{\mathsf{good},i-1} = \emptyset$. Let BAD be the event that any of the BAD_i occurs. At step i, if no abort occurred the size of $R_{\mathsf{good},i-1}$ is $M - i + 1$. Thus, BAD_i occurs if and only if all d neighbors of i end up in $R \backslash R_{\mathsf{good},i-1}$, the probability of which is

$$\Pr[\mathsf{BAD}_i] = \left(\frac{i-1}{M}\right)^d.$$

By a union-bound, it holds that

$$\Pr[\mathsf{BAD}] = \Pr[\mathsf{BAD}_1 \vee \cdots \vee \mathsf{BAD}_N] \leq N \cdot ((1-\varepsilon)M/M)^d = N \cdot (1-\varepsilon)^d.$$

□

Note that the failure probability in Lemma 4 is negligible for $N = \mathsf{poly}(\lambda)$ and $\varepsilon \cdot d \geq \omega(\log(\lambda))$.

3.1 Our Construction

We will now provide our construction of a local pseudo-entropy function. Our construction will start from the PEF construction of [BHK11] which is not local, and amplify this to a PEF which can be evaluated by a local algorithm.

Let $(\mathsf{Gen}', \mathsf{PEF}')$ be the family of pseudo-entropy functions (without local evaluation) from Theorem 1, and let PRF be a pseudorandom function which takes as input an $x \in \{0,1\}^\lambda$ and outputs a sequence of elements $(i_1, \ldots, i_d) \in [\ell]^d$.

$\mathsf{Gen}(1^\lambda, \ell)$: For $i = 1, \ldots, n$, compute $K_i \leftarrow \mathsf{Gen}'(1^\lambda)$ and choose $K^* \leftarrow \{0,1\}^\lambda$. Output $\mathsf{CRS} = K^*$ and $K = (K_1, \ldots, K_n)$.

$\mathsf{PEF}(\mathsf{CRS}, K, x)$:

- Parse $\mathsf{CRS} = K^*$
- Compute $(i_1, \ldots, i_d) \leftarrow \mathsf{PRF}(K^*, x)$
- Retrieve $K_{i_1}, \ldots, K_{i_d}$ via oracle access to K
- Compute and output $Y \leftarrow (\mathsf{PEF}'(K_{i_1}, x), \ldots, \mathsf{PEF}'(K_{i_d}, x))$

First note that PEF is local, as it only accesses K at $d = \mathsf{poly}(\lambda)$ locations $i_1, \ldots, i_d$. Moreover, the location it accesses is fixed by CRS and x.

Selective Security. We will first provide the punctured key generation and evaluation algorithms. Let $\mathsf{Gen}_2'(1^\lambda, \cdot)$ and $\mathsf{PEF}_2'(1^\lambda, \cdot)$ be the punctured key generation and evaluation algorithms for $(\mathsf{Gen}', \mathsf{PEF}')$.

- $\mathsf{Gen}_2(1^\lambda, \ell, x_1, \ldots, x_\kappa)$: Generate the key PRF key $K^* \leftarrow \{0,1\}^\lambda$ and set $\mathsf{CRS} = K^*$. Let **MATCH** be the event that for every index $i \in [\kappa]$ that there is an index j_i such that j_i appears in the list generated by $\mathsf{PRF}(K^*, x_i)$, but j_i appears in no other list generated by $\mathsf{PRF}(K^*, x_{i'})$ for $i' \neq i$. If the event holds, compute such a matching. For each $i = 1, \ldots, \kappa$, compute $(K_{j_i}, K^{\odot}_{j_i}) \leftarrow \mathsf{Gen}'_2(1^\lambda, x_i)$. For all remaining indices $i \in [n] \setminus \{j_1, \ldots, j_\kappa\}$, compute K_i via $K_i \leftarrow \mathsf{Gen}'(1^\lambda)$ and set $K^{\odot}_i = K_i$. Set $K = (K_1, \ldots, K_n)$, $K^{\odot} = (K^{\odot}_1, \ldots, K^{\odot}_n)$ and output $(\mathsf{CRS}, K, K^{\odot})$.

- $\mathsf{PEF}_2(\mathsf{CRS}, K^{\odot}, x)$:
  - Parse $\mathsf{CRS} = K^*$
  - Compute $(i_1, \ldots, i_d) \leftarrow \mathsf{PRF}(K^*, x)$
  - Compute and output $Y \leftarrow (\mathsf{PEF}'(K^{\odot}_{i_1}, x), \ldots, \mathsf{PEF}'(K^{\odot}_{i_d}, x))$.

Theorem 2. *Let $\delta > 0$ be a constant, let $\kappa = (1-\delta)n$ and let $\gamma = \mathsf{poly}(\lambda)$. Assume that $(\mathsf{Gen}', \mathsf{PEF}')$ is a family of 1-selective γ-pseudo-entropy functions and PRF is a pseudo-random function. Then $(\mathsf{Gen}, \mathsf{PEF})$ has punctured correctness, and satisfies the mode-indistinguishability and κ-selective $(\kappa \cdot \gamma)$-pseudo-entropy properties.*

Remark 2. *Before we prove this theorem, we stress that $\kappa \cdot \gamma$ can get arbitrary close to the entropy of the PEF key K. Observe that the key $K = (K_1, \ldots, K_n)$ supports puncturing κ inputs, which is nearly n since $\kappa = (1 - \delta)n$. Additionally, for every input x_i, the γ entropy of $\mathsf{PEF}(\mathsf{crs}, K, x_i)$ is nearly the entire entropy of some block K_{j_i} (by Theorem 1). Overall, the entropy of $(\mathsf{PEF}(\mathsf{crs}, K, x_1), \ldots, \mathsf{PEF}(\mathsf{crs}, K, x_\kappa))$ is nearly the entire entropy of the key K. In other words, for an adversary who may leak almost the entire key K, $(\mathsf{PEF}(\mathsf{crs}, K, x_1), \ldots, \mathsf{PEF}(\mathsf{crs}, K, x_\kappa))$ still contains unpredictability.*

Proof. The mode-indistinguishability property of $(\mathsf{Gen}, \mathsf{PEF})$ follows routinely from the mode indistingiushability property of $(\mathsf{Gen}', \mathsf{PEF}')$.

To see the pseudo-entropy security, observe that

$$
\begin{aligned}
&\tilde{H}_\infty\big(\mathsf{PEF}(\mathsf{CRS}, K, x_1), \ldots, \mathsf{PEF}(\mathsf{CRS}, K, x_\kappa) \bigm| \mathsf{CRS}, K^{\odot}\big) \\
&\geq \tilde{H}_\infty\big(\mathsf{PEF}'(K_{j_1}, x_1), \ldots, \mathsf{PEF}'(K_{j_k}, x_\kappa) \bigm| \mathsf{CRS}, K^{\odot}\big) && \text{(where } j_i \text{ is the perfect match for } i\text{)} \\
&= \sum_{i=1}^{\kappa} \tilde{H}_\infty\big(\mathsf{PEF}'(K_{j_i}, x_i) \bigm| K^{\odot}_{j_i}\big) && \text{(Since every } (K_{j_i}, K^{\odot}_{j_i}) \text{ is independent)} \\
&\geq \kappa \cdot \gamma. && \text{(by the pseudo-entropy property of } \mathsf{PEF}'\text{)}
\end{aligned}
$$

To show that $(\mathsf{Gen}, \mathsf{PEF})$ has punctured correctness, it suffices to show that the event $\neg\mathsf{MATCH}$ happens at most with negligible probability. Observe that we can efficiently determine whether MATCH happens. Assume towards contradiction to $\neg\mathsf{MATCH}$ happens with non-negligible probability ε. We will construct a PPT distinguisher $\mathcal{D}$ which distinguishes $\mathsf{PRF}(K^*, \cdot)$ from a uniformly random function with advantage $\varepsilon - \mathsf{negl}$. Given $x_1, \ldots, x_\kappa$ and oracle access to a function $\mathcal{O}(\cdot)$, the distinguisher checks whether the event $\neg\mathsf{MATCH}$ happens for $\mathcal{O}$, i.e. if for every index $i \in [\kappa]$ that there is an index j_i such that j_i appears in the list generated by $\mathcal{O}(x_i)$, but j_i appears in no other list generated by $\mathcal{O}(x_{i'})$ for $i' \neq i$. If $\neg\mathsf{MATCH}$ happens $\mathcal{D}$ outputs 1, otherwise a uniformly random bit $b \leftarrow \{0,1\}$. Observe that

$$\begin{aligned}\Pr[\mathcal{D}^{\mathcal{O}} = 1] &= \underbrace{\Pr[\mathcal{D}^{\mathcal{O}(\cdot)} = 1|\mathsf{MATCH}]}_{=\frac{1}{2}} \cdot \Pr[\mathsf{MATCH}] + \underbrace{\Pr[\mathcal{D}^{\mathcal{O}(\cdot)} = 1|\neg\mathsf{MATCH}]}_{=1} \cdot \Pr[\neg\mathsf{MATCH}] \\ &= \frac{1}{2}(1 - \Pr[\neg\mathsf{MATCH}]) + 1 \cdot \Pr[\neg\mathsf{MATCH}] \\ &= \frac{1}{2} - \frac{\Pr[\neg\mathsf{MATCH}]}{2}.\end{aligned}$$

Now, if $\mathcal{O}$ implements a $\mathsf{PRF}(K^*, \cdot)$, then it holds that $\Pr[\neg\mathsf{MATCH}] > \varepsilon$. On the other hand, if $\mathcal{O}$ implements a uniformly random function H, by Lemma 4, it holds that $\Pr[\neg\mathsf{MATCH}] < \mathsf{negl}$. It follows that the distinguishing advantage of $\mathcal{D}$ is at least $\varepsilon/2 - \mathsf{negl}$, which is non-negligible. This contradicts the pseudorandomness of PRF. □

4 Big-Key Identity-Based Key Encapsulation Mechanism

In this section, we define and build a big-key identity-based key encapsulation mechanism (IB-KEM). This construction of IB-KEM will have a large public parameter. In supporting material Section 5, we will show how to generically transform it into an IBE scheme with a short public parameter.

4.1 Definition

Syntactically, a big-key identity-based key encapsulation mechanism consists of the following efficient algorithms. All algorithms (except for Setup) implicitly take the public parameter pp as input. We omit it to avoid cluttering.

- $(\mathsf{pp}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$: This algorithm takes the security parameter as input, and samples the public parameter pp and a master secret-key msk.
- $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$: This algorithm takes the master secret-key msk and the identity id as inputs, and samples an identity secret-key $\mathsf{sk}_{\mathsf{id}}$. In particular, KeyGen has RAM access to msk.[6]
- $(\mathsf{ct}, u) \leftarrow \mathsf{Encap}(\mathsf{id})$: This algorithm takes the identity id as input, and samples a ciphertext ct and its associated encapsulated key u.
- $u = \mathsf{Dec}(\mathsf{id}, \mathsf{ct}, \mathsf{sk}_{\mathsf{id}})$: This algorithm takes the identity id, the ciphertext ct, and the identity secret-key $\mathsf{sk}_{\mathsf{id}}$ as inputs, and output a decapsulated key u.

Definition 3 (Selective Secure IB-KEM)**.** *We say that an IB-KEM* (Setup*,* KeyGen, Encap, Dec) *is selectively secure under bounded leakage if it satisfies the following correctness, efficiency and security properties.*

[6] The length of the master secret-key msk depends on the leakage parameter, ℓ, and hence is long. However, the running time of KeyGen will be independent of ℓ. That is, it will only read a few bits of msk to create the short identity secret-key.

- ***Correctness.*** *For any identity* id*, it holds that*

$$\Pr\left[\begin{matrix}(\mathsf{pp},\mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda), & (\mathsf{ct},u) \leftarrow \mathsf{Encap}(\mathsf{id}) \\ \mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(\mathsf{msk},\mathsf{id}), & u' = \mathsf{Dec}(\mathsf{id},\mathsf{ct},\mathsf{sk}_{\mathsf{id}})\end{matrix} \ : \ u = u'\right] = 1.$$

- ***Efficiency.*** *The running time of* KeyGen*,* Encap*, and* Dec *are independent of the leakage parameter* ℓ*. This implicitly mandates that the identity secret-key* $\mathsf{sk}_{\mathsf{id}}$ *is succinct (i.e., its length is independent of* ℓ*). Additionally, the length of the public parameter* pp *is also required to be succinct.*[7]

- ***Selective Security under Bounded Leakage.*** *Fix an* $\ell > 0$*. We say that an IB-KEM* $(\mathsf{Setup},\mathsf{KeyGen},\mathsf{Encap},\mathsf{Dec})$ *is selectively secure, if for all PPT adversaries* $\mathcal{A} = (\mathcal{A}_1,\mathcal{A}_2,\mathcal{A}_3)$*, for all non-negligible* ε*, it holds that*

$$\Pr_{(\mathsf{msk},\mathsf{pp},\mathcal{J},\mathsf{state},\mathsf{leak})}\left[\forall\ \mathsf{id} \in \mathcal{J}, \mathsf{Adv}^{\mathsf{id}}(\mathsf{msk},\mathsf{pp},\mathsf{state},\mathsf{leak}) \geq \varepsilon\right] = \mathsf{negl}(\lambda),$$

where $(\mathsf{msk},\mathsf{pp},\mathcal{J},\mathsf{state},\mathsf{leak})$ *are sampled from the Phase I of* $\mathrm{IND}^{\mathsf{blsKEM}}(1^\lambda)$ *(refer to Figure 1) and the random variable* $\mathsf{Adv}^{\mathsf{id}}(\mathsf{msk},\mathsf{pp},\mathsf{state},\mathsf{leak})$ *is defined as follows.*

$$\mathsf{Adv}^{\mathsf{id}}(\mathsf{msk},\mathsf{pp},\mathsf{state},\mathsf{leak}) = \left|\Pr[\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk},\mathsf{pp},\mathsf{state},\mathsf{leak}) = 1] - \frac{1}{2}\right|$$

Here, the random variable $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk},\mathsf{pp},\mathsf{state},\mathsf{leak})$ *is as defined in Phase II, and* $\mathcal{A}_3$ *is not allowed to query the* KeyGen *on* $\mathcal{J}$*.*

$\mathrm{IND}^{\mathsf{blsKEM}}(1^\lambda)$:

- ***Phase I.*** *The system is set up as follows.*
  1. *Let* $(\mathcal{J},\mathsf{state}) \leftarrow \mathcal{A}_1(1^\lambda)$*, where* $\mathcal{J}$ *is a set of identities of size* $\ell + 1$*.*
  2. $(\mathsf{msk},\mathsf{pp}) \leftarrow \mathsf{Setup}(1^\lambda)$*.*
  3. $f \leftarrow \mathcal{A}_2(\mathsf{state},\mathsf{pp})$*, where the output length of* f *is (at most)* ℓ*. Let* $\mathsf{leak} := f(\mathsf{msk})$*.*
- ***Phase II.*** *For any* $\mathsf{id} \in \mathcal{J}$*, we define a security game* $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk},\mathsf{pp},\mathsf{state},\ \mathsf{leak})$ *as follows*
  1. $(\mathsf{ct},u) \leftarrow \mathsf{Encap}(\mathsf{id})$*.*
  2. *Let* u' *be an independent random string.*
  3. *Sample* $b \leftarrow \{0,1\}$*.*
  4. *If* $b = 0$*, let* $b' = \mathcal{A}_3^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{state},\mathsf{leak},\mathsf{pp},\mathsf{id},\mathsf{ct},u)$*;*
  5. *If* $b = 1$*, let* $b' = \mathcal{A}_3^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{state},\mathsf{leak},\mathsf{pp},\mathsf{id},\mathsf{ct},u')$*;*
  6. *Output 1 if* $b = b'$*; otherwise, output 0.*

Figure 1: Selective security experiment for IB-KEMs

[7] The running time of Setup and the length of the master secret-key msk, however, will inevitably depend on the leakage parameter ℓ.

Remark 3. *Note that, in the above definition, the adversary $\mathcal{A}_2$ does not get access to the* KeyGen *oracle. This is not restrictive since the leakage function f gets access to the entire secret key* msk*. Hence, any leakage function f with access to* KeyGen *oracle can be transformed into a leakage function f' that does not have access to* KeyGen *oracle.*

4.2 Witness Encryption for NIZK of Commitment Scheme

As a crucial building block for our IBE scheme, we shall use a witness encryption scheme for NIZK of commitment scheme. This was recently introduced and constructed by Benhamouda and Lin [BL20]. Let us start with the definition.

Definition 4 ([BL20])**.** *A* witness encryption for NIZK of commitment scheme *that supports a circuit class $\mathcal{G}$ consists of the following efficient algorithms.*

- ***CRS Setup:*** $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$ *on input the security parameter λ, generates a CRS* crs.
- ***Commitment:*** $c \leftarrow \mathsf{Com}(\mathsf{crs}, x; r)$ *on input the CRS* crs *and a message x, generates a commitment c. The decommitment is the message x and the private randomness r.*
- ***Language $\mathcal{L}$:*** *A language $\mathcal{L}$ is defined by the CRS* crs *as follows. A statement* $\mathsf{st} = (c, G, y)$*, where c is a commitment and $G \in \mathcal{G}$ is a circuit, is in the language $\mathcal{L}$ with witness (x, r) if it holds that (1)* $c = \mathsf{Com}(\mathsf{crs}, x; r)$*; (2) $G(x) = y$.*
- ***NIZK Proof:*** $\pi \leftarrow \mathsf{Prove}(\mathsf{crs}, c, G, (x, r))$ *on input the CRS* crs*, a commitment c, a circuit $G \in \mathcal{G}$, and a decommitment (x, r), generates a proof π proving the statement $(c, G, G(x)) \in \mathcal{L}$ with witness (x, r).*
- ***Witness Encryption:*** $\mathsf{ct} \leftarrow \mathsf{WEnc}(\mathsf{crs}, \mathsf{msg}, (c, G, y))$ *on input the CRS* crs*, a message* msg*, and a statement (c, G, y), generates a ciphertext* ct.
- ***Witness Decryption:*** $\mathsf{msg} = \mathsf{WDec}(\mathsf{crs}, \mathsf{ct}, (c, G, y), \pi)$ *on input the CRS* crs*, a ciphertext* ct*, a statement (c, G, y), and a NIZK proof π, computes a message* msg.
- ***Simulated CRS:*** $(\mathsf{crs}, \tau) \leftarrow \mathsf{SimSetup}(1^\lambda)$ *on input the security parameter λ, generates a simulation CRS* crs *and its associated trapdoor τ.*
- ***Simulated Commitment:*** $(c, \mathsf{aux}) \leftarrow \mathsf{SimCom}(\mathsf{crs})$*, on input the CRS* crs*, generates a simulated commitment c with its auxiliary information* aux.
- ***Simulated Decommit:*** $r = \mathsf{SimDecom}(\mathsf{crs}, \tau, c, \mathsf{aux}, x)$*, on input the simulated CRS* crs *and its associated trapdoor τ, the simulated commitment c and its associated auxilliary information* aux*, and any message x, generates a decommitment r such that (x, r) is a valid decommitment of c with* crs.
- ***Simulated Proof:*** $\pi \leftarrow \mathsf{SimProve}((\mathsf{crs}, \tau, \mathsf{aux}), (c, G, G(x)))$ *on input the simulated CRS* crs*, its associated trapdoor τ, the auxiliary information* aux *for the commitment c, and finally a statement $(c, G, G(x))$, generates a simulated proof π proving the statement $(c, G, G(x))$.*

This set of algorithms satisfy the following guarantees.

- ***Perfect Correctness.*** *For all input x, circuit $G \in \mathcal{G}$, and message* msg*, it holds that*

$$\Pr\left[\begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda),\ c = \mathsf{Com}(\mathsf{crs}, x; r) \\ \mathsf{ct} \leftarrow \mathsf{WEnc}(\mathsf{crs}, \mathsf{msg}, (c, G, G(x))) \\ \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, c, G, (x, r)) \\ \mathsf{msg}' = \mathsf{WDec}(\mathsf{crs}, \mathsf{ct}, (c, G, G(x)), \pi) \end{array} : \mathsf{msg} = \mathsf{msg}'\right] = 1.$$

- ***Perfect binding using honest CRS.*** *For an honest CRS, the commitment is perfectly binding. That is, there do not exist (x, r) and (x', r') such that*

$$\mathsf{Com}(\mathsf{crs}, x; r) = \mathsf{Com}(\mathsf{crs}, x'; r'),$$

where $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$.

- ***(Perfect) Semantic Security.*** *Let* msg *and* msg' *be any two messages. For all circuit G, input x, and $y \neq G(x)$, it holds that*

$$\mathsf{WEnc}(\mathsf{crs}, \mathsf{msg}, (c, G, y)) \ \equiv\ \mathsf{WEnc}(\mathsf{crs}, \mathsf{msg}', (c, G, y)),$$

where $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$ *and* $c \leftarrow \mathsf{Com}(\mathsf{crs}, x)$*. That is, when the CRS* crs *and commitment c are sampled honestly, then the witness encryption satisfies perfect semantic security.*

- ***Zero-knowledge.***[8] *For any PPT adversary $(\mathcal{A}_1, \mathcal{A}_2)$, it holds that*

$$\left|\Pr\left[\begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{state}, x) \leftarrow \mathcal{A}_1(\mathsf{crs}) \\ c = \mathsf{Com}(\mathsf{crs}, x; r) \end{array} : \mathcal{A}_2^{\mathcal{O}_1(\cdot)}(\mathsf{state}, c, (x, r)) = 1\right] - \Pr\left[\begin{array}{c} (\mathsf{crs}, \tau) \leftarrow \mathsf{SimSetup}(1^\lambda) \\ (\mathsf{state}, x) \leftarrow \mathcal{A}_1(\mathsf{crs}) \\ (c, \mathsf{aux}) = \mathsf{SimCom}(\mathsf{crs}) \\ r = \mathsf{SimDecom}(\mathsf{crs}, \tau, c, \mathsf{aux}, x) \end{array} : \mathcal{A}_2^{\mathcal{O}_2(\cdot)}(\mathsf{state}, c, (x, r)) = 1\right]\right| = \mathsf{negl}(\lambda),$$

where $\mathcal{O}_1(G) := \mathsf{Prove}(\mathsf{crs}, c, G, (x, r))$ *and* $\mathcal{O}_2(G) := \mathsf{SimProve}((\mathsf{crs}, \tau, \mathsf{aux}), (c, G, G(x)))$*. That is, the adversary could choose the message x, and is given its commitment c with the decommitment (x, r). Still, given oracle access to the proof of $(c, G, G(x))$, where the adversary chooses the circuit G arbitrarily, it cannot distinguish the simulated proof from the honest proof.*

Observe that these properties implicitly guarantee additional properties. For example, the zero-knowledge property implies that the honest CRS and the simulated CRS are computationally indistinguishable. Since our construction does not explicitly use those properties, we do not state them explicitly here. We will refer the readers to [BL20] for details.

[8] Our definition is slightly different from the zero-knowledge definition in [BL20]. In particular, in our definition, the adversary is additionally given the decommitment r. Nonetheless, the construction of [BL20] satisfies our definition since the zero-knowledge property holds for any circuit that the adversary queries. For example, the adversary may query a circuit G defined to be $G(x) = x_1$, where $x = (x_1, \ldots, x_N)$. In this case, the construction of [BL20] simply sends the decommitment of x_1 as the proof. Therefore, without loss of generality, we may assume that the adversary also has the decommitment information.

Instantiation. We will use a witness encryption for NIZK of commitment scheme that supports all polynomial-size circuits, recently constructed by [BL20] under pairing assumptions.

Locality. The construction of Benhamouda and Lin [BL20] satisfies the following local property. To commit to a message $x = (x_1, \ldots, x_N)$, Com actually commits to every x_i independently. That is, $\mathsf{Com}(\mathsf{CRS}, x; r) = (\mathsf{Com}'(\mathsf{CRS}, x_1; r_1), \ldots, \mathsf{Com}'(\mathsf{CRS}, x_N; r_N))$, where Com' is some subroutine that commits a single group element. Moreover, suppose G is a circuit that only depends on m coordinates from x. Given RAM access to the commitment $c = (c_1, \ldots, c_N)$, where $c_i = \mathsf{Com}'(\mathsf{CRS}, x_i; r_i)$, the running times of both generating the NIZK proof π of the statement $(c, G, G(x))$ and the witness encryption/decryption with $\big((c, G, G(x))\, , \pi\big)$ depend only on the locality m. In particular, if G depends only on $x_{i_1}, \ldots, x_{i_m}$, then the statement $\mathsf{st} = (c, G, G(x))$ can be expressed succinctly as $\mathsf{st}' = \big((c_{i_1}, \ldots, c_{i_m}), G, G(x_{i_1}, \ldots, x_{i_m})\big)$ and the witness $(x_{i_1}, r_{i_1}, \ldots, x_{i_m}, r_{i_m})$ is succinct as well.

In summary, if the circuit G only depends on m coordinates of its input x, then the encryption/decryption and NIZK proof process all enjoy locality m.

4.3 Construction of Big-key IB-KEM

Our construction of the big-key identity-based key encapsulation mechanism is described in Figure 2. Note that, while the scheme below has a large public parameter pp, we will show how to transform the scheme to one with a short pp in Supporting material Section 5.1.

Construction Overview. Our construction employs witness encryption for NIZK of commitment scheme and a puncturable local pseudo-entropy function.

- **Setup.** Let $\ell > 0$ be a fixed parameter (which we will use for the leakage bound later). To set up a public parameter and a master public-key, we shall first sample a CRS $\mathsf{crs}_{\mathsf{pef}}$, a key k for the PEF, and also a CRS crs for the witness encryption for NIZK of commitment scheme. The $(\mathsf{crs}_{\mathsf{pef}}, \mathsf{crs})$ and the commitment c of the secret-key k shall be the public parameter. The master secret-key shall be the secret-key k and the necessary decommitment information $(r_1, \ldots, r_N)$.
- **Identity Secret-key.** The identity secret-key $\mathsf{sk}_{\mathsf{id}}$ consists of two parts. The first part is the evaluation of the PEF, i.e., $\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id}) = (y_1, \ldots, y_\lambda)$. Second, for every index $i \in [\lambda]$, we generate a proof π_i proving the statement that c is a commitment of the key k such that $\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id})_i = y_i$. Therefore, the identity secret-key $\mathsf{sk}_{\mathsf{id}}$ is $\{y_i, \pi_i\}_{i=1}^{\lambda}$.
- **Encapsulation.** To sample a ciphertext encapsulating a key, we shall use the witness encryption. In particular, we sample a random string $v = (v_1, \ldots, v_\lambda)$. For every index $i \in [\lambda]$, we encrypt v_i twice as[9]

$$\mathsf{ct}_0^i := \mathsf{WEnc}\Big(\mathsf{crs}, v_i, \big(c, (\mathsf{id}, i), 0\big)\Big) \text{ and } \mathsf{ct}_1^i := \mathsf{WEnc}\Big(\mathsf{crs}, v_i, \big(c, (\mathsf{id}, i), 1\big)\Big)\,.$$

That is, we encrypt v_i using two different statements. The 0-statement is that c is a commitment of k such that $\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id})_i = 0$ and the 1-statement is that c is a commitment of k such that $\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id})_i = 1$.[10] Finally, we ask the encryptor to sample an additional seed

[9] We abuse notation and write (id, i) for a circuit. Refer to the figure for the definition of the circuit (id, i).

[10] Observe that only one of the statements will be in the language $\mathcal{L}$ due to the perfect binding property.

s, and we shall apply the seeded extractor $\mathsf{Ext}(\cdot, s)$ on the string v. That is, the ciphertext is $\Big(\big\{\mathsf{ct}_0^i, \mathsf{ct}_1^i\big\}_{i=1}^{\lambda}, s\Big)$ and the encapsulated key is $u = \mathsf{Ext}(v, s)$.

Building Blocks:

1. $(\mathsf{Setup}', \mathsf{Com}, \mathsf{Prove}, \mathsf{WEnc}, \mathsf{WDec}, \mathsf{SimSetup}', \mathsf{SimCom}, \mathsf{SimDecom}, \mathsf{SimProve})$ be a witness encryption for NIZK of commitments (Definition 4).
2. $(\mathsf{Gen}, \mathsf{PEF})$ be a puncturable local pseudo-entropy function (Definition 2), where given a $\mathsf{crs}_{\mathsf{pef}}$ and key k generated by Gen, $\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \cdot) \colon \{0,1\}^{\lambda} \to \{0,1\}^{\lambda}$ accesses at most $m(\lambda)$ locations of the key k (locality). Fix the parameter $\ell > 0$, taken as input by Gen.
3. Let $\mathsf{Ext} \colon \{0,1\}^{\lambda} \times \{0,1\}^{\mu} \to \{0,1\}^{\lambda'}$ be a seeded randomness extractor.

Notation for circuits:

- For a fixed $\mathsf{crs}_{\mathsf{pef}}$, for brevity, we abuse notation and write (id, i) for a circuit $G \colon \{0,1\}^N \to \{0,1\}$ defined as
$$G(x) := \mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, x, \mathsf{id})_i.$$
That is, given the input x, $G(x)$ outputs the i^{th} bit of the output PEF with key x and input id.

The Construction:

- $\mathsf{Setup}(1^{\lambda})$: Let $(\mathsf{crs}_{\mathsf{pef}}, k) \leftarrow \mathsf{Gen}(1^{\lambda}, \ell)$. $\mathsf{crs} \leftarrow \mathsf{Setup}'(1^{\lambda})$. Let $k := (k_1, \ldots, k_N)$. For $i \in [N]$, sample r_i at random. For all $i \in [\lambda]$, let $c_i = \mathsf{Com}(\mathsf{crs}, k_i; r_i)$. Return $\mathsf{msk} := \{k_i, r_i\}_{i=1}^{N}$ and $\mathsf{pp} := (\mathsf{crs}_{\mathsf{pef}}, \mathsf{crs}, c_1, \ldots, c_N)$
- $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$: Given the input id, let $t_1, t_2, \ldots, t_m$ be the indices of the key k that $\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, \cdot, \mathsf{id})$ depends on. Let $y_i = \mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id})_i$. Let statement $\mathsf{st}_i := \Big((c_{t_1}, \ldots, c_{t_m}), (\mathsf{id}, i), y_i\Big) \in \mathcal{L}$. Define and return
$$\mathsf{sk}_{\mathsf{id}} := \Big\{ y_i, \mathsf{Prove}\Big(\mathsf{crs}, \mathsf{st}_i, \{k_{t_j}, r_{t_j}\}_{j=1}^{m}\Big)\Big\}_{i=1}^{\lambda}.$$
- $\mathsf{Encap}(\mathsf{id})$: For all $i \in [\lambda]$, sample $v_i \leftarrow \{0,1\}$. Let $v := (v_1, v_2, \ldots, v_{\lambda})$. Let $s \leftarrow \{0,1\}^{\mu}$. For all $i \in [\lambda]$, define
$$\mathsf{ct}_0^i := \mathsf{WEnc}\Big(\mathsf{crs}, v_i, \Big((c_{t_1}, \ldots, c_{t_m}), (\mathsf{id}, i), 0\Big)\Big);$$
$$\mathsf{ct}_1^i := \mathsf{WEnc}\Big(\mathsf{crs}, v_i, \Big((c_{t_1}, \ldots, c_{t_m}), (\mathsf{id}, i), 1\Big)\Big).$$
Let $\mathsf{ct} := \Big(\big\{\mathsf{ct}_0^i, \mathsf{ct}_1^i\big\}_{i=1}^{\lambda}, s\Big)$ and $u := \mathsf{Ext}(v, s)$. Return (ct, u).

- $\mathsf{Dec}\Big(\mathsf{id},\ \mathsf{ct} = \Big(\{\mathsf{ct}_0^i, \mathsf{ct}_1^i\}_{i=1}^{\lambda}, s\Big), \mathsf{sk}_{\mathsf{id}} = \{y_i, \pi_i\}_{i=1}^{\lambda}\Big)$: For all $i \in [\lambda]$, define

$$v_i := \mathsf{WDec}\left(\mathsf{crs}, \mathsf{ct}_{y_i}^i, \mathsf{st}_i, \pi_i\right).$$

  Let $v := (v_1, \ldots, v_n)$ and $u := \mathsf{Ext}(v, s)$. Return u.

Auxiliary Algorithms for the Security Proof:

- $\mathsf{SimSetup}(1^\lambda) : (\mathsf{crs}_{\mathsf{pef}}, k) \leftarrow \mathsf{Gen}(1^\lambda, \ell)$, $(\mathsf{crs}, \tau) \leftarrow \mathsf{SimSetup}'(1^\lambda)$. Let $(c_i, \mathsf{aux}_i) = \mathsf{SimCom}(\mathsf{crs})$, $k := (k_1, \ldots,\ k_N)$ and $r_i = \mathsf{SimDecom}(\mathsf{crs},\ \tau, c_i,\ \mathsf{aux}_i, k_i)$. Return $\mathsf{msk} := \{k_i, r_i\}_{i=1}^N$ and $\mathsf{pp} := (\mathsf{crs}_{\mathsf{pef}}, \mathsf{crs}, c_1, \ldots, c_n)$.

- $\mathsf{SimKeyGen}(\mathsf{msk}, \mathsf{id})$: Let $(t_1, \ldots, t_m)$, $(y_1, \ldots, y_\lambda)$, and $(\mathsf{st}_1, \ldots, \mathsf{st}_\lambda)$ be as defined in KeyGen. Define and return

$$\mathsf{sk}_{\mathsf{id}} := \Big\{y_i, \mathsf{SimProve}\Big((\mathsf{crs}, \tau, \{\mathsf{aux}_{t_j}\}_{j=1}^m), \mathsf{st}_i\Big)\Big\}_{i=1}^{\lambda}.$$

- $\mathsf{Encap}^*(\mathsf{id})$: For all $i \in [\lambda]$, sample $v_i \leftarrow \{0,1\}$. Sample $s \leftarrow \{0,1\}^\mu$. Define

$$\mathsf{ct}_0^i := \mathsf{WEnc}\Big(\mathsf{crs}, v_i, \Big((c_{t_1}, \ldots, c_{t_m}), (\mathsf{id}, i), 0\Big)\Big)$$
$$\mathsf{ct}_1^i := \mathsf{WEnc}\Big(\mathsf{crs}, v_i + 1, \Big((c_{t_1}, \ldots, c_{t_m}), (\mathsf{id}, i), 1\Big)\Big)$$

  Let $\mathsf{ct} := \Big(\{\mathsf{ct}_0^i, \mathsf{ct}_1^i\}_{i=1}^{\lambda}, s\Big)$. Return ct.
  Observe that Encap^* does not output an associated key u. In particular, the decryption of $\mathsf{ct} \leftarrow \mathsf{Encap}^*$ will be $\mathsf{Ext}((v_1 + y_1, \ldots, v_\lambda + y_\lambda), s)$.

Figure 2: Our Big-key IB-KEM

Remark 4 (Need for a $\mathsf{crs}_{\mathsf{pef}}$). *Note that the* Encap *algorithm above requires the knowledge of the exact m locations of k that were accessed by the* PEF. *This information is fixed and public, given the* $\mathsf{crs}_{\mathsf{pef}}$ *and the input* id. *Thus having a* $\mathsf{crs}_{\mathsf{pef}}$ *is essential to ensure that the* Encap *algorithm remains efficient and local (i.e., independent of ℓ). This explains why our PEF construction has a CRS.*

We will prove the selective security of the above construction, assuming the selective security of the underlying PEF, along with the security guarantees of the witness encryption scheme. We formally state the theorem below.

Theorem 3. *Assuming that the pseudo-entropy function* PEF *satisfies the selective security (Definition 2) and assuming the security of the witness encryption for NIZK of the commitment scheme (Definition 4), the IB-KEM construction from Figure 2 is a big-key identity-based key encapsulation mechanism that satisfies the selective security under bounded leakage (Definition 3). In particular, we can instantiate the underlying schemes to get a leakage rate (i.e., $\frac{\ell}{|\mathsf{msk}|}$, where ℓ is the size of the leakage allowed on* msk*) of* $1/3$.

The correctness of our construction follows from the correctness of the witness encryption scheme. The efficiency property follows from the locality of both the PEF, and the witness encryption for the NIZK of commitment scheme. We now give a full proof of the selective security under bounded leakage.

4.4 Proof of Selective Security Under Bounded Leakage

Proof Overview. Our selective security proof mainly consists of the following steps.

- **Switch to invalid ciphertext.** We first define another encapsulation algorithm Encap^* that generates an invalid ciphertext ct. ct is invalid in that the two ciphertexts ct_i^0 and ct_i^1 encrypt two different messages. Our first step is to switch from a valid ciphertext using Encap to an invalid ciphertext using Encap^*. Since only one of the two statements (i.e., $(c, (\mathsf{id}, i), 0)$ and $(c, (\mathsf{id}, i), 1)$) is in the language, by the semantic security of the witness encryption scheme, the two hybrids are indistinguishable.

- **Switch to the simulation mode.** Next, we define two auxiliary algorithms $\mathsf{SimSetup}$ and $\mathsf{SimKeyGen}$. In these two algorithms, instead of generates the CRS and proof honestly, we switch to the simulation mode. That is, the CRS and commitments are generated with trapdoors such that they are equivocal. Then, all the proofs in the identity secret-key are given by the simulated proof. By the zero-knowledge property of the witness encryption for NIZK of commitment scheme, these two hybrids are indistinguishable.

- **Switch to the punctured mode.** In this step, we shall sample the key of the PEF using the punctured mode. By invoking the mode-indistinguishability of the PEF, the two hybrids are indistinguishable. Note that the key k sampled in the punctured mode comes with a punctured key $k^{\odot}$, where the identities $\{\mathsf{id} \in \mathcal{J}\}$ are the punctured places. This allows us to sample identity secret-keys for all identities but those from the challenge set $\mathcal{J}$. Crucially, this implies that the entire view of the adversary can be simulated using only $k^{\odot}$, *without* k.

- **Invoke the security of PEF and the randomness extractor Ext.** Finally, we argue that the adversary cannot distinguish the key encapsulated inside the (invalid) ciphertext from a random string. We reduce this to the security of the PEF. Intuitively, the output of the PEF at $\mathcal{J}$, i.e., $\{\mathsf{PEF}(\mathsf{crs}, k, \mathsf{id}) : \mathsf{id} \in \mathcal{J}\}$, guarantees sufficiently high entropy even conditioned on the adversary's view (which only depends on $k^{\odot}$), and hence we can use the extractor security.

Proof. Now, we will prove that our scheme from Figure 2 satisfies the selective security under a bounded leakage from the master secret key, i.e., we show that for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$, trying to break the selective security game $\mathrm{IND}^{\mathsf{blsKEM}}(1^\lambda)$ (refer to Figure 1) under ℓ-leakage, and for all non-negligible ε, it holds that:

$$\Pr_{(\mathsf{msk},\mathsf{pp},\mathcal{J},\mathsf{state},\mathsf{leak})}\left[\forall\ \mathsf{id} \in \mathcal{J}, \mathsf{Adv}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak}) \geq \varepsilon\right] = \mathsf{negl}(\lambda),$$

where $(\mathsf{msk}, \mathsf{pp}, \mathcal{J}, \mathsf{state}, \mathsf{leak})$ are sampled from the Phase I of $\mathrm{IND}^{\mathsf{blsKEM}}(1^\lambda)$ and the random variable $\mathsf{Adv}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ is defined as follows.

$$\mathsf{Adv}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak}) = \left|\Pr[\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak}) = 1] - \frac{1}{2}\right|$$

Here, the random variable $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ is as defined in Phase II of $\mathrm{IND}^{\mathsf{blsKEM}}(1^\lambda)$, and $\mathcal{A}_3$ is not allowed to query the KeyGen on $\mathcal{J}$.
We prove this using a sequence of indistinguishable hybrids described below.

Hybrid 0: This hybrid is the real distribution $\mathrm{IND}^{\mathsf{blsKEM}}(1^\lambda)$ (recall that $\mathcal{A}_3$ is not allowed to query KeyGen on the challenge identities $\mathcal{J}$), defined as:

- **Phase I.** The system is set up as follows.
  1. Let $(\mathcal{J}, \mathsf{state}) \leftarrow \mathcal{A}_1(1^\lambda)$, where $\mathcal{J}$ is a set of identities such that $|\mathcal{J}| = \ell + 1$.
  2. $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{Setup}(1^\lambda)$.
  3. $f \leftarrow \mathcal{A}_2(\mathsf{state}, \mathsf{pp})$, where the output length of f is (at most) ℓ. Let $\mathsf{leak} := f(\mathsf{msk})$.
- **Phase II.** For any $\mathsf{id} \in \mathcal{J}$, we define a security game $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ as follows
  1. $(\mathsf{ct}, u) \leftarrow \mathsf{Encap}(\mathsf{id})$.
  2. Let u' be an independent random string.
  3. Sample $b \leftarrow \{0, 1\}$.
  4. If $b = 0$, let $b' = \mathcal{A}_3^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}, \mathsf{ct}, u)$;
  5. If $b = 1$, let $b' = \mathcal{A}_3^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}, \mathsf{ct}, u')$;
  6. Output 1 if $b = b'$; otherwise, output 0.

Hybrid 1: This hybrid is identical to Hybrid 0, except that for each $\mathsf{id} \in \mathcal{J}$, instead of using Encap to generate the challenge ciphertext and the key, we use Encap^* to sample the invalid ciphertext and give its decryption to the adversary when the choice bit b is 0.

- **Phase I.** The system is set up as follows.
  1. Let $(\mathcal{J}, \mathsf{state}) \leftarrow \mathcal{A}_1(1^\lambda)$, where $\mathcal{J}$ is a set of identities such that $|\mathcal{J}| = \ell + 1$.
  2. $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{Setup}(1^\lambda)$.
  3. $f \leftarrow \mathcal{A}_2(\mathsf{state}, \mathsf{pp})$, where the output length of f is (at most) ℓ. Let $\mathsf{leak} := f(\mathsf{msk})$.
- **Phase II.** For any $\mathsf{id} \in \mathcal{J}$, we define a security game $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ as follows
  1. $\mathsf{ct} \leftarrow \mathsf{Encap}^*(\mathsf{id})$.
  2. Let u' be an independent random string.
  3. Sample $b \leftarrow \{0, 1\}$.
  4. If $b = 0$, let $b' = \mathcal{A}_3^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}, \mathsf{ct}, \mathsf{Dec}(\mathsf{id}, \mathsf{ct}, \mathsf{sk}_{\mathsf{id}}))$;
  5. If $b = 1$, let $b' = \mathcal{A}_3^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}, \mathsf{ct}, u')$;
  6. Output 1 if $b = b'$; otherwise, output 0.

Claim 1. *Hybrid 0 and Hybrid 1 are identically distributed.*

Proof. For simplicity, let us consider the hybrid where we only switch from Encap to Encap^* for a single id (say, the first one in $\mathcal{J}$). The indistinguishability when we make the switch for each of the other $\mathsf{id} \in \mathcal{J}$ is entirely analogous.

Let us consider a sequence of hybrid $H_0, H_1, \ldots, H_\lambda$, where, in hybrid H_i, the ciphertext $\mathsf{ct} = \left(\left\{\mathsf{ct}_0^j, \mathsf{ct}_1^j\right\}_{j=1}^{\lambda}, s\right)$ corresponding to id is sampled as follows. For all $j > i$, $\mathsf{ct}_0^j, \mathsf{ct}_1^j$ are sampled according to Encap, but for all $j \le i$, $\mathsf{ct}_0^j, \mathsf{ct}_1^j$ are sampled according to Encap^*. We shall prove that for all i, H_{i-1} and H_i are identically distributed. In particular, if H_{i-1} and H_i are not identically distributed, we shall construct an adversary $\mathcal{B}$ that breaks the perfect semantic security of the witness encryption scheme.

$\mathcal{B}$ interacts with a challenger $\mathcal{C}$ for the witness encryption scheme. $\mathcal{B}$ receives from $\mathcal{C}$ a CRS crs. It proceeds to simulate the hybrid as described until it needs to sample ct_0^i and ct_1^i. Let $y_i = \mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id})_i$. It samples the bit v_i uniformly at random. It encrypts $\mathsf{ct}_{y_i}^{*,i}$ as

$$\mathsf{WEnc}(\mathsf{crs}, v_i, (c, (\mathsf{id}, i), y_i))).$$

Note that, $\mathsf{ct}_{y_i}^{*,i}$ is an encryption of $v_i + y_i$ in the subroutine Encap^*, but since v_i and $v_i + y_i$ are identically distributed, $\mathsf{ct}_{y_i}^{*,i}$ is also identically distributed in both hybrids. As for $\mathsf{ct}_{1-y_i}^{*,i}$, it sends the statement $(c, (\mathsf{id}, i), 1 - y_i)$ to the challenger together with two messages 0 and 1.

Observe that this statement is always not in the language $\mathcal{L}$ since the commitment c is perfectly binding under an honest CRS.

It receives back a ciphertext α encrypting either 0 or 1. It uses α for $\mathsf{ct}_{1-y_i}^{*,i}$. The rest of the hybrid is simulated exactly as described. In particular, $\mathsf{Dec}(\mathsf{id}, \mathsf{ct}^*, \mathsf{sk}_{\mathsf{id}})$ is identical to u. Depending on whether α is an encryption of v_i or not, we simulate either H_{i-1} or H_i.

Now, if H_{i-1} and H_i are not identically distributed, $\mathcal{B}$ breaks the perfect semantic security of the witness encryption scheme. This completes the proof. □

Hybrid 2: This hybrid is identical to Hybrid 1, except that we use the subroutines $\mathsf{SimSetup}$ and $\mathsf{SimKeyGen}$ instead of using Setup and KeyGen. This switches the actual NIZK proofs with the simulated ones.

- **Phase I.** The system is set up as follows.
    1. Let $(\mathcal{J}, \mathsf{state}) \leftarrow \mathcal{A}_1(1^\lambda)$, where $\mathcal{J}$ is a set of identities such that $|\mathcal{J}| = \ell + 1$.
    2. $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{SimSetup}(1^\lambda)$.
    3. $f \leftarrow \mathcal{A}_2(\mathsf{state}, \mathsf{pp})$, where the output length of f is (at most) ℓ. Let $\mathsf{leak} := f(\mathsf{msk})$.
- **Phase II.** For any $\mathsf{id} \in \mathcal{J}$, we define a security game $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ as follows
    1. $\mathsf{ct} \leftarrow \mathsf{Encap}^*(\mathsf{id})$.
    2. Let u' be an independent random string.
    3. Sample $b \leftarrow \{0, 1\}$.
    4. If $b = 0$, let $b' = \mathcal{A}_3^{\mathsf{SimKeyGen}(\mathsf{msk}, \cdot)}(\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}, \mathsf{ct}, \mathsf{Dec}(\mathsf{id}, \mathsf{ct}, \mathsf{sk}_{\mathsf{id}}))$;
    5. If $b = 1$, let $b' = \mathcal{A}_3^{\mathsf{SimKeyGen}(\mathsf{msk}, \cdot)}(\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}, \mathsf{ct}, u')$;
    6. Output 1 if $b = b'$; otherwise, output 0.

Claim 2. *Hybrid 1 and Hybrid 2 are computationally indistinguishable.*

Proof. If Hybrid 1 and Hybrid 2 are computationally distinguishable, we shall construct an adversary $\mathcal{B}$ that breaks the zero-knowledge property of the witness encryption for NIZK of commitment scheme.

The reduction is straightforward. $\mathcal{B}$ interacts with the challenger $\mathcal{C}$, where the challenger is either in honest mode or simulation mode (refer to the definition of the zero-knowledge property).

The challenger samples the crs and send it to $\mathcal{B}$. $\mathcal{B}$ samples a random key k and send it to the challenger. The challenger replies back with the commitments $c = (c_1, \ldots, c_N)$ and the decommitments $(k_1, r_1), \ldots, (k_N, r_N)$. Now, $\mathcal{B}$ proceeds to simulate the hybrid as described. During the simulation, when a query is sent for the identity secret-key $\mathsf{sk}_{\mathsf{id}} = \{y_i, \pi_i\}_{i=1}^{\lambda}$, $\mathcal{B}$ first computes $(y_1, \ldots, y_\lambda) = \mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id})$. She can compute this as she knows the key k. Then, she obtains the proof π_i by quering the challenger with the circuit (id, i).

Depending on whether the challenger is in the honest mode or simulation mode, she simulates either Hybrid 1 or Hybrid 2. Therefore, if the hybrids are computationally distinguishable, $\mathcal{B}$ breaks the zero-knowledge property of the witness encryption for NIZK of commitment scheme. □

Hybrid 3: This hybrid is identical to Hybrid 2, except that we will switch to using the punctured key of the PEF (punctured at the points $\mathsf{id} \in \mathcal{J}$) for answering all the $\mathsf{SimKeyGen}$ queries.

- **Phase I.** The system is set up as follows.
  1. Let $(\mathcal{J}, \mathsf{state}) \leftarrow \mathcal{A}_1(1^\lambda)$, where $\mathcal{J}$ is a set of identities such that $|\mathcal{J}| = \ell + 1$.
  2. $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{SimSetup}^{\odot}(1^\lambda)$. Here, $\mathsf{SimSetup}^{\odot}$ first generates $(\mathsf{crs}_{\mathsf{pef}}, k, k^{\odot}) \leftarrow \mathsf{Gen}_2(1^\lambda, N, \mathcal{J})$ and uses k in msk and pp, generated as in $\mathsf{SimSetup}$.
  3. $f \leftarrow \mathcal{A}_2(\mathsf{state}, \mathsf{pp})$, where the output length of f is (at most) ℓ. Let $\mathsf{leak} := f(\mathsf{msk})$.
- **Phase II.** For any $\mathsf{id} \in \mathcal{J}$, we define a security game $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ as follows
  1. $\mathsf{ct} \leftarrow \mathsf{Encap}^*(\mathsf{id})$.
  2. Let u' be an independent random string.
  3. Sample $b \leftarrow \{0, 1\}$.
  4. If $b = 0$, let $b' = \mathcal{A}_3^{\mathsf{SimKeyGen}^{\odot}(\mathsf{msk}, \cdot)}(\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}, \mathsf{ct}, \mathsf{Dec}(\mathsf{id}, \mathsf{ct}, \mathsf{sk}_{\mathsf{id}}))$;
  5. If $b = 1$, let $b' = \mathcal{A}_3^{\mathsf{SimKeyGen}^{\odot}(\mathsf{msk}, \cdot)}(\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}, \mathsf{ct}, u')$;
     Here, $\mathsf{SimKeyGen}^{\odot}$ works exactly like $\mathsf{SimKeyGen}$, except that it uses $\mathsf{PEF}_2(\mathsf{crs}_{\mathsf{pef}}, k^{\odot}, .)$ for the PEF evaluations.
  6. Output 1 if $b = b'$; otherwise, output 0.

Claim 3. *Hybrid 2 and Hybrid 3 are computationally indistinguishable.*

Proof. We use the mode indistinguishability of the PEF to prove the claim. Particularly, if Hybrid 2 and Hybrid 3 were computationally distinguishable, we can build an adversary $\mathcal{B}$ breaking the mode indistinguishability of the PEF.

$\mathcal{B}$ sends the challenge inputs $\mathcal{J}$ and receives the $(\mathsf{crs}_{\mathsf{pef}}, k)$ from the mode indistinguishability challenger, which either corresponds to the actual key generation or the punctured mode. Having this, $\mathcal{B}$ can simulate the entire hybrids, while using k to answer the $\mathsf{SimKeyGen}$ or $\mathsf{SimKeyGen}^{\odot}$. Since the queries do not contain the punctured points $\mathcal{J}$, by the punctured correctness, the $\mathsf{SimKeyGen}^{\odot}$ responses will be same as the the PEF evaluations on k. Depending on whether the challenger returns the actual PEF key or the one in the punctured mode, $\mathcal{B}$ simulates Hybrid 2 or Hybrid 3.

Thus, if the two hybrids are distinguishable, $\mathcal{B}$ can break the mode indistinguishability of PEF. This completes the proof of the claim. □

Observe that, in the case $b = 0$, in Hybrid 3, $\mathsf{Dec}(\mathsf{id}, \mathsf{ct}, \mathsf{sk}_{\mathsf{id}}) = \mathsf{Ext}((v_1 + y_1, v_2 + y_2, \ldots, v_\lambda + y_\lambda), s)$, where $(y_1, \cdots, y_\lambda) = \mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id})$, the PEF output on the original key k. We will use this in completing the proof below.

Proving Selective Security: To finish proving the selective security, we need to show that for all non-negligible ε, it holds that:

$$\Pr_{(\mathsf{msk}, \mathsf{pp}, \mathcal{J}, \mathsf{state}, \mathsf{leak})} \left[\forall \ \mathsf{id} \in \mathcal{J}, \mathsf{Adv}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak}) \geq \varepsilon\right] = \mathsf{negl}(\lambda), \qquad (1)$$

where $(\mathsf{msk}, \mathsf{pp}, \mathcal{J}, \mathsf{state}, \mathsf{leak})$ are sampled from the Phase I of Hybrid 3 and the random variable $\mathsf{Adv}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ is defined as follows.

$$\mathsf{Adv}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak}) = \left|\Pr[\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak}) = 1] - \frac{1}{2}\right|$$

Here, the random variable $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ is as defined in Phase II of Hybrid 3. By the $|\mathcal{J}|$-selective, $\gamma \cdot |\mathcal{J}|$-pseudo-entropy security of the PEF (Theorem 2), we have that

$$\tilde{H}_\infty\Big(\{\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id}) : \mathsf{id} \in \mathcal{J}\} \ \Big| \ \mathsf{crs}_{\mathsf{pef}}, k^{\odot}\Big) \geq \gamma \cdot |\mathcal{J}|.$$

Here, note that the leakage $f(\mathsf{msk})$ in Hybrid 4, takes as input k and $(r_1, \cdots, r_N)$, and depends on pp, which in turn depends on $\mathsf{crs}_{\mathsf{pef}}$. Hence, we can define the following function g on the PEF key k, by hardwiring the values $(\mathsf{crs}_{\mathsf{pef}}, \tau, \{c_i, \mathsf{aux}_i\}_{i=1}^{\lambda})$:

$$g(k_1, k_2, \ldots, k_N) := \left\{ \begin{array}{c} \forall i, \ r_i = \mathsf{SimDecom}(\mathsf{crs}, \tau, c_i, \mathsf{aux}_i, k_i) \\ \text{Output } f((k_1, r_1), \ldots, (k_N, r_N)) \end{array} \right\}.$$

Thus, $f(\mathsf{msk}) = g(k)$, in Hybrid 4. Now, by Lemma 1, in the presence of this ℓ-bit leakage on msk we get that

$$\tilde{H}_\infty\Big(\{\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id}) : \mathsf{id} \in \mathcal{J}\} \ \Big| \ \mathsf{crs}_{\mathsf{pef}}, k^{\odot}, f(\mathsf{msk})\Big) \geq \gamma \cdot |\mathcal{J}| - \ell.$$

Now, by Lemma 1, with overwhelming probability over the fixing of $\mathsf{crs}_{\mathsf{pef}}, k^{\odot}, f(\mathsf{msk})$, we have

$$H_\infty\Big(\{\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id}) : \mathsf{id} \in \mathcal{J}\}\Big) \geq \Theta(\gamma \cdot |\mathcal{J}| - \ell).$$

Next, by Lemma 2, there exists a distribution I over the identities $\mathcal{J}$ such that

$$\tilde{H}_\infty(\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, I)|I) \geq \frac{\Theta(\gamma \cdot |\mathcal{J}| - \ell)}{|\mathcal{J}|} - \log|\mathcal{J}| = \Theta(\gamma) \qquad \text{(Recall } |\mathcal{J}| = \ell + 1.)$$

In other words, with high probability (in particular, over the observed leakage and I), there exists an $\mathsf{id}^* \in \mathcal{J}$ such that the min-entropy of $\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id}^*)$ is $\geq t_{ext}$, where we set $t_{ext} = \Theta(\gamma)$. Now, by the definition of randomness extractor, we can send $\mathcal{A}_3$ a uniform string u' irrespective of the choice of b in Hybrid 4, making the output of $\mathsf{Exp}^{\mathsf{id}^*}$ uniformly random (as b' would be uncorrelated to b).

The extractor security can be applied in $\mathsf{Exp}^{\mathsf{id}^*}$, because:

- The source $\mathsf{PEF}(\mathsf{crs}_{\mathsf{pef}}, k, \mathsf{id}^*)$ has high entropy, given $\mathsf{crs}_{\mathsf{pef}}, k^{\odot}$ and $f(\mathsf{msk})$.
- The view of the adversary in this game is $\mathsf{state}, \mathsf{leak}, \mathsf{pp}, \mathsf{id}^*, \mathsf{ct} = \left(\{\mathsf{ct}_0^i, \mathsf{ct}_1^i\}_{i=1}^{\lambda}, s\right)$, where the seed s is uniformly random and independent from everything else in the hybrid.
- $(v_1, v_2, \ldots, v_\lambda)$ is independent of $(\mathsf{crs}_{\mathsf{pef}}, f(\mathsf{msk}), \mathsf{id}^*, k^{\odot})$, but is correlated with ct and, hence, the adversary's view.

Thus, given the adversary's view in $\mathsf{Exp}^{\mathsf{id}^*}$, it cannot distinguish

$$\mathsf{Ext}\Big((v_1 + y_1, \ldots, v_\lambda + y_\lambda), s\Big),$$

which is what $\mathcal{A}_3$ gets in Hybrid 3 when $b = 0$, from uniform since $(y_1, \ldots, y_\lambda)$ is sampled from a high min-entropy distribution that is independent of $(v_1, \ldots, v_\lambda)$.

Hence, in Hybrid 3, with high probability, there exists $\mathsf{id}^* \in \mathcal{J}$ such that $\mathsf{Exp}^{\mathsf{id}}(\mathsf{msk}, \mathsf{pp}, \mathsf{state}, \mathsf{leak})$ in Phase II, outputs 1 with probability $1/2+\mathsf{negl}(\lambda)$ (where $\mathsf{negl}(\lambda)$ comes from the extractor security error), which implies that the security as needed in Equation 1 holds.

The Claims 1, 2 and 3 and the above argument complete the security proof.

Instantiation and Parameters. We can instantiate our construction with the PEF from Theorem 2, the witness encryption for NIZK of commitment scheme from [BL20] (see Section 4.2) and any randomness extractor (e.g., left-over hash from [HILL99]). We allow a leakage of ℓ bits from our msk. Now, our msk consists of the PEF key k and additionally the randomness r_i's used in the commitment scheme. The witness encryption from [BL20] uses 2 random group elements to commit to a single group element (i.e., the ratio of k_i (being committed) to length of randomness r_i is $1/2$). Since the PEF gives a leakage rate of 1 (Remark 2), our big-key IB-KEM allows a leakage rate of $1/3$. □

5 Big-key IBE scheme

Given a big-key IB-KEM scheme, the construction of the big-key IBE scheme is straightforward. One simply uses the encapsulated key as a one-time pad to mask the message. Since, the adversary cannot distinguish the encapsulated key from random string, it cannot learn any information on the masked message. For completeness, we provide the construction below.

A big-key IBE scheme consists of the following efficient algorithms.

- $(\mathsf{pp}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$: This algorithm takes the security parameter as input, and samples the public parameter pp and a master secret-key msk.
- $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$: This algorithm takes the master secret-key msk and the identity id as inputs, and samples an identity secret-key $\mathsf{sk}_{\mathsf{id}}$.
- $c \leftarrow \mathsf{Encrypt}(\mathsf{id}, m)$: This algorithm takes the identity id and a message m as inputs, and samples a ciphertext c.
- $m = \mathsf{Decrypt}(\mathsf{id}, c, \mathsf{sk}_{\mathsf{id}})$: This algorithm takes the identity id, the ciphertext c, and the identity secret-key $\mathsf{sk}_{\mathsf{id}}$ as inputs, and output a message m.

Construction. Given a IB-KEM scheme $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Dec})$, one may construct an IBE scheme as follows.

- The Setup and KeyGen for the IBE scheme are identical to the Setup and KeyGen of the IB-KEM scheme.
- $\mathsf{Encrypt}(\mathsf{id}, m)$ is defined to be
$$\left\{ \begin{matrix} (\mathsf{ct}, u) \leftarrow \mathsf{Encap}(\mathsf{id}),\ \ c' = u \oplus m \\ \text{Output } c = (\mathsf{ct}, c') \end{matrix} \right\}.$$
- $\mathsf{Decrypt}(\mathsf{id}, c = (\mathsf{ct}, c'), \mathsf{sk}_{\mathsf{id}})$ is defined to be
$$\left\{ \begin{matrix} u = \mathsf{Dec}(\mathsf{id}, \mathsf{ct}, \mathsf{sk}_{\mathsf{id}}),\ \ m = c' \oplus u \\ \text{Output } m \end{matrix} \right\}.$$

Figure 3: Our Big-key IBE scheme

The correctness follows from the correctness of the IB-KEM scheme. The security is also immediate as for all message m, the security of the IB-KEM scheme implies that $\mathsf{Encrypt}(\mathsf{id}, m) = (\mathsf{ct}, c')$ is indistinguishable from (ct, c''), where c'' is an independent uniform string.

5.1 Big-key IBE with Short Public Parameter

In order to transform the IBE scheme $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ from Figure 3 into one with a short public parameter pp, we need the following primitive, known as Non-interactive Secure Computation (NISC) from [CDG+17].

Model for NISC in RAM setting. [CDG+17] Suppose $D \in \{0,1\}^M$ is a large database and P be a program with running time t and a short input x. $\mathsf{NISC} = (\mathsf{nisc.Setup}, \mathsf{nisc.EncData}, \mathsf{nisc.EncProg}, \mathsf{nisc.Dec})$ has the following syntax.

- **Setup:** $\mathsf{nisc.crs} \leftarrow \mathsf{nisc.Setup}(1^\lambda)$.
  The setup outputs the common reference string.
- **Database compressing:** $(m_1, \tilde{D}) \leftarrow \mathsf{nisc.EncData}(\mathsf{nisc.crs}, D)$.
  On input the common reference string and the database $D \in \{0,1\}^M$, it outputs a short message m_1 and a larger state $\tilde{D}$.
- **Program Encryption:** $m_2 \leftarrow \mathsf{nisc.EncProg}(\mathsf{nisc.crs}, m_1, (P, x, t))$.
  It takes as input the $\mathsf{nisc.crs}$, a message m_1, a RAM program P with input x and maximum run-time t. It then outputs another message m_2.
- **Decryption:** $y \leftarrow \mathsf{nisc.Dec}^{\tilde{D}}(\mathsf{nisc.crs}, m_2)$.
  The decryption is modelled as a RAM program getting read and write access to arbitrary locations of its database initially containing $\tilde{D}$. On input $\mathsf{nisc.crs}$ and m_2, it outputs y.

The following conditions are satisfied by NISC:

- **Correctness:** For every database $D \in \{0,1\}^M$, where $M = \mathsf{poly}(\lambda)$ for any polynomial function $\mathsf{poly}(.)$, for every RAM program (P, x, t), it holds that $\Pr[\mathsf{nisc.Dec}^{\tilde{D}}(\mathsf{nisc.crs}, m_2) = P^D(x)] = 1$, where $\mathsf{nisc.crs} \leftarrow \mathsf{nisc.Setup}(1^\lambda)$, $(m_1, \tilde{D}) \leftarrow \mathsf{nisc.EncData}(\mathsf{nisc.crs}, D)$, $m_2 \leftarrow \mathsf{nisc.EncProg}(\mathsf{nisc.crs}, m_1, (P, x, t))$.

- **Privacy:** There exists a PPT simulator $\mathsf{nisc.Sim}$ such that for every database $D \in \{0,1\}^M$, where $M = \mathsf{poly}(\lambda)$ for any polynomial function $\mathsf{poly}(.)$, and for every RAM program (P, x, t), let $y = P^D(x)$ be the output of the program, and $\mathsf{MemAccess}$ be the memory access pattern, then it holds that

$$(\mathsf{nisc.crs}, D, (m_1, \tilde{D}), m_2) \approx_c \mathsf{nisc.Sim}(1^\lambda, D, (y, \mathsf{MemAccess})),$$

  where $\mathsf{nisc.crs} \leftarrow \mathsf{nisc.Setup}(1^\lambda)$, $(m_1, \tilde{D}) \leftarrow \mathsf{nisc.EncData}(\mathsf{nisc.crs}, D)$, $m_2 \leftarrow \mathsf{nisc.EncProg}(\mathsf{nisc.crs}, m_1, (P, x, t))$.

- **Efficiency:** The length of m_1 is a fixed polynomial in λ, independent of the size of D. The algorithm $\mathsf{nisc.EncData}$ runs in time $M \cdot \mathsf{poly}(\lambda, \log M)$, $\mathsf{nisc.EncProg}$ and $\mathsf{nisc.Dec}$ run in time $t \cdot \mathsf{poly}(\lambda, \log M)$.

Given a IBE scheme $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ with large public parameter $\mathsf{pp} \in \{0,1\}^N$ and with the feature that $\mathsf{Encrypt}$ only makes RAM access to only a small part of pp, determined by the id (as is the case for both our KEM construction and our IBE scheme in Figure 3), we build the following IBE scheme with a short public parameter using NISC.

Construction. The new scheme $(\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Encrypt}', \mathsf{Decrypt}')$ is as described below:

- $\mathsf{Setup}'(1^\lambda) : (\mathsf{pp}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$. $\mathsf{nisc.crs} \leftarrow \mathsf{nisc.Setup}(1^\lambda)$. $(h, \tilde{\mathsf{pp}}) \leftarrow \mathsf{nisc.EncData}(\mathsf{nisc.crs}, \mathsf{pp})$. Output $\mathsf{pp}' = (\mathsf{nisc.crs}, h)$ and $\mathsf{msk}' = (\mathsf{msk}, \tilde{\mathsf{pp}})$.

- $\mathsf{KeyGen}'(\mathsf{msk}', \mathsf{id}) : \mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$. Let the part of $\tilde{\mathsf{pp}}$ accessed by $\mathsf{nisc.Dec}$, which depends on id, be denoted by $\tilde{\mathsf{pp}}_{\mathsf{id}}$. Output $\mathsf{sk}'_{\mathsf{id}} = (\mathsf{sk}_{\mathsf{id}}, \tilde{\mathsf{pp}}_{\mathsf{id}})$. Note that since $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$ only access a small part of pp (dependent on id, denoted by $\mathsf{pp}_{\mathsf{id}}$), the corresponding part of $\tilde{\mathsf{pp}}$ (denoted by $\tilde{\mathsf{pp}}_{\mathsf{id}}$ above) accessed by $\mathsf{nisc.Dec}$ is also small.

- $\mathsf{Encrypt}'(\mathsf{id}, m) : c \leftarrow \mathsf{nisc.EncProg}(\mathsf{nisc.crs}, h, (\mathsf{Encrypt}, m, t))$, where $\mathsf{Encrypt}$ is considered as the RAM program making access to the large pp. Output c.

- $\mathsf{Decrypt}'(\mathsf{id}, c, \mathsf{sk}_{\mathsf{id}})$: Output $m \leftarrow \mathsf{nisc.Dec}(\mathsf{nisc.crs}, \tilde{\mathsf{pp}}_{\mathsf{id}}, c)$. Note that the actual $\mathsf{nisc.Dec}$ makes RAM access to $\tilde{\mathsf{pp}}$, but here we give the accessed locations of $\tilde{\mathsf{pp}}$, i.e., $\tilde{\mathsf{pp}}_{\mathsf{id}}$ as an input.

The correctness of the above construction follows from the correctness of the the NISC scheme and the underlying IBE scheme. Further, for all messages m, the privacy of NISC implies that c reveals nothing more than $\mathsf{pp}_{\mathsf{id}}$ and $\mathsf{Encrypt}(\mathsf{id}, m)$, and hence by the security of the underlying IBE scheme, the security of $(\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Encrypt}', \mathsf{Decrypt}')$ follows.
By the efficiency of NISC, the size of the public parameter $\mathsf{pp}' = (\mathsf{nisc.crs}, h)$ is short and independent of the size of $\mathsf{pp} \in \{0,1\}^N$. Further, $\mathsf{Encrypt}'$ and $\mathsf{Decrypt}'$ are both efficient as $\mathsf{nisc.EncProg}$ and $\mathsf{nisc.Dec}$ both run in time $t \cdot \mathsf{poly}(\lambda, \log N)$, where t is the runtime of $\mathsf{Encrypt}$.

Acknowledgements

This research is supported in part by DARPA under Agreement No. HR00112020026, AFOSR Award FA9550-19-1-0200, NSF CNS Award 1936826, and research grants by the Sloan Foundation, and Visa Inc. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government or DARPA.

References

[ADN+10] Joël Alwen, Yevgeniy Dodis, Moni Naor, Gil Segev, Shabsi Walfish, and Daniel Wichs. Public-key encryption in the bounded-retrieval model. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 113–134, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-13190-5_6.

[ADW09] Joël Alwen, Yevgeniy Dodis, and Daniel Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 36–54, Santa Barbara, CA, USA, August 16–20, 2009. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-03356-8_3.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany. doi:10.1007/3-540-44647-8_13.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-54631-0_29.

[BHK11] Mark Braverman, Avinatan Hassidim, and Yael Tauman Kalai. Leaky pseudo-entropy functions. In Bernard Chazelle, editor, *ICS 2011: 2nd Innovations in Computer Science*, pages 353–366, Tsinghua University, Beijing, China, January 7–9, 2011. Tsinghua University Press.

[BK12] Zvika Brakerski and Yael Tauman Kalai. A parallel repetition theorem for leakage resilience. In Ronald Cramer, editor, *TCC 2012: 9th Theory of Cryptography Conference*, volume 7194 of *Lecture Notes in Computer Science*, pages 248–265, Taormina, Sicily, Italy, March 19–21, 2012. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-28914-9_14.

[BKR16] Mihir Bellare, Daniel Kane, and Phillip Rogaway. Big-key symmetric encryption: Resisting key exfiltration. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer*

Science, pages 373–402, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-53018-4_14.

[BL20] Fabrice Benhamouda and Huijia Lin. Mr NISC: Multiparty reusable non-interactive secure computation. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020: 18th Theory of Cryptography Conference, Part II*, volume 12551 of *Lecture Notes in Computer Science*, pages 349–378, Durham, NC, USA, November 16–19, 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-64378-2_13.

[CDD$^+$07] David Cash, Yan Zong Ding, Yevgeniy Dodis, Wenke Lee, Richard J. Lipton, and Shabsi Walfish. Intrusion-resilient key exchange in the bounded retrieval model. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 479–498, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-70936-7_26.

[CDG$^+$17] Chongwon Cho, Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, and Antigoni Polychroniadou. Laconic oblivious transfer and its applications. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 33–65, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-63715-0_2.

[CDRW10] Sherman S. M. Chow, Yevgeniy Dodis, Yannis Rouselakis, and Brent Waters. Practical leakage-resilient identity-based encryption from simple assumptions. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010: 17th Conference on Computer and Communications Security*, pages 152–161, Chicago, Illinois, USA, October 4–8, 2010. ACM Press. doi:10.1145/1866307.1866325.

[Cha12] Aldar C.-F. Chan. Distributed private key generation for identity based cryptosystems in ad hoc networks. *IEEE Wirel. Commun. Lett.*, 1(1):46–48, 2012. doi:10.1109/WCL.2012.120211.110130.

[CPM$^+$18] Giovanni Camurati, Sebastian Poeplau, Marius Muench, Tom Hayes, and Aurélien Francillon. Screaming channels: When electromagnetic side channels meet radio transceivers. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 163–177, Toronto, ON, Canada, October 15–19, 2018. ACM Press. doi:10.1145/3243734.3243802.

[CZLC16] Yu Chen, Zongyang Zhang, Dongdai Lin, and Zhenfu Cao. Generalized (identity-based) hash proof system and its applications. *Secur. Commun. Networks*, 9(12):1698–1716, 2016. doi:10.1002/sec.827.

[DFR$^+$07] Ivan Damgård, Serge Fehr, Renato Renner, Louis Salvail, and Christian Schaffner. A tight high-order entropic quantum uncertainty relation with applications. In Alfred Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 360–378, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-74143-5_20.

[DG17a] Nico Döttling and Sanjam Garg. From selective IBE to full IBE and selective HIBE. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 372–408, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-70500-2_13.

[DG17b] Nico Döttling and Sanjam Garg. Identity-based encryption from the Diffie-Hellman assumption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 537–569, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-63688-7_18.

[DGGM19] Nico Döttling, Sanjam Garg, Vipul Goyal, and Giulio Malavolta. Laconic conditional disclosure of secrets and applications. In David Zuckerman, editor, *60th Annual Symposium on Foundations of Computer Science*, pages 661–685, Baltimore, MD, USA, November 9–12, 2019. IEEE Computer Society Press. doi:10.1109/FOCS.2019.00046.

[DGHM18] Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, and Daniel Masny. New constructions of identity-based and key-dependent message secure encryption schemes. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 10769 of *Lecture Notes in Computer Science*, pages 3–31, Rio de Janeiro, Brazil, March 25–29, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-76578-5_1.

[DLW06] Giovanni Di Crescenzo, Richard J. Lipton, and Shabsi Walfish. Perfectly secure password protocols in the bounded retrieval model. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 225–244, New York, NY, USA, March 4–7, 2006. Springer, Heidelberg, Germany. doi:10.1007/11681878_12.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008. doi:10.1137/060651380.

[Dzi06a] Stefan Dziembowski. Intrusion-resilience via the bounded-storage model. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 207–224, New York, NY, USA, March 4–7, 2006. Springer, Heidelberg, Germany. doi:10.1007/11681878_11.

[Dzi06b] Stefan Dziembowski. On forward-secure storage (extended abstract). In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 251–270, Santa Barbara, CA, USA, August 20–24, 2006. Springer, Heidelberg, Germany. doi:10.1007/11818175_15.

[GHM$^+$19] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, Ahmadreza Rahimi, and Sruthi Sekar. Registration-based encryption from standard assumptions. In Dongdai Lin and Kazue Sako, editors, *PKC 2019: 22nd International Conference on Theory and*

Practice of Public Key Cryptography, Part II, volume 11443 of *Lecture Notes in Computer Science*, pages 63–93, Beijing, China, April 14–17, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-17259-6_3.

[GHMR18] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ahmadreza Rahimi. Registration-based encryption: Removing private-key generator from IBE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018: 16th Theory of Cryptography Conference, Part I*, volume 11239 of *Lecture Notes in Computer Science*, pages 689–718, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-03807-6_25.

[Goy07] Vipul Goyal. Reducing trust in the PKG in identity based cryptosystems. In Alfred Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 430–447, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-74143-5_24.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. doi:10.1137/S0097539793244708.

[HLWW13] Carmit Hazay, Adriana López-Alt, Hoeteck Wee, and Daniel Wichs. Leakage-resilient cryptography from minimal assumptions. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 160–176, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-38348-9_10.

[JP11] Abhishek Jain and Krzysztof Pietrzak. Parallel repetition for leakage resilience amplification revisited. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 58–69, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-19571-6_5.

[KG10] Aniket Kate and Ian Goldberg. Distributed private-key generators for identity-based cryptography. In Juan A. Garay and Roberto De Prisco, editors, *Security and Cryptography for Networks, 7th International Conference, SCN 2010, Amalfi, Italy, September 13-15, 2010. Proceedings*, volume 6280 of *Lecture Notes in Computer Science*, pages 436–453. Springer, 2010. doi:10.1007/978-3-642-15317-4_27.

[LRW11] Allison B. Lewko, Yannis Rouselakis, and Brent Waters. Achieving leakage resilience through dual system encryption. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 70–88, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-19571-6_6.

[LW10] Allison B. Lewko and Brent Waters. On the insecurity of parallel repetition for leakage resilience. In *51st Annual Symposium on Foundations of Computer Science*, pages 521–530, Las Vegas, NV, USA, October 23–26, 2010. IEEE Computer Society Press. doi:10.1109/FOCS.2010.57.

[MW20] Tal Moran and Daniel Wichs. Incompressible encodings. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 494–523, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-56784-2_17.

[NY19] Ryo Nishimaki and Takashi Yamakawa. Leakage-resilient identity-based encryption in bounded retrieval model with nearly optimal leakage-ratio. In Dongdai Lin and Kazue Sako, editors, *PKC 2019: 22nd International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 11442 of *Lecture Notes in Computer Science*, pages 466–495, Beijing, China, April 14–17, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-17253-4_16.