A preliminary version of this paper appears in *Advances in Cryptology – ASIACRYPT '09*, Lecture Notes in Computer Science Vol. 5912, pp. 232–249, Mitsuru Matsui ed., Springer-Verlag, 2009. This is the full version.

Hedged Public-Key Encryption: How to Protect Against Bad Randomness

Mihir Bellare[*] Zvika Brakerski[†] Moni Naor[‡] Thomas Ristenpart[§]
Gil Segev[¶] Hovav Shacham[||] Scott Yilek[**]

April 21, 2012

Abstract

Public-key encryption schemes rely for their IND-CPA security on per-message fresh randomness. In practice, randomness may be of poor quality for a variety of reasons, leading to failure of the schemes. Expecting the systems to improve is unrealistic. What we show in this paper is that we can, instead, improve the cryptography to offset the lack of possible randomness. We provide public-key encryption schemes that achieve IND-CPA security when the randomness they use is of high quality, but, when the latter is not the case, rather than breaking completely, they achieve a weaker but still useful notion of security that we call IND-CDA. This hedged public-key encryption provides the best possible security guarantees in the face of bad randomness. We provide simple RO-based ways to make in-practice IND-CPA schemes hedge secure with minimal software changes. We also provide non-RO model schemes relying on lossy trapdoor functions (LTDFs) and techniques from deterministic encryption. They achieve adaptive security by establishing and exploiting the anonymity of LTDFs which we believe is of independent interest.

[*] Department of Computer Science and Engineering, University of California, San Diego, USA. Supported in part by NSF grants CCF 0915675, CNS 0904380 and CNS 1116800. `http://cseweb.ucsd.edu/~mihir/`.

[†] Stanford University, Stanford, CA 94305, USA. Email: `zvika@stanford.edu`. This work was completed while the author was a Ph.D. student at the Weizmann Institute of Science.

[‡] Incumbent of the Judith Kleeman Professorial Chair, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel. Email: `moni.naor@weizmann.ac.il`. Research supported in part by a grant from the Israel Science Foundation.

[§] University of Wisconsin–Madison, Madison, WI 53715, USA. Email: `rist@cs.wisc.edu`. This work was completed while the author was a Ph.D. student at the University of California, San Diego.

[¶] Microsoft Research, Mountain View, CA 94043, USA. Email: `gil.segev@microsoft.com`. This work was completed while the author was a Ph.D. student at the Weizmann Institute of Science, and supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities.

[||] Department of Computer Science and Engineering, University of California, San Diego, USA. Supported in party by a MURI grant administered by the Air Force Office of Scientific Research.

[**] This work was completed while the author was a Ph.D. student at the University of California, San Diego supported by NSF grant CNS 0831536.

Contents

1 Introduction **3**

2 Preliminaries **6**

3 Attacks when Randomness is Bad **7**

4 Security against Chosen Distribution Attack **8**

4.1 Sources 8

4.2 Indistinguishability under chosen-distribution attack 9

4.3 IND-CDA for non-distinct sources 11

5 The Role of Anonymity in Adaptive CDA Security **12**

6 Constructions of Hedged Public-key Encryption **18**

6.1 Hedging in the Random Oracle Model 18

6.2 Hedging via Composition 25

6.3 Hedging by Randomizing Deterministic Encryption 27

6.4 RtD and PtD Without Universality 29

A Adaptive Variants of the Leftover Hash Lemma **32**

1 Introduction

Cryptography ubiquitously assumes that parties have access to sufficiently good randomness. In practice this assumption is often violated. This can happen because of faulty implementations, side-channel attacks, system resets or for a variety of other reasons. The resulting cryptographic failures can be spectacular [23, 25, 32, 2, 14]. What can we do about this? One answer is that system designers should build "better" systems, but this is clearly easier said than done. The reality is that random number generation is a complex and difficult task, and it is unrealistic to think that failures will never occur. We propose a different approach: designing schemes in such a way that poor randomness will have as little as possible impact on the security of the scheme in the following sense. With good randomness the scheme achieves whatever (strong) security notion one is targeting, but when the same scheme is fed bad (even adversarially chosen) randomness, rather than breaking completely, it achieves some weaker but still useful notion of security that is the best possible under the circumstances. We call this "hedged" cryptography.

Previous work by Rogaway [35], Rogaway and Shrimpton [36], and Kamara and Katz [29] considers various forms of hedging for the symmetric encryption setting. In this paper, we initiate a study of hedged public-key encryption. We address two central foundational questions, namely to find appropriate definitions and to efficiently achieve them. Let us now look at all this in more detail.

The problem. Achieving the standard IND-CPA notion of privacy [24] *requires* the encryption algorithm to be randomized. In addition to the public key and message, it takes as input a random string that needs to be freshly and independently created for *each and every* encryption.

Weak (meaning, low-entropy) randomness does not merely imply a loss of theoretical security. It can lead to catastrophic attacks. For example, weak-randomness based encryption is easily seen to allow recovery of the plaintext from the ciphertext for the quadratic residuosity scheme of [24] as well as the El Gamal encryption scheme [22]. Brown [14] presents such an attack on RSA-OAEP [9] with encryption exponent 3. Ouafi and Vaudenay [33] present such an attack on Rabin-SAEP [12]. We present an alternative attack in Section 3.

The above would be of little concern if we could guarantee good randomness. Unfortunately, this fails to be true in practice. Here, an "entropy-gathering" process is used to get a seed which is then stretched to get "random" bits for the application. The theory of cryptographically strong pseudorandom number generators [10] implies that the stretching can in principle be sound, and extractors further allow us to reduce the requirement on the seed from being uniformly distributed to having high min-entropy, but we still need a sufficiently good seed. (No amount of cryptography can create randomness out of nothing!) In practice, entropy might be gathered from timing-related operating system events or user keystrokes. As evidence that this process is error-prone, consider the recent randomness failure in Debian Linux, where a bug in the OpenSSL package led to insufficient entropy gathering and thence to practical attacks on the SSH [32] and SSL [2, 40] protocols. Other exploits include [26, 20].

The new notion. The idea is to provide two tiers of security. First, when the "randomness" is really random, the scheme should meet the standard IND-CPA notion of security. Otherwise, rather than failing completely, it should gracefully achieve some weaker but as-good-as-possible notion of security. The first important question we then face is to pick and formally define this fallback notion.

Towards this, we begin by suggesting that the *message* being encrypted may also have entropy or uncertainty from the point of view of the adversary. (If not, what privacy is there to be preserved by encryption?) We propose to harvest this. In this regard, the first requirement that might come to mind is that encryption with weak (even adversarially-known) randomness should be as secure as deterministic encryption, meaning achieve an analog of the PRIV notion of [6]. But achieving this would require that the message by itself have high min-entropy. We can do better. Our new target notion of security, that we call Indistinguishability under a Chosen Distribution Attack (IND-CDA), asks that security is guaranteed as long as the *joint* distribution of the message and randomness has sufficiently high min-entropy. In this

	Non-adaptive H-IND	Adaptive H-IND
REwH1	IND-CPA	IND-CPA + ANON-CPA
REwH2	IND-CPA	IND-CPA
RtD	IND-CPA, PRIV	IND-CPA, (u-)LTDF
PtD	(u-)LTDF	(u-)LTDF

Figure 1: Table entries for the first two rows indicate the assumptions made on the (randomized) encryption scheme that underlies the RO-model hedged schemes in question. The entries for standard model scheme RtD are the assumptions on the underlying randomized and deterministic encryption schemes, respectively, and for PtD, on the underlying deterministic encryption scheme, which is the only primitive it uses.

way, we can exploit for security whatever entropy might be present in the randomness *or* the message, and in particular achieve security even if neither taken alone is random enough.

Notice that if the message and randomness together have low min-entropy, then we cannot hope to achieve security, because an adversary can recover the message with high probability by trial encryption with all message-randomness pairs that occur with a noticeable probability. In a nutshell, our new notion asks that this necessary condition is also sufficient, and in this way is requiring security that is as good as possible.

We denote by H-IND our notion of *hedged* security that is satisfied by encryption schemes that are secure both in the sense of IND-CPA and in the sense of IND-CDA.

Adaptivity. Our IND-CDA definition generalizes the indistinguishability-style formalizations of PRIV-secure deterministic encryption [7, 11], which in turn extended entropic security [19]. But we consider a new dimension, namely, adaptivity. Our adversary is allowed to specify joint message-randomness distributions on to-be-encrypted challenges. The adversary is said to be adaptive if these queries depend on the replies to previous ones. Non-adaptive H-IND means IND-CPA plus non-adaptive IND-CDA and adaptive H-IND means IND-CPA plus adaptive IND-CDA.

Non-adaptive IND-CDA is a notion of security for randomized schemes that becomes identical to PRIV in the special case that the scheme is deterministic. Adaptive IND-CDA, when restricted to deterministic schemes, is an adaptive strengthening of PRIV that we think is interesting in its own right. As a consequence of the results discussed below, we get the first deterministic encryption schemes that achieve this stronger notion.

Schemes with random oracles. Our random oracle (RO) model schemes and their attributes are summarized in the first two rows of the table of Figure 1. Both REwH1 and REwH2 efficiently transform an arbitrary (randomized) IND-CPA scheme into a H-IND scheme with the aid of the RO. They are simple ways to make in-practice encryption schemes H-IND secure with minimal software changes. REwH1 has the advantage of not changing the public key and thus not requiring new certificates. It always provides non-adaptive H-IND security. It provides adaptive H-IND security if the starting scheme has the extra property of being anonymous in the sense of [4]. Anonymity is possessed by some deployed schemes like DHIES [1], making REwH1 attractive in this case. But some in-practice schemes, notably RSA ones, are not anonymous. If one wants adaptive H-IND security in this case we suggest REwH2, which provides it assuming only that the starting scheme is IND-CPA. It does this by adding a randomizer to the public key, so it does require new certificates. The schemes are extensions of the EwH deterministic encryption scheme of [6] and similar to [21].

Schemes without random oracles. It is easy to see that even the existence of a non-adaptively secure IND-CDA encryption scheme implies the existence of a PRIV-secure deterministic encryption (DE) scheme. Achieving PRIV without ROs is already hard. Indeed, fully PRIV-secure DE without ROs has not yet been built. Prior work, however, does show how to construct PRIV-secure DE without ROs for block sources [11]. (Messages being encrypted have high min-entropy even conditioned on previous messages.) But H-IND introduces three additional challenges: (1) the min-entropy guarantee is on the joint message-randomness distribution rather than merely on the message; (2) we want a single scheme that is not only IND-CDA secure but also IND-CPA-secure; and (3) the adversary's queries may be adaptive.

We are able to overcome these challenges to the best extent possible. We provide schemes that are H-IND-secure in the same setting as the best known PRIV ones, namely, for block sources, where we suitably extend the latter notion to consider both randomness and messages. Furthermore, we achieve these results under the same assumptions as previous work.

Our standard model schemes and their attributes are summarized in the last two rows of the table of Figure 1. RtD is formed by the generic composition of a deterministic scheme and a randomized scheme and achieves non-adaptive H-IND security as long as the base schemes meet their regular conditions. (That is, the former is PRIV-secure for block sources and the latter is IND-CPA.) Adaptive security requires that the deterministic scheme be a u-LTDF. (A lossy trapdoor function whose lossy branch is a universal hash function [34, 11].) PtD is simpler, merely concatenating the message to the randomness and then applying deterministic encryption. It achieves both non-adaptive and adaptive H-IND under the assumption that the deterministic scheme is a u-LTDF. For both schemes, the universality assumption on the LTDF can be dropped by modifying the scheme and using the crooked leftover hash lemma as per [11]. (This is why the "u" is parenthesized in the table of Figure 1.)

Anonymous LTDFs. Also of independent interest, we show that any u-LTDF is anonymous. Here we refer to a new notion of anonymity for trapdoor functions that we introduce, one that strengthens the notion of [4]. This step exploits an adaptive variant of the leftover hash lemma of [28].

Why anonymity? It is exploited in our proofs of adaptive security. Our new notion of anonymity for trapdoor functions is matched by a corresponding one for encryption schemes. We show that any encryption scheme that is both anonymous and non-adaptive H-IND secure is also adaptively H-IND secure. Anonymity of the u-LTDF, in our encryption schemes based on the latter primitive, allows us to show that these schemes are anonymous and thereby lift their non-adaptive security to adaptive.

Related work. In the symmetric setting, several works have recognized and addressed the problem of security in the face of bad randomness. Concern over the quality of available randomness is one of Rogaway's motivations for introducing nonce-based symmetric encryption [35], where security relies on the nonce never repeating rather than being random. Rogaway and Shrimpton [36] provide a symmetric authenticated encryption scheme that defaults to a PRF when the randomness is known.

Kamara and Katz [29] provide symmetric encryption schemes secure against chosen-randomness attack (CRA). Here the adversary can obtain encryption under randomness of its choice but privacy is only required for messages encrypted with perfect, hidden randomness. Entropy in the messages is not considered or used. We in contrast seek privacy even when the randomness is bad as long as there is compensating entropy in the message. Also we deal with the public key setting.

Many works consider achieving strong cryptography given only a "weak random source" [31, 17, 13]. This is a source that does have high min-entropy but may not produce truly random bits. They show that many cryptographic tasks including symmetric encryption [31], commitment, secret-sharing, and zero knowledge [17] are impossible in this setting. We are not in this setting. We do assume a small amount of initial good randomness to produce keys. (This makes sense because it is one-time and because otherwise we can't hope to achieve anything anyway.) On the other hand our assumption on the randomness available for encryption is even weaker than in the works mentioned. (We do not even assume it has high min-entropy.) Our key idea is to exploit the entropy in the message, which is not done in [31, 17, 13]. This allows us to circumvent their negative results.

Waters independently proposed hedge security as well as the PtD construction as a way to achieve it [39].

We should note that the term hedging was previously used by Shoup to describe an encryption scheme that is simultaneously provably secure under one set of assumptions in the random oracle model and provably secure under a (stronger) set of assumptions in the standard model [38].

2 Preliminaries

Notation. Vectors are written in boldface, e.g. $\mathbf{x}$. If $\mathbf{x}$ is a vector then $|\mathbf{x}|$ denotes its length and $\mathbf{x}[i]$ denotes its i^{th} component for $1 \leq i \leq |\mathbf{x}|$. We say that $\mathbf{x}$ is a vector over D if $\mathbf{x}[i] \in D$ for all $1 \leq i \leq |\mathbf{x}|$. Throughout, $k \in \mathbb{N}$ denotes the security parameter and 1^k its unary encoding. Unless otherwise indicated, an algorithm is randomized. The set of possible outputs of algorithm A on inputs $x_1, x_2, \ldots$ is denoted $[A(x_1, x_2, \ldots)]$. "PT" stands for polynomial-time.

Games. Our security definitions and proofs use code-based games [8], and so we recall some background from [8]. A game (look at Figure 2 for examples) has an **Initialize** procedure, procedures to respond to adversary oracle queries, and a **Finalize** procedure. A game G is executed with an adversary A as follows. First, **Initialize** executes, and its outputs are the inputs to A. Then A executes, its oracle queries being answered by the corresponding procedures of G. When A terminates, its output becomes the input to the **Finalize** procedure. The output of the latter is called the output of the game, and we let $\mathrm{G}^A \Rightarrow y$ denote the event that this game output takes value y. Our convention is that the running time of an adversary is the time to execute the adversary with the game that defines security, so that the running time of all game procedures is included.

Public-key encryption. A public-key encryption (PKE) scheme is a tuple of PT algorithms $\mathcal{AE} = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ with associated message length parameter $n(\cdot)$ and randomness length parameter $\rho(\cdot)$. The parameter generation algorithm $\mathcal{P}$ takes as input 1^k and outputs a parameter string par. The key generation algorithm $\mathcal{K}$ takes input par and outputs a key pair (pk, sk). The encryption algorithm $\mathcal{E}$ takes inputs pk, message $m \in \{0,1\}^{n(k)}$ and coins $r \in \{0,1\}^{\rho(k)}$ and returns the ciphertext denoted $\mathcal{E}(pk, m\,;\, r)$. The deterministic decryption algorithm $\mathcal{D}$ takes input sk and ciphertext c and outputs either $\perp$ or a message in $\{0,1\}^{n(k)}$. For vectors $\mathbf{m}, \mathbf{r}$ with $|\mathbf{m}| = |\mathbf{r}| = v$ we denote by $\mathcal{E}(pk, \mathbf{m}\,;\, \mathbf{r})$ the vector $(\mathcal{E}(pk, \mathbf{m}[1]\,;\, \mathbf{r}[1]), \ldots, \mathcal{E}(pk, \mathbf{m}[v]\,;\, \mathbf{r}[v]))$. We say that $\mathcal{AE}$ is deterministic if $\mathcal{E}$ is deterministic. (That is, $\rho(\cdot) = 0$.)

We consider the standard IND-CPA notion of security, captured by the game $\mathrm{IND}_{\mathcal{AE}}$ where $\mathcal{AE} = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ is an encryption scheme. In the game, **Initialize** chooses a random bit b, generates parameters $par \leftarrow\!\!\$\; \mathcal{P}(1^k)$ and generates a key pair $(pk, sk) \leftarrow\!\!\$\; \mathcal{K}(par)$ before returning pk to the adversary. Procedure **LR**, on input messages m_0 and m_1, returns $c \leftarrow\!\!\$\; \mathcal{E}(pk, m_b)$. Lastly, procedure **Finalize** takes as input a guess bit b' and outputs true if $b = b'$ and false otherwise. An IND-CPA adversary makes zero or more queries (m_0, m_1) to **LR** with $|m_0| = |m_1|$. Since a simple hybrid shows that allowing a single query is sufficient, we will unless otherwise noted restrict attention to the case where adversaries make a single query. For IND-CPA adversary A we let $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},A}(k) = 2 \cdot \Pr\left[\,\mathrm{IND}^A_{\mathcal{AE},k} \Rightarrow \mathsf{true}\,\right] - 1$. We say $\mathcal{AE}$ is IND-CPA secure if $\mathbf{Adv}^{\text{ind}}_{\mathcal{AE},A}(\cdot)$ is negligible for all PT IND-CPA adversaries A.

Following [6], for any k we define the maximum public-key collision probability by

$$\mathsf{maxpk}_{\mathcal{AE}}(k) = \max_{w \in \{0,1\}^*} \Pr\left[\, pk = w \;:\; par \leftarrow\!\!\$\; \mathcal{P}(1^k)\,;\, (pk, sk) \leftarrow\!\!\$\; \mathcal{K}(par) \,\right] .$$

Universal hash functions. A family of functions is a tuple of algorithms $\mathcal{H} = (\mathcal{P}, \mathcal{K}, F)$ with associated message length $n(\cdot)$. It is required that the domain of $F(K, \cdot)$ is $\{0,1\}^n$ for every k, every $par \in [\mathcal{P}(1^k)]$, and every $K \in [\mathcal{K}(par)]$. We say that $\mathcal{H}$ is universal if for every k, all $par \in [\mathcal{P}(1^k)]$, and all distinct $x_1, x_2 \in \{0,1\}^{n(k)}$, the probability that $F(K, x_1) = F(K, x_2)$ is at most $1/|R(par)|$ where $R(par) = \{\, F(K, x) \,:\, K \in [\mathcal{K}(par)] \text{ and } x \in \{0,1\}^n \,\}$ and the probability is over $K \leftarrow\!\!\$\; \mathcal{K}(par)$. Similarly, we say $\mathcal{H}$ is pairwise-independent if for every k, all $par \in [\mathcal{P}(1^k)]$, all distinct $x_1, x_2 \in \{0,1\}^{n(k)}$, and all $y_1, y_2 \in R(par)$, the probability that $F(K, x_1) = y_1 \wedge F(K, x_2) = y_2$ is at most $1/|R(par)|^2$, where again the probability is over $K \leftarrow\$ \mathcal{K}(par)$.

We say a family $\mathcal{H}$ has 2^t-bounded range if for all k, all $par \in \mathcal{P}(1^k)$, $|R(par)| \leq 2^{t(k)}$.

We say a family $\mathcal{H}$ is efficiently invertible if there is an efficient algorithm F^{-1} such that for all 1^k, all $par \in [\mathcal{P}(1^k)]$, all $K \in [\mathcal{K}(par)]$, and all $x \in \{0,1\}^{n(k)}$ it is the case that $F^{-1}(K, F(K,x)) = x$.

Lossy Trapdoor Functions (LTDFs). To a deterministic PKE scheme (recall that a family of injective trapdoor functions and a deterministic encryption scheme are, syntactically, the same object) $\mathcal{AE} = (\mathcal{P}_\mathrm{d}, \mathcal{K}_\mathrm{d}, \mathcal{E}_\mathrm{d}, \mathcal{D}_\mathrm{d})$ with message length $n_\mathrm{d}(\cdot)$ we can associate an (n_d, ℓ)-lossy key generator $\mathcal{K}_l$. This is a PT algorithm that, on input par, outputs a value pk for which the map $\mathcal{E}_\mathrm{d}(pk, \cdot)$ has image size at most $2^{n_\mathrm{d}(k)-\ell(k)}$. The parameter ℓ is called the lossiness of the lossy key generator. We associate to $\mathcal{AE}$, lossy key generator $\mathcal{K}_l$, and a LOS adversary A the function $\mathbf{Adv}^{\mathrm{los}}_{\mathcal{AE},\mathcal{K}_l,A}(k) = 2 \cdot \Pr\left[\, \mathrm{LOS}^A_{\mathcal{AE},\mathcal{K}_l,k} \Rightarrow \mathsf{true} \,\right] - 1$, where game $\mathrm{LOS}_{\mathcal{AE},\mathcal{K}_l}$ works as follows. **Initialize** chooses a random bit b and generates parameters $par \leftarrow\$ \mathcal{P}_\mathrm{d}(1^k)$, if $b = 0$ runs $(pk, sk) \leftarrow\$ \mathcal{K}_\mathrm{d}(par)$ and if $b = 1$ runs $pk \leftarrow\$ \mathcal{K}_l(par)$. It then returns pk (to the adversary A). When A finishes, outputting guess b', **Finalize** returns true if $b = b'$. We say $\mathcal{K}_l$ is *universal-inducing* if $\mathcal{H} = (\mathcal{P}_\mathrm{d}, \mathcal{K}_l, \mathcal{E}_\mathrm{d})$ is a family of universal hash functions with message length n_d.

A deterministic encryption scheme $\mathcal{AE}$ is a (n_d, ℓ)-lossy trapdoor function (LTDF) if there exists a (n_d, ℓ)-lossy key generator such that $\mathbf{Adv}^{\mathrm{los}}_{\mathcal{AE},\mathcal{K}_l,A}(\cdot)$ is negligible for all PT A. We say it is a universal (n_d, ℓ)-lossy trapdoor function (u-LTDF) if in addition $\mathcal{K}_l$ is universal-inducing.

Lossy trapdoor functions were introduced by Peikert and Waters [34], and can be based on a variety of number-theoretic assumptions, including the hardness of the decisional Diffie-Hellman problem, the worst-case hardness of lattice problems, and the hardness of Paillier's composite residuosity problem [34, 11, 37]. Boldyreva et al. [11] observed that the DDH-based construction is universal.

3 Attacks when Randomness is Bad

The traditional security model of IND-CPA for PKE schemes, given in the last section, mandates good per-message randomness. In this section we highlight the catastrophic attacks that can occur when randomness is bad. Consider encryption $\mathcal{E}(pk, m\,;\,r)$ of a message m under public key pk and randomness r. For the attacks discussed below we assume that the random number generator is broken but not necessarily under adversarial control (as was the case in the Debian vulnerability [32] and other weak PRNG vulnerabilities [2, 26, 20]). Broken means the value r is predictable by the adversary (technically, has little or even no min-entropy). Our eventual security definitions will make no such simplifying assumption and will instead ask that our schemes achieve (the best possible) security even in the face of adversarially-subverted random number generators.

Plaintext recovery attacks. Many prominent PKE schemes are vulnerable to fast plaintext recovery attacks when randomness is predictable. As mentioned in the introduction both El Gamal encryption [22] and Goldwasser-Micali encryption [24] are vulnerable. For the former, encryption under public key X is $\mathcal{E}(X, M\,;\,r) = (g^r, X^r{\cdot}M)$, so the ability to predict r immediately gives X^r and leads to message recovery. The Goldwasser-Micali scheme fails analogously.

One can utilize Coppersmith's method in the univariate case [16, 27] to recover plaintexts from Rabin-SAEP [12] ciphertexts when randomness is known. The Rabin-SAEP padding function [12] for m-bit message M and s_1-bit randomness r is $\big((m \,\|\, 0^{s_0}) \oplus H(r)\big) \,\|\, r$, where H is a random oracle mapping s_1-bit strings to $(m + s_0)$-bit strings; the bit sizes must satisfy $m < n/4$ and $m + s_0 < n/2$, where n is the bitlength of the Rabin modulus N. For a ciphertext C whose randomness r is known, we can write $f(x) = (x \cdot 2^{s_0+s_1} + a)^2 - C$ where a is known. There exists a small root x_0 of $f(x) \bmod N$: one such that $x_0 < 2^{n/4}$; computing this root reveals the plaintext M. (Specifically, let $H(r) = h_L \,\|\, h_R$, where h_L is m bits long and h_R is s_0 bits long. Then $a = h_R \,\|\, r$ and $x_0 = M \oplus h_L$.) By Coppersmith's method in the univariate case [16, 27] it is possible to find a root to a degree-δ polynomial modulo N if that root is smaller than $N^{1/\delta}$, which the parameters here easily satisfy. Thus a single ciphertext with known (not necessarily adversarially-generated) randomness suffices to leak the plaintext. In fact, this is used crucially in the proof of security of Rabin-SAEP to handle decryption queries. We note that a recent work by Ouafi and Vaudenay [33] against the SQUASH-0 hash function also gives a "known-coins" message recovery attack against Rabin-SAEP.

Brown [14] gives an attack against RSA-OAEP [9] with $e = 3$ and known randomness. The attack is based on Coppersmith's method and is essentially the same as the one we described above against Rabin-SAEP. One difference is that exponentiation by $e = 3$ yields a cubic polynomial, reducing the size of the small roots that can be extracted; another is that OAEP padding includes two Feistel rounds instead of one so there are two unknowns, which means that the attack is only heuristic.

All hybrid encryption (KEM/DEM) schemes are vulnerable to plaintext recovery when randomness is predictable, which is unfortunate due to the wide use of hybrid encryption in practice. Briefly hybrid encryption $\mathcal{E}(pk, M\,;\,r)$ first runs a key encapsulation routine $(c_1, K) \leftarrow \psi(pk\,;\,r)$ and then encrypts the message via symmetric encryption $c_2 \leftarrow E(K, M)$. The full ciphertext is (c_1, c_2). If r is predictable, then an adversary can run $\psi(pk\,;\,r)$ itself to recompute K and use it to recover the plaintext. Note the structure of some KEM/DEMs is such that even when r is not predictable, but even just re-used, then attacks exist. Consider when the symmetric encryption is CTR-mode encryption. Then encrypting two messages m and m' under the same randomness r would immediately reveal $m \oplus m'$ to the adversary.

Ciphertexts leak plaintext, randomness equality. As pointed out in the introduction, there exists an inherent insecurity for any PKE scheme when both messages and randomness are predictable. Given a ciphertext $\mathcal{E}(pk, m\,;\,r)$ the adversary can easily determine the message via a trial-encryption brute-force attack. Thus, any message-privacy security notion for PKE when randomness is bad will require that the pair (m, r) has high min-entropy.

4 Security against Chosen Distribution Attack

When randomness may be bad, traditional notions such as IND-CPA are no longer achievable. We therefore formalize a new security goal to complement IND-CPA: indistinguishability under chosen distribution attack. The adversary attempts to learn partial information about challenge messages when the message and randomness are together sampled from an unpredictable source.

4.1 Sources

We generalize the notion of a source to consider a joint distribution on the messages and the randomness with which they will be encrypted. A t-source ($t \geq 1$) with vector length $v(\cdot)$, message length $n(\cdot)$, and randomness length $\rho(\cdot)$ is a probabilistic algorithm $\boldsymbol{\mathcal{M}}$ that on input 1^k returns a $(t+1)$-tuple $(\mathbf{m}_0, \ldots, \mathbf{m}_{t-1}, \mathbf{r})$. The vectors $\mathbf{m}_0, \ldots, \mathbf{m}_{t-1}$ each have $v(k)$ elements in $\{0,1\}^{n(k)}$ and $\mathbf{r}$ has $v(k)$ elements in $\{0,1\}^{\rho(k)}$. We say that $\boldsymbol{\mathcal{M}}$ has min-entropy $\mu(\cdot)$ if

$$\Pr\left[\, (\mathbf{m}_b[i], \mathbf{r}[i]) = (m, r) \,\right] \leq 2^{-\mu(k)}$$

for all $k \in \mathbb{N}$, all $b \in \{0, \ldots, t-1\}$, all $i \in \{1, \ldots, |\mathbf{r}|\}$, all $(m, r) \in \{0,1\}^{n(k)} \times \{0,1\}^{\rho(k)}$, and where the probability is over the coins used to run $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\!\!\$\; \boldsymbol{\mathcal{M}}(1^k)$. We say it has conditional min-entropy $\mu(\cdot)$ if

$$\Pr\left[\, (\mathbf{m}_b[i], \mathbf{r}[i]) = (m, r) \;\middle|\; \forall j < i\ (\mathbf{m}_b[j], \mathbf{r}[j]) = (\mathbf{m}'[j], \mathbf{r}'[j]) \,\right] \leq 2^{-\mu(k)}$$

for all $k \in \mathbb{N}$, all $b \in \{0, \ldots, t-1\}$, all i, all (m, r), all vectors $\mathbf{m}', \mathbf{r}'$, and over the coins used by $\boldsymbol{\mathcal{M}}$. In the random oracle (RO) model, message sources have access to the RO. In this setting, the (conditional) min-entropy requirement is independent of the coins used by the RO, meaning the bound must hold for any fixed choice of function as the RO.

```
procedure Initialize($1^k$):
    $par \leftarrow\$ \mathcal{P}(1^k)$
    $(pk, sk) \leftarrow\$ \mathcal{K}(par)$
    $b \leftarrow\$ \{0,1\}$
    Ret $par$

procedure LR($\boldsymbol{\mathcal{M}}$):
    If $\mathsf{pkout} = \mathsf{true}$ then
        Ret $\perp$
    $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ \boldsymbol{\mathcal{M}}(1^k)$
    Ret $\mathcal{E}(pk, \mathbf{m}_b; \mathbf{r})$

procedure RevealPK():
    $\mathsf{pkout} \leftarrow \mathsf{true}$
    Ret $pk$

procedure Finalize($b'$):
    Ret $(b = b')$
```

Figure 2: Game $\mathrm{CDA}_{\mathcal{AE},k}$

For any pair of vectors $(\mathbf{m}, \mathbf{r})$ of length v, we define the equality pattern of $(\mathbf{m}, \mathbf{r})$ to be the bit-valued matrix $E^{(\mathbf{m},\mathbf{r})}$ with v rows and v columns for which $E^{(\mathbf{m},\mathbf{r})}_{i,j} = 1$ if $(\mathbf{m}[i], \mathbf{r}[i]) = (\mathbf{m}[j], \mathbf{r}[j])$ and $E^{(\mathbf{m},\mathbf{r})}_{i,j} = 0$ otherwise for $1 \le i \le j \le v$. This always-symmetric matrix describes the equality relations between all elements of the two vectors. A *distinct* t-source $\boldsymbol{\mathcal{M}}$ with vector length $v(\cdot)$ is one for which

$$\Pr\left[\, E^{(\mathbf{m}_b,\mathbf{r})} = I_{v(k)} \ :\ (\mathbf{m}_0, \ldots, \mathbf{m}_{t-1}, \mathbf{r}) \leftarrow\$ \boldsymbol{\mathcal{M}}(1^k) \,\right] = 1$$

for all $k \in \mathbb{N}$ and all $b \in \{0, \ldots, t-1\}$ and where $I_{v(k)}$ denotes the $v(k)$ by $v(k)$ identity matrix.

We fix some notation for referring to commonly used types of sources. A t-source with vector length $v(\cdot)$, message length $n(\cdot)$, randomness length $\rho(\cdot)$, and min-entropy $\mu(\cdot)$ is referred to as

- a (μ, v, n, ρ)-mr-source when $t = 1$ and $\rho(\cdot) > 0$;
- a (μ, v, n)-m-source when $t = 1$ and $\rho(\cdot) = 0$;
- a (μ, v, n, ρ)-mmr-source when $t = 2$ and $\rho(\cdot) > 0$; and
- a (μ, v, n)-mm-source when $t = 2$ and $\rho(\cdot) = 0$.

Each “m” indicates the source outputting one message vector and an “r” indicates a randomness vector. When the source has *conditional* min-entropy $\mu(\cdot)$ we write block-source instead of source for each of the above.

4.2 Indistinguishability under chosen-distribution attack

Let $\mathcal{AE} = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. A CDA adversary is one whose **LR** queries are all mmr-sources. Game $\mathrm{CDA}_{\mathcal{AE}}$ of Figure 2 provides the adversary with two oracles. The advantage of CDA adversary A is

$$\mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},A}(k) = 2 \cdot \Pr\left[\, \mathrm{CDA}^{A}_{\mathcal{AE},k} \Rightarrow \mathsf{true} \,\right] - 1 \ .$$

In the random oracle model we allow all algorithms in Game CDA to access the random oracle; importantly, this includes the mmr-sources.

Discussion. Adversary A can query **LR** with an mmr-source of its choice, an output $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r})$ of which represents choices of message vectors to encrypt and randomness with which to encrypt them. (An alternative formulation might have CDA adversaries query two mr-sources, and distinguish between the encryption of samples taken from one of these. But this would mandate that schemes ensure privacy of messages *and* randomness.) This allows A to dictate a joint distribution on the messages and randomness. In this way it conservatively models even adversarially-subverted random number generators. Multiple **LR** queries are allowed. In the most general case these queries may be adaptive, meaning depend on answers to previous queries.

Given that multiple **LR** queries are allowed, one may ask why an mmr-source needs to produce message and randomness vectors rather than simply a single pair of messages and a single choice of randomness. The reason is that the coordinates in a vector all depend on the same coins underlying an execution of $\boldsymbol{\mathcal{M}}$, but the coins underlying the execution of the sources in different queries are independent.

Note that **Initialize** does not return the public key pk to A. A can get it at any time by calling **RevealPK** but once it does this, **LR** will return $\perp$. The reason is that we inherit from deterministic encryption the unavoidable limitation that encryption cannot hide public-key related information about the plaintexts [6]. (When the randomness has low entropy, the ciphertext itself is such information.)

As we saw in the previous section, no encryption scheme is secure when both messages and randomness are predictable. Formally, this means chosen-distribution attacks are trivial when adversaries can query mmr-sources of low min-entropy. Our notions (below) will therefore require security only for sources that have high min-entropy or high conditional min-entropy.

Similarly, we inherit from deterministic encryption the unavoidable limitation that we cannot allow arbitrary equality patterns. For simplicity we have restricted attention to adversaries that query distinct sources. These always output message, randomness pairs that are distinct. A detailed discussion of the role of equality patterns is given in Section 4.3.

Notions. We can assume (without loss of generality) that a CDA adversary makes a single **RevealPK** query and then no further **LR** queries. We say A is a (μ, v, n, ρ)-adversary if all of its **LR** queries are distinct (μ, v, n, ρ)-mmr-sources. We say that a PKE scheme $\mathcal{AE}$ with message length $n(\cdot)$ and randomness length $\rho(\cdot)$ is IND-CDA secure for distinct (μ, v, n, ρ)-mmr-sources if for all PT (μ, v, n, ρ) adversaries A the function $\mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},A}(\cdot)$ is negligible. Scheme $\mathcal{AE}$ is H-IND secure for distinct (μ, v, n, ρ)-mmr-sources if it is IND-CPA secure and IND-CDA secure for (μ, v, n, ρ)-mmr-sources. We can extend these notions to distinct mmr-block-sources by restricting to adversaries that query distinct mmr-block-sources. Theorem 4.1 below allows one to generalize from distinct sources to sources with other equality patterns.

On adaptivity. We can consider non-adaptive IND-CDA security by restricting attention in the notions above to adversaries that only make a single **LR** query. Why do we not focus solely on this (simpler) security goal? The standard IND-CPA setting (implicitly) provides security against multiple, adaptive **LR** queries. This is true because in that setting a straightforward hybrid argument shows that security against multiple adaptive **LR** queries is implied by security against a single **LR** query [5, 3]. We wish to maintain the same standard of adaptive security in the IND-CDA setting. Unfortunately, in the IND-CDA setting, unlike the IND-CPA setting, adaptive security is not implied by non-adaptive security. In short this is because a CDA adversary necessarily cannot learn the public key before (or while) making **LR** queries. To see the separation, consider a PKE scheme that appends to every ciphertext the public key used. This will not affect the security of the scheme when an adversary can only make a single query. However, an adaptive CDA adversary can query an mmr-source, learn the public key, and craft a second source that uses the public key to ensure ciphertexts which leak the challenge bit.

Given this, our primary goal is the stronger notion of adaptive security. That said, non-adaptive hedge security is also relevant because in practice adaptive adversaries might be rare and, as we will see, one can find non-adaptively-secure schemes that are more efficient and/or have proofs under weaker assumptions.

Adaptive PRIV. A special case of our framework occurs when the PKE scheme $\mathcal{AE}$ being considered has randomness length $\rho(k) = 0$ for all k (meaning also that adversaries query mm-sources, instead of mmr-sources). In this case we are considering deterministic encryption, and the IND-CDA definition and notions give a strengthening (by way of adaptivity) of the PRIV security notion from [6, 7, 11]. (For non-adaptive adversaries the definitions are equivalent.) For clarity we will use PRIV to refer to this special case, and let $\mathbf{Adv}^{\mathrm{priv}}_{\mathcal{AE},A}(k) = \mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},A}(k)$.

Resource usage. Recall that by our convention, the running time of a CDA adversary is the time for the execution of the adversary with game $\mathrm{CDA}_{\mathcal{AE},k}$. Thus, A being PT implies that the mmr-sources that comprise A's **LR** queries are also PT. This is a distinction from [11] which will be important in our results. Note that in practice we do not expect to see sources that are not PT, so our definition is not restrictive. Non-PT sources were needed in [11] for showing that single-message security implied (non-adaptive) multi-message security for deterministic encryption of block sources.

4.3 IND-CDA for non-distinct sources

Above we restricted the IND-CDA security notion to consider only attackers that query *distinct* t-sources. These are sources that output vectors that have the identity equality pattern. Here we investigate relaxations of this requirement, showing that any achievable relaxation is implied by security against distinct sources.

Unachievable notions. We start with having no restrictions on equality patterns entirely. A relaxed source IND-CDA adversary is one whose **LR** queries are all mmr-sources, these not necessarily being distinct. Then, as in the deterministic encryption setting [6], we have that no encryption scheme can be hedge secure against such adversaries. Let A be the adversary that makes a query $\boldsymbol{\mathcal{M}}$ which returns $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) = ((a,a),(a,a'),(r,r))$ for some $a \neq a'$ and random r. Then A can win trivially because the (two) components of the returned vector $\mathbf{c}$ are equal if $b = 0$ and unequal otherwise. This example points to a fundamental limitation with encryption: equality of plaintext and randomness is leaked by ciphertexts. To have an acheivable notion of security, then, we must ensure that CDA adversaries cannot use plaintext-randomness equalities in order to trivially learn the challenge bit b.

Recall the equality pattern definition from the last section. The equality patterns for the pairs of vectors $((a,a),(r,r))$ and $((a,a'),(r,r))$ used in the attack of the last paragraph are

$$E^{(\mathbf{m}_0,\mathbf{r})} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad \text{and} \quad E^{(\mathbf{m}_1,\mathbf{r})} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} .$$

In that example, the adversary takes advantage of the fact that $E^{(\mathbf{m}_0,\mathbf{r})} \neq E^{(\mathbf{m}_1,\mathbf{r})}$. We must exclude such "trivial" adversaries by restricting attention to adversaries that only query sources $\boldsymbol{\mathcal{M}}$ that do not leak information via equality-patterns. One might therefore be tempted to just enforce that equality patterns do not leak anything about b directly. This can be captured by requiring that any mmr-source outputs vectors $\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}$ such that $E^{(\mathbf{m}_0,\mathbf{r})} = E^{(\mathbf{m}_1,\mathbf{r})}$ holds with high probability. However, this relaxation is still trivial to win against: an attacker can choose $\boldsymbol{\mathcal{M}}$ so that the equality pattern encodes (say) all the bits that are common between the first messages of $\mathbf{m}_0$ and $\mathbf{m}_1$.

An achievable notion. We now give a restriction that is sufficient to bar trivial adversaries. A t-source $\boldsymbol{\mathcal{M}}$ has equality-pattern respect $\zeta(\cdot)$ if there exists a family of reference equality-patterns $\{\hat{E}^{v(k)}\}_{k\in\mathbb{N}}$ such that

$$\Pr\left[\bigvee_b E^{(\mathbf{m}_b,\mathbf{r})} \neq \hat{E}^{v(k)} \;:\; (\mathbf{m}_0,\ldots,\mathbf{m}_{t-1},\mathbf{r}) \leftarrow\$\; \boldsymbol{\mathcal{M}}(1^k) \right] \leq 2^{-\zeta(k)} \qquad (1)$$

for all $k \in \mathbb{N}$ and all $b \in \{0,\ldots,t-1\}$. In the ROM the probability above must hold with respect to any fixed RO (i.e., the probability is taken over just the coins used by $\boldsymbol{\mathcal{M}}$ directly.) An IND-CDA adversary has equality-pattern respect $\zeta(\cdot)$ if all mmr-sources it queries have equality-pattern respect at least $\zeta(\cdot)$. Intuitively, as long as $\zeta(k)$ is large enough for every k of interest, the equality pattern cannot leak any information to the attacker — it is almost always certainly some fixed equality pattern. We note that with probability related to their conditional min-entropy, block sources already output vectors whose equality pattern is the identity matrix.

The following lemma shows that the fixed equality pattern might as well be the identity equality pattern. In other words, the relaxation to non-distinct sources above is equivalent to security against distinct sources.

Theorem 4.1 Let $\mathcal{AE} = (\mathcal{P},\mathcal{K},\mathcal{E},\mathcal{D})$ be an encryption scheme with message length $n(\cdot)$ and randomness length $\rho(\cdot)$. Let A be a IND-CDA adversary making $q(\cdot)$ **LR** queries, each being a (μ, v, n, ρ)-mmr-source with equality-pattern respect at least $\zeta(\cdot)$. Then there exists IND-CDA adversary B such that for all k

$$\mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},A}(k) \leq \mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},B}(k) + \frac{4q(k)}{2^{\zeta(k)}} .$$

B makes $q(\cdot)$ **LR** queries, each being a distinct (μ, v', n, ρ)-mmr-source with $v' \leq v$. Adversary B runs in at most twice the running time of A. $\square$

Proof: Fix any k and let $v = v(k)$, $q = q(k)$, and $\zeta = \zeta(k)$. Let adversary B work as follows. It runs A, outputs the same bit output by A, and responds to **LR** queries as follows.

Let $\boldsymbol{\mathcal{M}}$ be an mmr-source queried by A to **LR**. Then B runs $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\!\!{\scriptstyle\$}\; \boldsymbol{\mathcal{M}}$ and computes the equality pattern $\hat{E} = E^{(\mathbf{m}_0,\mathbf{r})}$. Adversary B derives from $\hat{E}$ a vector $\mathbf{p}$ of size v, defined as follows. Let $c = 1$. Then for $j = 1$ to v do the following. Let $i \leq j$ be the least value i such that $\hat{E}_{i,j} = 1$. If $i = j$ then increment c and let $\mathbf{p}[j] = c$. Otherwise, let $\mathbf{p}[j] = \mathbf{p}[i]$. It sets v' to be the final value of c, which is the number of distinct message, randomness pairs in $(\mathbf{m}_0, \mathbf{r})$. The vector $\mathbf{p}$ keeps track of which message, randomness pairs are (with high probability) duplicates of others for the source $\boldsymbol{\mathcal{M}}$.

Adversary B then defines a distinct mmr-source $\boldsymbol{\mathcal{M}}'$ that works as follows. It runs $\boldsymbol{\mathcal{M}}$ to get vectors $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r})$. It then defines vectors $(\mathbf{m}'_0, \mathbf{m}'_1, \mathbf{r}')$ as follows. For each $1 \leq j \leq v$, it lets $i = \mathbf{p}[j]$ and sets $(\mathbf{m}'_0[i], \mathbf{m}'_1[i], \mathbf{r}'[i]) = (\mathbf{m}_0[j], \mathbf{m}_1[j], \mathbf{r}[j])$. Then, it outputs the vectors $(\mathbf{m}'_0, \mathbf{m}'_1, \mathbf{r}')$. By construction $|\mathbf{m}'_0| = |\mathbf{m}'_1| = |\mathbf{r}'| = v'$ and, moreover, $\boldsymbol{\mathcal{M}}'$ is a distinct mmr-source with min-entropy $\mu(k)$. If $\boldsymbol{\mathcal{M}}$ is a block-source, then $\boldsymbol{\mathcal{M}}'$ additionally has conditional min-entropy at least $\mu(k)$.

Adversary B queries $\boldsymbol{\mathcal{M}}'$ and retrieves a vector $\mathbf{c}$ of ciphertexts. It then uses $\mathbf{p}$ to determine which of the ciphertexts in $\mathbf{c}$ should have been duplicates. That is, for $1 \leq j \leq v$, it lets $i = \mathbf{p}[j]$ and sets $\mathbf{c}'[j] = \mathbf{c}[i]$. It then returns $\mathbf{c}'$ to adversary A.

We bound the advantage of A by that of B. The simulation by B is correct (it matches the IND-CDA$_{\mathcal{AE},k}$ game A expects) as long as for each **LR** query the equality pattern computed by B matches the equality pattern of the vectors output by $\boldsymbol{\mathcal{M}}$ when run within $\boldsymbol{\mathcal{M}}'$. Since $\boldsymbol{\mathcal{M}}$ is equality pattern respecting, the equality pattern of its output always matches a reference $\hat{E}^v$ with probability at least $1 - 2^{-\zeta}$. So for any query by A, the two patterns resulting from the two runs of $\boldsymbol{\mathcal{M}}$ will not match with probability at most $2 \cdot 2^{-\zeta}$. A union bound gives that the probability of failure across all queries is at most $2q \cdot 2^{-\zeta}$. Thus

$$\mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},A}(k) \leq 2\left(\Pr\left[\,\mathrm{CDA}^{B}_{\mathcal{AE},k} \Rightarrow \mathsf{true}\,\right] + \frac{2q}{2^{\zeta}}\right) - 1 = \mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},B}(k) + \frac{4q}{2^{\zeta}} .$$

▮

5 The Role of Anonymity in Adaptive CDA Security

Before detailing specific constructions, we first provide some general results on the relationship between anonymity and adaptive hedge security. For encryption schemes, anonymity (also called key privacy) requires that ciphertexts leak no information about the public key used to perform encryption. In the randomized setting, this was first formalized by Bellare et al. [4]. Here we will review the key-privacy notion for randomized encryption from [4], present a weaker variant of it that will be useful later, and then give a new notion of key privacy in the face of chosen-distribution attacks. The last proves to be sufficient for a general implication that any encryption scheme which is anonymous and non-adaptively hedge secure is also adaptively hedge secure. We will finish by showing that all universal LTDFs are anonymous in this new sense.

Some intuition. We start by highlighting some basic issues related to anonymity and adaptive hedge security. As discussed in the previous section, IND-CDA security has the limitation that an adversary cannot know the public key until after all of its **LR** queries are made. The reason is that no encryption scheme can be secure against chosen-distribution attacks when **LR** queries can be made knowing the public key. Let $\mathcal{AE} = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. Given a public key pk for $\mathcal{AE}$, let $\boldsymbol{\mathcal{M}}$ be the distribution that samples (m_0, m_1, r) from the set of all triples for which the first bit of the ciphertext output by $\mathcal{E}(pk, m_b; r)$ is b. Note that $\boldsymbol{\mathcal{M}}$ has high conditional min-entropy for sufficiently long messages and that it can be implemented efficiently using a loop that samples uniformly and checks the result using pk. An adversary can always then win a variant of game CDA that instead returns pk at the end of **Initialize** using $\boldsymbol{\mathcal{M}}$.

Moreover, consider the situation in which $\mathcal{AE}$ reveals public keys via its encryption algorithm. For example, it prepends all ciphertexts with the public key pk used. (Note this does not impact message privacy.) No such encryption scheme will meet adaptive IND-CDA security, because the adversary can query an arbitrary high conditional min-entropy source, extract pk from the resulting ciphertexts, and then has enough information to query the source $\boldsymbol{\mathcal{M}}$ described above in its second query.

This implies that achieving adaptively-secure hedge encryption requires an encryption scheme that, minimally, does not allow recovery of a public key from ciphertexts. In fact, we will formalize a stronger notion of key privacy under chosen distribution attacks and show that this is sufficient for achieving adaptive security.

Key-privacy with good randomness. First, however, we review the key privacy notion of [4], referred to as indistinguishability of keys. It applies to settings where randomness is always good. Let $\mathcal{AE}$ be an encryption scheme. Game $\mathrm{IK}_{\mathcal{AE},k}$ is shown in Figure 3. The advantage of an IK adversary A is

$$\mathbf{Adv}^{\mathrm{ik}}_{\mathcal{AE},A}(k) = 2 \cdot \Pr\left[\, \mathrm{IK}^{A}_{\mathcal{AE},k} \Rightarrow \mathsf{true} \,\right] - 1 \, .$$

We say that a PKE scheme $\mathcal{AE}$ with message length $n(\cdot)$ and randomness length $\rho(\cdot)$ is IK secure if for all PT adversaries A the function $\mathbf{Adv}^{\mathrm{ik}}_{\mathcal{AE},A}(\cdot)$ is negligible.

Figure 3 also details game KR-UMA, which formalizes a weaker anonymity notion when good randomness is used. Let $\mathcal{AE}$ be an encryption scheme. This notion, called key recovery under unknown message attack, will be useful as a technical tool in Section 6. The advantage of a KR-UMA adversary A is

$$\mathbf{Adv}^{\mathrm{kr\text{-}uma}}_{\mathcal{AE},A}(k) = \Pr\left[\, \mathrm{KR\text{-}UMA}^{A}_{\mathcal{AE},k} \Rightarrow \mathsf{true} \,\right] \, .$$

We say that a PKE scheme $\mathcal{AE}$ with message length $n(\cdot)$ and randomness length $\rho(\cdot)$ is KR-UMA secure if for all PT adversaries A the function $\mathbf{Adv}^{\mathrm{ik}}_{\mathcal{AE},A}(\cdot)$ is negligible.

The following gives that IK security implies KR-UMA security.

Theorem 5.1 Let $\mathcal{AE}$ be a PKE scheme and A be a KR-UMA adversary making at most q_e queries to **Enc** and q_c queries to **Check**. Then there exists an IK adversary B such that for all k

$$\mathbf{Adv}^{\mathrm{kr\text{-}uma}}_{\mathcal{AE},A}(k) \leq \mathbf{Adv}^{\mathrm{ik}}_{\mathcal{AE},B}(k) + q_c \cdot \mathsf{maxpk}_{\mathcal{AE}}(k) \, .$$

Adversary B runs in time that of A and makes q_e **LR**. $\square$

Proof: We build IK adversary B from the KR-UMA adversary A. Adversary B, on input (pk_0, pk_1), first runs $A(par)$. When A makes a **Enc** query, B picks a random message and queries its **LR** oracle on it, returning the result. When A makes a **Check** query, B determines if $pk_1 = pk$, returning true if so and false otherwise. When A finishes, B outputs 1 if **Check** ever returned true and otherwise returns 0. Now consider the case that $a = 1$ in the execution of $\mathrm{IK}^{B}_{\mathcal{AE},k}$. Then B's simulation of $\mathrm{KR\text{-}UMA}_{\mathcal{AE},k}$ is perfect, meaning

$$\Pr\left[\, \mathrm{IK}^{B}_{\mathcal{AE},k} \Rightarrow \mathsf{true} \;\middle|\; a = 1 \,\right] = \Pr\left[\, \mathrm{KR\text{-}UMA}^{A}_{\mathcal{AE},k} \Rightarrow \mathsf{true} \,\right] = \mathbf{Adv}^{\mathrm{kr\text{-}uma}}_{\mathcal{AE},A}(k) \, .$$

Next consider if $a = 0$. Then the execution of A and its queries' responses are entirely independent of pk_1, and so the probability that A queries **Check** on pk_1 is at most $q_c \cdot \mathsf{maxpk}_{\mathcal{AE}}(k)$. Thus

$$\Pr\left[\, \mathrm{IK}^{B}_{\mathcal{AE},k} \Rightarrow \mathsf{true} \;\middle|\; a = 0 \,\right] \leq q_c \cdot \mathsf{maxpk}_{\mathcal{AE}}(k) \, .$$

```
Game $\mathrm{IK}_{\mathcal{AE},k}$

procedure Initialize($1^k$):
    $par \leftarrow_\$ \mathcal{P}(1^k)$
    $(pk_0, sk_0) \leftarrow_\$ \mathcal{K}(par)$
    $(pk_1, sk_1) \leftarrow_\$ \mathcal{K}(par)$
    $a \leftarrow_\$ \{0,1\}$
    Ret $(pk_0, pk_1)$

procedure LR($m$):
    $r \leftarrow_\$ \{0,1\}^{\rho(k)}$
    Ret $\mathcal{E}(pk_a, m\,;\,r)$

procedure Finalize($a'$):
    Ret $(a = a')$
```

```
Game KR-UMA$_{\mathcal{AE},k}$

procedure Initialize($1^k$):
    $par \leftarrow_\$ \mathcal{P}(1^k)$
    $(pk, sk) \leftarrow_\$ \mathcal{K}(par)$
    Ret $par$

procedure Enc:
    $m \leftarrow_\$ \{0,1\}^{n(k)}$
    $r \leftarrow_\$ \{0,1\}^{\rho(k)}$
    Ret $\mathcal{E}(pk, m\,;\,r)$

procedure Check($pk'$):
    win $\leftarrow (pk = pk')$
    Ret win

procedure Finalize():
    Ret win
```

```
Game ANON$_{\mathcal{AE},k}$

procedure Initialize($1^k$):
    $par \leftarrow_\$ \mathcal{P}(1^k)$
    $(pk_0, sk_0) \leftarrow_\$ \mathcal{K}(par)$
    $(pk_1, sk_1) \leftarrow_\$ \mathcal{K}(par)$
    $a \leftarrow_\$ \{0,1\}$
    Ret $par$

procedure Enc($\boldsymbol{\mathcal{M}}$):
    If pkout = true
        Ret $\bot$
    $(\mathbf{m}, \mathbf{r}) \leftarrow_\$ \boldsymbol{\mathcal{M}}(1^k)$
    Ret $\mathcal{E}(pk_0, \mathbf{m}; \mathbf{r})$

procedure LR($\boldsymbol{\mathcal{M}}$):
    $(\mathbf{m}, \mathbf{r}) \leftarrow_\$ \boldsymbol{\mathcal{M}}(1^k)$
    $\mathbf{c} \leftarrow \mathcal{E}(pk_a, \mathbf{m}; \mathbf{r})$
    pkout $\leftarrow$ true
    Ret $(pk_0, pk_1, \mathbf{c})$

procedure Finalize($a'$):
    Ret $(a = a')$
```

Figure 3: Key privacy games.

Combining all the above we derive that

$$\begin{aligned}\mathbf{Adv}^{\mathrm{ik}}_{\mathcal{AE},B}(k) &= \Pr\left[\,\mathrm{IK}^{B}_{\mathcal{AE},k} \Rightarrow \mathsf{true} \;\middle|\; a=1\,\right] - \Pr\left[\,\mathrm{IK}^{B}_{\mathcal{AE},k} \Rightarrow \mathsf{true} \;\middle|\; a=0\,\right] \\ &\geq \mathbf{Adv}^{\text{kr-uma}}_{\mathcal{AE},A}(k) - q_c \cdot \mathsf{maxpk}_{\mathcal{AE}}(k)\end{aligned}$$

∎

Key-privacy under chosen-distribution attack. We now formalize a notion of anonymity for chosen-distribution attacks. Let $\mathcal{AE} = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. Game ANON$_{\mathcal{AE}}$ shown in Figure 3 provides the adversary with two oracles. An ANON adversary A is one whose queries are all mr-sources. The advantage of ANON adversary A is

$$\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{AE},A}(k) = 2 \cdot \Pr\left[\,\mathrm{ANON}^{A}_{\mathcal{AE},k} \Rightarrow \mathsf{true}\,\right] - 1\;.$$

We say that a PKE scheme $\mathcal{AE}$ with message length $n(\cdot)$ and randomness length $\rho(\cdot)$ is ANON secure for distinct (μ, v, n, ρ)-mr-sources if for all PT adversaries A that only query distinct (μ, v, n, ρ)-mr-sources the function $\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{AE},A}(\cdot)$ is negligible. We can extend this notion to mr-block-sources in the obvious way.

In the special case that the randomness length of $\mathcal{AE}$ is always zero, the ANON definition formalizes anonymity for deterministic encryption or, equivalently, trapdoor functions, generalizing a definition from [4].

While we will use ANON mainly as a technical tool to show schemes meet adaptive IND-CDA, it is also of independent interest as a new security target for PKE schemes when key privacy is important. (That is, one might want to hedge against bad randomness for anonymity as well as message privacy.)

From non-adaptive to adaptive hedge security. The following theorem shows that achieving ANON security and non-adaptive IND-CDA security are sufficient for achieving adaptive IND-CDA security.

Theorem 5.2 Let $\mathcal{AE} = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme with message length $n(\cdot)$ and randomness length $\rho(\cdot)$. Let A be a IND-CDA adversary making $q(\cdot)$ **LR** queries, each being a distinct (μ, v, n, ρ)-mmr-source (resp. block-source). Then there exist IND-CDA adversary B and ANON adversary C such that for all k

$$\mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},A}(k) \le q(k)\cdot \mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},B}(k) + 2q(k)\cdot \mathbf{Adv}^{\mathrm{anon}}_{\mathcal{AE},C}(k)\ .$$

B makes one **LR** query consisting of a distinct (μ, v, n, ρ)-mmr-source (resp. block-source). C makes at most $q(k) - 1$ **Enc** queries and one **LR** query, all these consisting of distinct (μ, v, n, ρ)-mr-sources (resp. block-sources). Both B and C run in the same time as A. $\square$

Before giving the proof we first fix some useful definitions. Let game $\mathrm{CDA1}_{\mathcal{AE},k}$ be the same as game $\mathrm{CDA}_{\mathcal{AE},k}$ (Figure 2) except that the line of code $b \leftarrow\!\!{\scriptstyle\$}\ \{0,1\}$ is replaced by $b \leftarrow 1$ and **Finalize** is omitted. (Recall that when **Finalize** is omitted, the output of the game is the output of A.) Similarly define $\mathrm{CDA0}_{\mathcal{AE},k}$ except with $b \leftarrow 0$. Then a standard argument gives that

$$\mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},A}(k) = \Pr\left[\,\mathrm{CDA1}^{A}_{\mathcal{AE},k} \Rightarrow 1\,\right] - \Pr\left[\,\mathrm{CDA0}^{A}_{\mathcal{AE},k} \Rightarrow 1\,\right]\ . \tag{2}$$

We can analogously define $\mathrm{ANON1}_{\mathcal{AE},k}$ and $\mathrm{ANON0}_{\mathcal{AE},k}$.

Proof of Theorem 5.2: Fix a $k \in \mathbb{N}$ and let $q = q(k)$. Let A be a IND-CDA adversary against $\mathcal{AE}$. We perform a hybrid argument to bound A's advantage. Let $\mathrm{HYB}_0, \cdots, \mathrm{HYB}_q$ be a sequence of hybrid games that work as shown in Figure 4 (boxed statement omitted). In game HYB_i the first $q - i$ **LR** queries are answered using the $\mathbf{m}_1$ vector and the last i **LR** queries are answered using the $\mathbf{m}_0$ vector. Note that $\mathrm{HYB}_0 = \mathrm{CDA1}$ while $\mathrm{HYB}_q = \mathrm{CDA0}$. Also defined in Figure 4 are games $\mathrm{HYB}'_0, \ldots, \mathrm{HYB}'_q$. Each HYB'_i is the same as HYB_i except that a distinct key is used to answer the $(q-i)^{th}$ **LR** query. Let $h_i = \Pr\left[\,\mathrm{HYB}^A_i \Rightarrow 1\,\right]$ for $0 \le i \le q$ and let $h'_i = \Pr\left[\,\mathrm{HYB}'^A_i \Rightarrow 1\,\right]$ for $0 \le i \le q$. Then a union bound gives that

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},A}(k) &= \sum_{0\le i\le q-1} (h_i - h'_i + h'_i - h'_{i+1} + h'_{i+1} - h_{i+1}) \\
&= \sum_{0\le i\le q-1} (h_i - h'_i) + \sum_{0\le i\le q-1} (h'_i - h'_{i+1}) + \sum_{0\le i\le q-1} (h'_{i+1} - h_{i+1})\ .
\end{aligned} \tag{3}$$

We bound each of the three sums by appropriate adversaries, details of which are given in Figure 4. First we define ANON adversaries C_i, parameterized by $i \in [0\,..\,q-1]$, and ANON adversaries $\overline{C}_i$, parameterized by $i \in [1\,..\,q]$. The difference between C_i and $\overline{C}_i$ is that the latter outputs the complement of A's output bit. By construction we have that

$$h_i = \Pr\left[\,\mathrm{ANON1}^{C_{i_1}}_{\mathcal{AE},k} \Rightarrow 1\,\right] \quad \text{and} \quad h'_i = \Pr\left[\,\mathrm{ANON0}^{C_{i,1}}_{\mathcal{AE},k} \Rightarrow 1\,\right]$$

for $i \in [0\,..\,q-1]$ and also that

$$h'_i = \Pr\left[\,\mathrm{ANON1}^{\overline{C}_i}_{\mathcal{AE},k} \Rightarrow 1\,\right] \quad \text{and} \quad h_i = \Pr\left[\,\mathrm{ANON0}^{\overline{C}_i}_{\mathcal{AE},k} \Rightarrow 1\,\right]$$

for $i \in [1\,..\,q]$. Let C be the adversary that first chooses $d \leftarrow\!\!{\scriptstyle\$}\ \{0,1\}$ and, then $j \leftarrow\!\!{\scriptstyle\$}\ [0 + d\,..\,(q-1) + d]$. It then outputs $b' \oplus d$. By construction if $d = 1$, then C implements $\overline{C}_j$ and otherwise implements C_j. Then we have that

$$\begin{aligned}
\sum_{i=0}^{q-1}(h_i - h'_i) + \sum_{i=1}^{q}(h'_i - h_i) &= \sum_{i=0}^{q-1}\Big(\Pr\left[\,\mathrm{ANON1}^{C_i}_{\mathcal{AE},k} \Rightarrow 1\,\right] - \Pr\left[\,\mathrm{ANON0}^{C_i}_{\mathcal{AE},k} \Rightarrow 1\,\right]\Big) \\
&\quad + \sum_{i=1}^{q}\Big(\Pr\left[\,\mathrm{ANON1}^{\overline{C}_i}_{\mathcal{AE},k} \Rightarrow 1\,\right] - \Pr\left[\,\mathrm{ANON0}^{\overline{C}_i}_{\mathcal{AE},k} \Rightarrow 1\,\right]\Big) \\
&= 2q\cdot \mathbf{Adv}^{\mathrm{anon}}_{\mathcal{AE},k}(C)
\end{aligned} \tag{4}$$

where the last equality follows from multiplying the first sum by $2q\cdot \Pr[j = i \wedge d = 0]$ and the second sum by $2q\cdot \Pr[j = i \wedge d = 1]$ (both products equal one). (The events "$j = 1$", "$d = 0$", and "$d = 1$" are defined over the coins used in executing the respective ANON games with C.)

```
Games $\mathrm{HYB}_i$ , $\boxed{\mathrm{HYB}'_i}$

procedure Initialize($1^k$):
$(pk_0, sk_0) \leftarrow\$ \mathcal{K}(1^k)$
$(pk_1, sk_1) \leftarrow\$ \mathcal{K}(1^k)$
Ret $1^k$

procedure RevealPK():
Ret $pk_0$

procedure LR($\boldsymbol{\mathcal{M}}$):
$j \leftarrow j + 1$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ \boldsymbol{\mathcal{M}}(1^k)$
If $j > q - i$ then $b \leftarrow 0$ else $b \leftarrow 1$
$a \leftarrow 0$ $\boxed{\text{; If } j = i \text{ then } a \leftarrow 1}$
$\mathbf{c} \leftarrow \mathcal{E}(pk_a, \mathbf{m}_b; \mathbf{r})$
Ret $\mathbf{c}$
```

```
adversary $C_i(par)$:
Run $A(par)$

  On query RevealPK():
  Ret $pk_0$

  On query LR($\boldsymbol{\mathcal{M}}$):
  $j \leftarrow j + 1$
  If $j > q - i$ then $b \leftarrow 0$ else $b \leftarrow 1$
  If $j = q - i$ then $(pk_0, pk_1, \mathbf{c}) \leftarrow \mathbf{LR}(\boldsymbol{\mathcal{M}}_b)$
  Else $\mathbf{c} \leftarrow \mathbf{Enc}(\boldsymbol{\mathcal{M}}_b)$
  Ret $\mathbf{c}$

When $A$ halts with output $b'$
Ret $1 - b'$
```

```
adversary $\overline{C}_i(par)$:
Run $A(par)$

  On query RevealPK():
  Ret $pk_0$

  On query LR($\boldsymbol{\mathcal{M}}$):
  $j \leftarrow j + 1$
  If $j > q - i$ then $b \leftarrow 0$ else $b \leftarrow 1$
  If $j = q - i$ then $(pk_0, pk_1, \mathbf{c}) \leftarrow \mathbf{LR}(\boldsymbol{\mathcal{M}}_b)$
  Else $\mathbf{c} \leftarrow \mathbf{Enc}(\boldsymbol{\mathcal{M}}_b)$
  Ret $\mathbf{c}$

When $A$ halts with output $b'$
Ret $b'$
```

```
adversary $B_i(par)$:
$(pk_1, sk_1) \leftarrow\$ \mathcal{K}(par)$
Run $A(par)$

  On query RevealPK():
  Ret $pk_0$

  On query LR($\boldsymbol{\mathcal{M}}$):
  $j \leftarrow j + 1$
  If $j = q - i$ then
      $\mathbf{c} \leftarrow \mathbf{LR}(\boldsymbol{\mathcal{M}})$
  Else
      $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ \boldsymbol{\mathcal{M}})$
      If $j < q - i$ then $\mathbf{c} \leftarrow \mathcal{E}(pk_0, \boldsymbol{\mathcal{M}}_1 \,;\, \mathbf{r})$
      If $j > q - i$ then $\mathbf{c} \leftarrow \mathcal{E}(pk_0, \boldsymbol{\mathcal{M}}_0 \,;\, \mathbf{r})$
  Ret $\mathbf{c}$

When $A$ halts with output $b'$
Ret $b'$
```

Figure 4: Hybrid games and adversaries used in proof of Theorem 5.2. For an mmr-source $\boldsymbol{\mathcal{M}}$, the mr-source $\boldsymbol{\mathcal{M}}_b$ runs $\boldsymbol{\mathcal{M}}$ to get $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r})$ and outputs $(\mathbf{m}_b, \mathbf{r})$.

For the remaining sum, we define CDA adversaries B_i for $i \in [0\,..\,q-1]$. By construction

$$h'_i - h'_{i+1} = \Pr\left[\,\mathrm{CDA1}^{B_i}_{\mathcal{AE},k} \Rightarrow 1\,\right] - \Pr\left[\,\mathrm{CDA0}^{B_i}_{\mathcal{AE},k} \Rightarrow 1\,\right]$$

for $i \in [0\,..\,q-1]$. Let B be the CDA adversary that chooses $c \leftarrow\!\!\$\ \{0,\ldots,q-1\}$ and then executes the code of B_c. A straightforward analysis gives that

$$\sum_{i=0}^{q-1}(h'_i - h'_{i+1}) = q \cdot \mathbf{Adv}^{\mathrm{cda}}_{\mathcal{AE},k}(B)\ . \tag{5}$$

Substituting into (3) according to (4) and (5) completes the proof. ▌

Given a non-adaptively IND-CDA secure scheme, Theorem 5.2 reduces the task of showing it adaptively secure to that of showing it meets the ANON definition. Of course, ANON is still an adaptive notion. (Adversaries can formulate their **LR** query to be a source that's a function of previously seen ciphertexts.) Nevertheless, it formalizes a sufficient condition for adaptive CDA security of any PKE scheme and captures the relationship between adaptivity and anonymity. We believe this is an interesting (and novel) application of anonymity.

Universal LTDFs are anonymous. We now establish that u-LTDFs are anonymous. While this result might also be of general interest, it will be specifically useful for schemes based on u-LTDFs. Intuitively u-LTDFs are anonymous because the lossy mode admits a universal hash, implying that no information about the public key is leaked by outputs (generated from sources with high conditional min-entropy). One might expect that formalizing this intuition would follow from straightforward application of the Leftover Hash Lemma (LHL) [28]. However our anonymity definitions are adaptive, so one cannot apply the LHL (or even the generalized LHL [18]) directly. Rather, we first show an adaptive variant of the LHL is implied by the standard LHL via a hybrid argument. See Appendix A for details. Here we use it to prove the following theorem.

Theorem 5.3 Let $\mathcal{AE}_{\mathrm{d}} = (\mathcal{P}_{\mathrm{d}}, \mathcal{K}_{\mathrm{d}}, \mathcal{E}_{\mathrm{d}}, \mathcal{D}_{\mathrm{d}})$ be a (deterministic) encryption scheme with message length $n(\cdot)$ and an associated universal-inducing (n,ℓ)-lossy key generator $\mathcal{K}_l$. Let A be an ANON adversary making $q(\cdot)$ **Enc** queries and a single **LR** query, each of these being a (μ, v, n)-m-block-source. Then there exists LOS adversary B such that for all k

$$\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{AE}_{\mathrm{d}},A}(k) \le 2 \cdot \mathbf{Adv}^{\mathrm{los}}_{\mathcal{AE}_{\mathrm{d}},B}(k) + 3 \cdot q(k) \cdot v(k) \cdot \sqrt{2^{n(k)-\ell(k)-\mu(k)}}\ .$$

B runs in time that of A. □

Before giving the proof, we first consider RtD and PtD when instantiated with a deterministic encryption scheme that is a u-LTDF. We can apply Theorem 5.3 to conclude ANON security for both schemes. Combining this with Theorems 6.2 and 5.2 yields proof of adaptive hedge security for RtD. Likewise, combining it with Theorems 6.3 and 5.2 yields proof of adaptive hedge security for PtD. Also Theorems 5.2 and 5.3 combine with [11, Th. 5.1] to give the first adaptively-secure deterministic encryption scheme (based on u-LTDFs).

Proof of of Theorem 5.3: Let $\mathcal{K}_0$ denote $\mathcal{K}_{\mathrm{d}}$ and $\mathcal{K}_1$ denote $\mathcal{K}_l$. We define games $\mathrm{H}_{\alpha,\beta,a}$ for $\alpha, \beta, a \in \{0,1\}$. For $\alpha, \beta, a \in \{0,1\}$ let $p(\alpha,\beta,a) = \Pr\left[\,\mathrm{H}^A_{\alpha,\beta,a}(k) \Rightarrow 1\,\right]$. Here α selects between the normal or universal modes for pk_0, β selects between the normal and universal modes for pk_1, and a selects which of pk_0 or pk_1 is used to respond to the **LR** query of A. Then

$$\begin{aligned}\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{AE},A}(k) &= p(0,0,1) - p(0,0,0)\\ &= \big(p(0,0,1) - p(1,1,1)\big) + \big(p(1,1,1) - p(1,1,0)\big) + \big(p(1,1,0) - p(0,0,0)\big)\ .\end{aligned}$$

For $a \in \{0,1\}$ we can design B_a so that $|p(1,1,a) - p(0,0,a)| \le 2 \cdot \mathbf{Adv}^{\mathrm{los}}_{\mathcal{AE},\mathcal{K}_l,B_a}(k)$. Key here is that the message sources that A queries to its oracles are efficient so B_a can sample from them. We describe

```
Game $\mathrm{H}_{\alpha,\beta,a}$

procedure Initialize($1^k$):
    $(pk_0, sk_0) \leftarrow\!\!\$\ \mathcal{K}_\alpha(1^k)$
    $(pk_1, sk_1) \leftarrow\!\!\$\ \mathcal{K}_\beta(1^k)$

procedure Enc($\boldsymbol{\mathcal{M}}$):
    $(\mathbf{m}, \mathbf{r}) \leftarrow\!\!\$\ \boldsymbol{\mathcal{M}}$
    $\mathbf{c} \leftarrow \mathcal{E}_\mathrm{d}(pk_0, \mathbf{r} \,\|\, \mathbf{m})$
    Ret $\mathbf{c}$

procedure LR($\boldsymbol{\mathcal{M}}$):
    $\mathbf{m} \leftarrow\!\!\$\ \boldsymbol{\mathcal{M}}$
    $\mathbf{c} \leftarrow \mathcal{E}_\mathrm{d}(pk_a, \mathbf{m})$
    Ret $(pk_0, pk_1, \mathbf{c})$

procedure Finalize($a'$):
    Ret $a'$
```

Figure 5: Games $\mathrm{H}_{\alpha,\beta,a}$ for $\alpha, \beta, a \in \{0,1\}$ used in the proof that any universal LTDF is anonymous.

$B_a(pk_0, pk_1)$. It runs A. When A makes query $\mathbf{Enc}(\boldsymbol{\mathcal{M}})$ it lets $\mathbf{m} \leftarrow\!\!\$\ \boldsymbol{\mathcal{M}}$ and returns $\mathcal{E}_\mathrm{d}(pk_0, \mathbf{m})$ to A. When A makes query $\mathbf{LR}(\boldsymbol{\mathcal{M}})$ it lets $\mathbf{m} \leftarrow\!\!\$\ \boldsymbol{\mathcal{M}}$ and $\mathbf{c} \leftarrow \mathcal{E}_\mathrm{d}(pk_a, \mathbf{m})$ and returns $(pk_0, pk_1, \mathbf{c})$ to A. Let d denote the output of A. Then B_0 returns $1 - d$ and B_1 returns d.

Define game R to be work like games $\mathrm{H}_{1,1,a}$ except that all queries are answered by selecting $\mathbf{c}[i] \leftarrow\!\!\$\ R$ for $1 \le i \le |\mathbf{m}|$ (instead of applying $\mathcal{E}_\mathrm{d}$ to $\mathbf{m}$). Here R is the range of $\mathcal{H} = (\mathcal{K}_l, \mathcal{E}_\mathrm{d})$. Let $p_R = \Pr[\mathrm{R}^A(k) \Rightarrow 1]$. Now we bound

$$p(1,1,1) - p(1,1,0) = \big(p(1,1,1) - p_R\big) + \big(p_R - p(1,1,0)\big)$$

term by term.

We design LH adversary A_0 so that $p_R - p(1,1,0) \le \mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},A_0}(k)$. Adversary A_0 first computes $(pk_1, sk_1) \leftarrow\!\!\$\ K_1(1^k)$. It runs A, forwarding any $\mathbf{Enc}$ query $\boldsymbol{\mathcal{M}}$ of A to its $\mathbf{RoR}$ oracle and returning the result. It forwards any $\mathbf{LR}$ query of A to its $\mathbf{RoR}$ oracle gets back $\mathbf{c}$, queries $\mathbf{RevealPK}$ to retrieve pk_0, and returns $(pk_0, pk_1, \mathbf{c})$ to A. It outputs what A outputs.

We design LH adversary A_1 so that $p(1,1,1) - p_R \le \mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},A_1}(k)$. Adversary A_1 first computes $(pk_0, sk_0) \leftarrow\!\!\$\ \mathcal{K}_1(1^k)$. It runs A. When A makes $\mathbf{Enc}$ query $\boldsymbol{\mathcal{M}}$ it lets $\mathbf{m} \leftarrow\!\!\$\ \boldsymbol{\mathcal{M}}$ and $\mathbf{c} \leftarrow \mathcal{E}_\mathrm{d}(pk_0, \mathbf{m})$ and returns $\mathbf{c}$ to A. When A makes $\mathbf{LR}$ query $\boldsymbol{\mathcal{M}}$ it queries its $\mathbf{RoR}$ oracle to get $\mathbf{c}$, queries $\mathbf{RevealPK}$ to get pk_1, and returns $(pk_0, pk_1, \mathbf{c})$ to A. It outputs what A outputs. ▌

6 Constructions of Hedged Public-key Encryption

We now build hedged public-key encryption schemes. These are schemes that simultaneously achieve IND-CPA security and IND-CDA security. Such schemes do not sacrifice any security when randomness is good, but should randomness be poor, IND-CDA provides another line of defense. We start with schemes in the RO model which are easy to deploy and fast. We then analyze constructions that can achieve security in the standard model.

6.1 Hedging in the Random Oracle Model

Randomized-encrypt-with-hash. Let $\mathcal{AE}_\mathrm{r} = (\mathcal{P}_\mathrm{r}, \mathcal{K}_\mathrm{r}, \mathcal{E}_\mathrm{r}, \mathcal{D}_\mathrm{r})$ be a (randomized) PKE scheme with message length $n_\mathrm{r}(\cdot)$ and randomness length $\rho(\cdot)$. Let $\mathcal{R} : \{0,1\}^* \to \{0,1\}^*$ be a random oracle. Let $\mathsf{REwH}[\mathcal{AE}_\mathrm{r}] = (\mathcal{P}_\mathrm{r}, \mathcal{K}, \mathcal{E}, \mathcal{D}_\mathrm{r})$ be the scheme parametrized by randomizer length κ that works as follows. Parameter generation and decryption are the same as in $\mathcal{AE}_\mathrm{r}$. Key generation, on input par_r, runs $\mathcal{K}_\mathrm{r}(par_\mathrm{r})$ to get $(pk_\mathrm{r}, sk_\mathrm{r})$, chooses $K \leftarrow\!\!\$\ \{0,1\}^{\kappa(k)}$, and lets $pk = (pk_\mathrm{r}, K)$ and $sk = sk_\mathrm{r}$. Encryption is defined by

$$\mathcal{E}^{\mathcal{R}}((pk_\mathrm{r}, K), m\,;\, r) = \mathcal{E}_\mathrm{r}(pk_\mathrm{r}, m\,;\, r')$$

where r' is the first $\rho(k)$ bits of $\mathcal{R}(pk_\mathrm{r} \,\|\, K \,\|\, r \,\|\, m)$.

Intuitively, the random oracle provides perfect and (as long as m and r are hard to predict) private randomness. When the randomizer length $\kappa(k) = 0$ for all k, we refer to the scheme as $\mathsf{REwH1}$, while when $\kappa(k) > 0$ for all k we refer to the scheme as $\mathsf{REwH2}$. The scheme extends the Encrypt-with-Hash deterministic encryption scheme from [6], which is a special case of $\mathsf{REwH1}$ when r has length 0 and κ

is 0. The scheme is also reminiscent of constructions in the symmetric setting that utilize a PRF to ensure good randomness [29, 36], as well as schemes using the Fujisaki-Okamoto transform [21].

REwH IS HEDGE SECURE. The next theorem will establish the hedge security of REwH for various instantiations. The scheme achieves non-adaptive IND-CDA security when $\mathcal{AE}_{\mathrm{r}}$ is IND-CPA. It achieves adaptive IND-CDA security when $\mathcal{AE}_{\mathrm{r}}$ is additionally key-anonymous in the sense of [4] or when the randomizer length κ is sufficiently large (e.g., $\kappa(k) \geq k$).

Theorem 6.1 [REwH is H-IND secure] Let $\mathcal{AE}_{\mathrm{r}} = (\mathcal{P}_{\mathrm{r}}, \mathcal{K}_{\mathrm{r}}, \mathcal{E}_{\mathrm{r}}, \mathcal{D}_{\mathrm{r}})$ be a PKE scheme with message length $n(\cdot)$ and randomness length ρ and let $\mathcal{AE} = \mathsf{REwH}[\mathcal{AE}_{\mathrm{r}}] = (\mathcal{P}_{\mathrm{r}}, \mathcal{K}_{\mathrm{r}}, \mathcal{E}, \mathcal{D}_{\mathrm{r}})$ be the PKE scheme constructed from it.

- (IND-CPA) Let A be an IND-CPA adversary Then there exists an IND-CPA adversary B such that for all k
$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},A}(k) \leq 2\cdot \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_{\mathrm{r}},B}(k)$$
where B runs in time at most $\max\{\mathsf{Time}(A), 2h\cdot \mathsf{Time}(\mathcal{E}_{\mathrm{r}})\}$.
- (IND-CDA) Let A be an adversary that makes $q(\cdot)$ **LR** queries each consisting of a distinct (μ, v, n, ρ)-mmr-source and making at most $h(\cdot)$ random oracle queries. Then there exists an IND-CPA adversary B such that for all k
$$\mathbf{Adv}^{\text{cda}}_{\mathcal{AE},A}(k) \leq 2\cdot \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_{\mathrm{r}},B}(k) + \frac{(q(k))^2 v(k) + h(k)}{2^{\mu(k)}} + \chi$$
where
$$\chi = \begin{cases} \min\left\{\dfrac{h(k)}{2^{\kappa(k)}}, \mathbf{Adv}^{\text{kr-uma}}_{\mathcal{AE}_{\mathrm{r}},C}(k)\right\} & \text{if } q(\cdot) \neq 1 \\ \dfrac{h(k)\cdot \mathsf{maxpk}_{\mathcal{AE}_{\mathrm{r}}}(k)}{2^{\kappa(k)}} & \text{if } q(\cdot) = 1 \end{cases}$$
Adversary B runs in time at most that of A and makes $q(k)v(k)$ queries. Adversary C runs in time at most that of A and makes $q(k)v(k)$ **Enc** queries and $h(k)$ **Check** queries. □

Proof: Fix some $k \in \mathbb{N}$ and let $q = q(k)$, $v = v(k)$, $\kappa = \kappa(k)$, and $n = n(k)$. We begin by proving the IND-CPA portion of the theorem. Let A be an IND-CPA adversary that makes one **LR** query and does not repeat any **Hash** queries (this is without loss of generality). Games G_0, G_1, and G_2 are shown in Figure 6. All the games include a **Finalize** procedure (not shown explicitly) that is the same as the IND-CPA **Finalize** procedure. Game G_0 (boxed statement included) implements exactly the IND-CPA$_{\mathcal{AE},k}$ game. Game G_1 removes the boxed statement, which ensured consistency between the use of **Hash** in **LR** and with direct queries to **Hash** by A. In G_1, independent randomness (which is never used again in the game) is used to encrypt the challenge message. Game G_2 makes this explicit. We will now justify that

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},A}(k) &= 2\cdot \Pr\left[\,\text{IND-CPA}^{A}_{\mathcal{AE},k} \Rightarrow \mathsf{true}\,\right] - 1 \\
&= 2\cdot \Pr\left[\,\mathrm{G}_0^A \Rightarrow \mathsf{true}\,\right] - 1 && (6)\\
&\leq 2\cdot \left(\Pr\left[\,\mathrm{G}_1^A \Rightarrow \mathsf{true}\,\right] + \Pr\left[\,\mathrm{G}_1^A \text{ sets } \mathsf{bad}\,\right]\right) - 1 && (7)\\
&= 2\cdot \left(\Pr\left[\,\mathrm{G}_2^A \Rightarrow \mathsf{true}\,\right] + \Pr\left[\,\mathrm{G}_2^A \text{ sets } \mathsf{bad}\,\right]\right) - 1 && (8)\\
&\leq 2\cdot \left(\Pr\left[\,\text{IND-CPA}^{B_1}_{\mathcal{AE}_{\mathrm{r}},k} \Rightarrow \mathsf{true}\,\right] + \Pr\left[\,\text{IND-CPA}^{B_2}_{\mathcal{AE}_{\mathrm{r}},k} \Rightarrow \mathsf{true}\,\right]\right) - 1 && (9)\\
&= 2\cdot \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_{\mathrm{r}},B}(k) && (10)
\end{aligned}$$

By construction G_0 is equivalent to IND$_{\mathcal{AE},k}$, justifying (6). The fundamental lemma of game-playing [8] justifies (7) and by construction G_2 and G_1 are equivalent, justifying (8). Let adversary B_1 work as

```
procedure Initialize($1^k$):                    Game $\boxed{G_0}$ , $G_1$
$par \leftarrow_{\$} \mathcal{P}_r(1^k)$ ; $(pk_r, sk_r) \leftarrow_{\$} \mathcal{K}_r(par)$
$K \leftarrow_{\$} \{0,1\}^{\kappa}$ ; $pk \leftarrow (pk_r, K)$
$b \leftarrow_{\$} \{0,1\}$
Ret $pk$

procedure LR($m_0, m_1$):
$r \leftarrow_{\$} \{0,1\}^{\rho}$
$r' \leftarrow_{\$}$ Hash($pk, r, m_b$)
$c \leftarrow \mathcal{E}(pk_r, m_b \,;\, r')$
Ret $c$

procedure Hash($P, R, M$):
$Y \leftarrow_{\$} \{0,1\}^{\rho}$
If $P = pk \wedge \mathtt{H}[P,R,M] \neq \perp$ then
    $\mathsf{bad} \leftarrow \mathsf{true}$ $\boxed{;\ Y \leftarrow \mathtt{H}[P,R,M]}$
$\mathtt{H}[P,R,M] \leftarrow Y$
Ret $Y$
```

```
procedure Initialize($1^k$):                    Game $G_2$
$par \leftarrow_{\$} \mathcal{P}_r(1^k)$ ; $(pk_r, sk_r) \leftarrow_{\$} \mathcal{K}_r(par)$
$K \leftarrow_{\$} \{0,1\}^{\kappa}$ ; $pk \leftarrow (pk_r, K)$
$b \leftarrow_{\$} \{0,1\}$
Ret $pk$

procedure LR($m_0, m_1$):
$r \leftarrow_{\$} \{0,1\}^{\rho}$
$r' \leftarrow_{\$} \{0,1\}^{\rho}$ ; Hash($pk, r, m_b$)
$c \leftarrow \mathcal{E}(pk_r, m_b \,;\, r')$
Ret $c$

procedure Hash($P, R, M$):
$Y \leftarrow_{\$} \{0,1\}^{\rho}$
If $P = pk \wedge \mathtt{H}[P,R,M] \neq \perp$ then
    $\mathsf{bad} \leftarrow \mathsf{true}$ $\boxed{;\ Y \leftarrow \mathtt{H}[P,R,M]}$
$\mathtt{H}[P,R,M] \leftarrow Y$
Ret $Y$
```

Figure 6: Games used in the IND-CPA proof for Theorem 6.1.

follows. It simulates game G_2 for A, using its **LR** oracle to answer A's **LR** query. It outputs whatever bit A outputs. Then, $\Pr[G_2^A \Rightarrow \mathsf{true}] = \Pr[\mathrm{IND}^{B_1}_{\mathcal{AE}_r,k} \Rightarrow \mathsf{true}]$.

Since bad is only set in G_2 if A queries **Hash**(P, R, M) with $R = r$ and the choice of r is independent of the answers given to A's queries, we have that

$$\Pr\left[\, G_2^A \text{ sets } \mathsf{bad} \,\right] \leq \frac{h}{2^{\rho}} \,.$$

However, it is easy to construct an IND-CPA adversary against $\mathcal{AE}_r$ that has advantage $h/2^{\rho}$ using time $2 \cdot \mathsf{Time}(\mathcal{E}_r)$. Let adversary B_2 work as follows. It queries its **LR** oracle on two distinct messages m_0, m_1 to get back ciphertext c. It then repeats h times the following procedure: (1) choose a value r uniformly from $\{0,1\}^{\rho}$ (but without replacement between iterations); (2) run $c_0 \leftarrow \mathcal{E}_r(pk_r, m_0 \,;\, r)$ and $c_1 \leftarrow \mathcal{E}_r(pk_r, m_0 \,;\, r)$; and (3) if $c = c_0$ let $b' = 0$ or if $c = c_1$ let $b' = 1$. Finally B_2 outputs b' if it was set during an iteration and otherwise outputs a random bit. Let "Succ" be the event that one of the values r chosen by B_2 matches the randomness used in responding to the **LR** query. Let "nSucc" be the complementary event. We have that

$$\begin{aligned}
\Pr\left[\,\mathrm{IND}^{B_2}_{\mathcal{AE}_r,k} \Rightarrow \mathsf{true}\right] &= \Pr\left[\,\mathrm{IND}^{B_2}_{\mathcal{AE}_r,k} \Rightarrow 1 \;\middle|\; \mathsf{Succ}\right] \cdot \Pr\left[\,\mathsf{Succ}\,\right] \\
&\quad + \Pr\left[\,\mathrm{IND}^{B_2}_{\mathcal{AE}_r,k} \Rightarrow 1 \;\middle|\; \mathsf{nSucc}\right] \cdot \Pr\left[\,\mathsf{nSucc}\,\right] \\
&= \frac{h}{2^{\rho}} + \frac{1}{2}\left(1 - \frac{h}{2^{\rho}}\right) \\
&= \frac{h}{2^{\rho+1}} + \frac{1}{2} \,.
\end{aligned}$$

Here we have used the fact that $\Pr[\mathsf{Succ}] = h/2^{\rho}$ and that b' will only be set if event Succ occurs. Applying the definition of advantage gives that $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_r,k}(B_2) = h/2^{\rho} \geq \Pr[G_2^A \text{ sets } \mathsf{bad}\,]$. We have justified equation (9). Let adversary B choose $d \leftarrow_{\$} \{1,2\}$ and execute the code of B_d. Then, we have that

$$\Pr\left[\,\mathrm{IND}^{B}_{\mathcal{AE}_r,k} \Rightarrow \mathsf{true}\,\right] = \frac{1}{2}\left(\Pr\left[\,\mathrm{IND}^{B_1}_{\mathcal{AE}_r,k} \Rightarrow \mathsf{true}\right] + \Pr\left[\,\mathrm{IND}^{B_2}_{\mathcal{AE}_r,k} \Rightarrow \mathsf{true}\right]\right)$$

and combining this with the definition of IND advantage gives (10)

We now prove the IND-CDA portion of the theorem. Informally, the sequence of games moves from the setting of the IND-CDA experiment to one in which true random coins are used to answer challenge queries. This is accomplished by setting a flag bad should the random oracle be queried on domain points colliding with the public key and challenge randomness, message pairs output by a message sampler. If bad is never set (such a query never occurs) then the challenge encryptions can be done with random coins. Note that both message samplers $\mathcal{M}$ and the adversary itself can query the random oracle. While $\mathcal{M}$ knows the challenge messages to be handled, we will show that the adversary (and hence, any queried $\mathcal{M}$) does not know the public key pk before the query to **RevealPK**. This is because at least one of the following arguments: $\mathcal{AE}_{\rm r}$ is anonymous, we are considering only non-adaptive hedge security, or the randomizer has sufficient length to be unpredictable. Once we are in a setting in which true random coins are used, then we can use the IND-CPA security of $\mathcal{AE}_{\rm r}$ to conclude security.

To formalize this we use a sequence of games $\mathrm{G}_0 \longrightarrow \mathrm{G}_1 \longrightarrow \mathrm{G}_2 \longrightarrow \mathrm{G}_3 \longrightarrow \mathrm{G}_4 \longrightarrow \mathrm{G}_5$. The games can be found in Figures 7 and 9. We will justify the following sequence of inequalities.

$$\begin{aligned}
\Pr\left[\,\mathrm{CDA}^{A}_{\mathcal{AE},k} \Rightarrow \mathsf{true}\,\right] &= \Pr\left[\,\mathrm{G}_0^A \Rightarrow \mathsf{true}\,\right] && (11)\\
&\le \Pr\left[\,\mathrm{G}_1^A \Rightarrow \mathsf{true}\,\right] + \Pr\left[\,\mathrm{G}_1^A \text{ sets } \mathsf{bad}\,\right] && (12)\\
&= \Pr\left[\,\mathrm{G}_2^A \Rightarrow \mathsf{true}\,\right] + \Pr\left[\,\mathrm{G}_2^A \text{ sets } \mathsf{bad}\,\right] && (13)\\
&\le \Pr\left[\,\mathrm{G}_3^A \Rightarrow \mathsf{true}\,\right] + \Pr\left[\,\mathrm{G}_3^A \text{ sets } \mathsf{bad}\,\right] + 2\cdot\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_{\rm r},B}(k) && (14)\\
&= \frac{1}{2} + \Pr\left[\,\mathrm{G}_3^A \text{ sets } \mathsf{bad}\,\right] + 2\cdot\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_{\rm r},B}(k)\ . && (15)
\end{aligned}$$

Unless otherwise indicated, the **Initialize**, **RevealPK**, and **Finalize** procedures used in each game are those shown at the top of Figure 7. Game G_0 (boxed statement included) implements game $\mathrm{CDA}^{A}_{\mathcal{AE},k}$, justifying (11). Game G_1 excludes the boxed statement, which ensured consistent responses for hash queries associated to challenge points. Since games G_0 and G_1 are identical-until-bad we can apply the fundamental lemma of game-playing [8] to derive (12). In G_1 removal of the boxed statement means that the coins $\mathbf{r}_c[i]$ used with $\mathcal{E}_{\rm r}$ are not used elsewhere in the game — the table entries $\mathtt{H}[P,R,M]$ storing values $\mathbf{r}_c[i]$ are never referred to again because $P = pk$. Game G_2 (boxed statement omitted) is the same as G_1 except that queries to **Hash** made in the **LR** procedure are handled directly. This implements the same functionality as in G_1, justifying (13). In this game it is clear that $\mathcal{E}_{\rm r}$ uses randomness not associated with any hash query. Game G_3 is the same as G_2 except that, now, challenge queries are responded to by encrypting all zero messages.

We justify (14) using two IND-CPA adversaries B_1 and B_2 that each make qv queries. Both adversaries run A, simulating for A exactly game G_2 in the case that the IND-CPA challenge bit is one and simulating G_3 in the case that the IND-CPA challenge bit is zero. It does this using its own **LR** oracle to answer the A's challenge queries. Adversary B_1 uses the output bit of A to determine the IND-CPA challenge bit. Adversary B_2 uses whether bad was set to true in the course of the game; if so it guesses that the IND-CPA challenge bit was set to 1. By construction then we have that

$$\Pr\left[\,\mathrm{G}_2^A \Rightarrow \mathsf{true}\,\right] = \Pr\left[\mathrm{IND1}^{B_1}_{\mathcal{AE}_{\rm r},k} \Rightarrow 1\right] \quad \text{and} \quad \Pr\left[\,\mathrm{G}_3^A \Rightarrow \mathsf{true}\,\right] = \Pr\left[\mathrm{IND0}^{B_1}_{\mathcal{AE}_{\rm r},k} \Rightarrow 1\right]$$

and that

$$\Pr\left[\,\mathrm{G}_2^A \text{ sets } \mathsf{bad}\,\right] = \Pr\left[\mathrm{IND1}^{B_2}_{\mathcal{AE}_{\rm r},k} \Rightarrow 1\right] \quad \text{and} \quad \Pr\left[\,\mathrm{G}_3^A \text{ sets } \mathsf{bad}\,\right] = \Pr\left[\mathrm{IND0}^{B_2}_{\mathcal{AE}_{\rm r},k} \Rightarrow 1\right]\ .$$

Let B choose $d \leftarrow\!\!\$\ \{1,2\}$ and execute B_d. Then a standard argument justifies (14).

In game G_3 the responses to queries by adversary A are independent of the challenge bit b. Thus $\Pr[\mathrm{G}_3^A \Rightarrow \mathsf{true}] = 1/2$, justifying (15).

All that remains is to bound the probability that bad is set in game G_3. In game G_3 all query responses are independent of the outputs of $\mathcal{M}$. Thus we can delay executing $\mathcal{M}$: in game G_4 (Figure 9) all $\mathcal{M}$

```
procedure Initialize($1^k$):                                          $G_0 - G_5$
$par_r \leftarrow\$ \mathcal{P}_r(1^k)$ ; $(pk_r, sk_r) \leftarrow\$ \mathcal{K}_r(par_r)$ ; $K \leftarrow\$ \{0,1\}^\kappa$
$pk \leftarrow (pk_r, K)$ ; $b \leftarrow\$ \{0,1\}$
```

```
procedure RevealPK():                      $G_0 - G_3$
pkout $\leftarrow$ true ; Ret $pk$

procedure Finalize($b'$):                  $G_0 - G_3$
Ret $(b = b')$
```

```
procedure LR($\boldsymbol{\mathcal{M}}$):
$c \leftarrow c + 1$
$(\mathbf{m}_0^*, \mathbf{m}_1^*, \mathbf{r}^*) \leftarrow\$ \boldsymbol{\mathcal{M}}^{\mathbf{Hash}}(1^k)$
$\mathbf{m}_c \leftarrow \mathbf{m}_b^*$ ; $\mathbf{r}_c \leftarrow \mathbf{r}^*$
For $i = 1 \ldots v$ do
    $\mathbf{r}_c'[i] \leftarrow \mathbf{Hash}(pk, \mathbf{r}_c[i], \mathbf{m}_c[i])$
Ret $\mathcal{E}(pk_r, \mathbf{m}_c ; \mathbf{r}_c')$

procedure Hash($P, R, M$):                 $\boxed{G_0}$, $G_1$
$Y \leftarrow\$ \{0,1\}^\rho$
If $P = pk \wedge \mathtt{H}[P,R,M] \neq \bot$ then
    bad $\leftarrow$ true ; $\boxed{Y \leftarrow \mathtt{H}[P,R,M]}$
If $P \neq pk \wedge \mathtt{H}[P,R,M] \neq \bot$ then $Y \leftarrow \mathtt{H}[P,R,M]$
Ret $\mathtt{H}[P,R,M] \leftarrow Y$
```

```
procedure LR($\boldsymbol{\mathcal{M}}$):
$c \leftarrow c + 1$
$(\mathbf{m}_0^*, \mathbf{m}_1^*, \mathbf{r}^*) \leftarrow\$ \boldsymbol{\mathcal{M}}^{\mathbf{Hash}}(1^k)$
$\mathbf{m}_c \leftarrow \mathbf{m}_b^*$ ; $\mathbf{r}_c \leftarrow \mathbf{r}^*$
For $i = 1 \ldots v$ do
    $\mathbf{r}_c'[i] \leftarrow\$ \{0,1\}^\rho$ ; $\mathbf{Hash}(pk, \mathbf{r}_c[i], \mathbf{m}_c[i])$
    $\boxed{\mathbf{m}_c[i] \leftarrow\$ \{0,1\}^n}$
Ret $\mathcal{E}(pk_r, \mathbf{m}_c ; \mathbf{r}_c')$

procedure Hash($P, R, M$):                 $G_2$, $\boxed{G_3}$
$Y \leftarrow\$ \{0,1\}^\rho$
If $P = pk \wedge \mathtt{H}[P,R,M] \neq \bot$ then bad $\leftarrow$ true
If $P \neq pk \wedge \mathtt{H}[P,R,M] \neq \bot$ then $Y \leftarrow \mathtt{H}[P,R,M]$
Ret $\mathtt{H}[P,R,M] \leftarrow Y$
```

Figure 7: Games used in the IND-CDA proof for Theorem 6.1.

algorithms are executed in **RevealPK**. The hash queries associated to the resulting challenge message, randomness pairs are deferred until **Finalize**. Any sequence of queries that lead to bad being set in game G_3 results in bad being set in game G_4, meaning that $\Pr[G_3 \text{ sets bad}] = \Pr[G_4 \text{ sets bad}]$.

In game G_5 (Figure 9, boxed statement excluded) we split the setting of bad into two cases. Flag bad_1 is set in the case that pkout has not yet been set, while flag bad_2 is set in the case that pkout has been set to true. Note that for the former we have dropped the requirement that $\mathtt{H}[P,R,M] \neq \bot$ for queries that occur before pkout is set. Moreover, we emphasize that hash queries from an execution of $\mathcal{M}$ happen before pkout is set while the hash queries related to resulting challenge message, randomness pairs happen after pkout is set. Game G_6 is the same as G_5 except that the boxed statement is included. It follows bad_1 being set, however, so G_5 and G_6 are identical-until-bad_1. We have that

$$\begin{aligned}\Pr\left[\, G_4^A \text{ sets bad} \,\right] &\leq \Pr\left[\, G_5^A \text{ sets } \mathsf{bad}_1 \vee G_5^A \text{ sets } \mathsf{bad}_2 \,\right] \\ &= \Pr\left[\, G_6^A \text{ sets } \mathsf{bad}_1 \vee G_6^A \text{ sets } \mathsf{bad}_2 \,\right] \\ &\leq \Pr\left[\, G_6^A \text{ sets } \mathsf{bad}_1 \,\right] + \Pr\left[\, G_6^A \text{ sets } \mathsf{bad}_2 \,\right] \;.\end{aligned}$$

We bound the probability of setting each flag in turn.

Upper bound on setting bad_2. We start with bounding the probability of setting bad_2, which corresponds to a hash query made after the public key is revealed. Note that in game G_6 no query after pkout = true can set bad_2 because of a query made when pkout = false. In particular, no queries made by a message sampler $\boldsymbol{\mathcal{M}}$ set $\mathtt{H}[pk, \cdot, \cdot]$ entries. Thus the setting of bad_2 can only occur because of an A query and a query made in **Finalize** are the same or because two queries in **Finalize** are the same. More formally, the cases are: (1) there exists values $1 \leq u \leq c$ and $1 \leq y \leq v$ such that a previous query $\mathbf{Hash}(P, R, M)$ with $R = \mathbf{r}_u[y]$ and $M = \mathbf{m}_u[y]$ was made by A; or (2) there exists values $1 \leq t < u \leq c$ and $1 \leq x \leq y \leq v$ such that $\mathbf{r}_t[x] = \mathbf{r}_t[y]$ and $\mathbf{m}_u[x] = \mathbf{m}_u[y]$. (Note that $u \neq v$ because we assume that

```
Adversary $B_1(pk_{\mathrm{r}})$:
$b \leftarrow\$ \{0,1\}$ ; $K \leftarrow\$ \{0,1\}^{\kappa}$ ; $pk \leftarrow (pk_{\mathrm{r}}, K)$
Run $A(1^k)$

  On query $\mathbf{Hash}(P, R, M)$:
  $Y \leftarrow\$ \{0,1\}^{\rho}$
  If $P = pk \wedge \mathtt{H}[P, R, M] \neq \perp$ then $\mathsf{bad} \leftarrow \mathsf{true}$
  If $P \neq pk \wedge \mathtt{H}[P, R, M] \neq \perp$ then $Y \leftarrow \mathtt{H}[P, R, M]$
  Ret $\mathtt{H}[P, R, M] \leftarrow Y$

  On query $\mathbf{LR}(\boldsymbol{\mathcal{M}})$:
  $c \leftarrow c + 1$
  $(\mathbf{m}_0^*, \mathbf{m}_1^*, \mathbf{r}^*) \leftarrow\$ \boldsymbol{\mathcal{M}}^{\mathbf{Hash}}(1^k)$
  $\mathbf{m}_c \leftarrow \mathbf{m}_b^*$ ; $\mathbf{r}_c \leftarrow \mathbf{r}^*$
  For $i = 1, \ldots, v$ do
      $\mathbf{Hash}(pk, \mathbf{r}_c[i], \mathbf{m}_c[i])$
      $\mathbf{m}_c[i] \leftarrow\$ \{0,1\}^n$
      $\mathbf{ctxt}[i] \leftarrow \mathbf{LR}_B(\mathbf{m}_c[i], \mathbf{m}_b^*[i])$
  Ret $\mathbf{ctxt}$

  On query $\mathbf{RevealPK}(j)$:
  Ret $pk$

When $A$ halts with output $b'$,
return 1 if $b = b'$ and 0 otherwise.
```

```
Adversary $B_2(pk_{\mathrm{r}})$:
$b \leftarrow\$ \{0,1\}$ ; $K \leftarrow\$ \{0,1\}^{\kappa}$ ; $pk \leftarrow (pk_{\mathrm{r}}, K)$
Run $A(1^k)$

  On query $\mathbf{Hash}(P, R, M)$:
  $Y \leftarrow\$ \{0,1\}^{\rho}$
  If $P = pk \wedge \mathtt{H}[P, R, M] \neq \perp$ then $\mathsf{bad} \leftarrow \mathsf{true}$
  If $P \neq pk \wedge \mathtt{H}[P, R, M] \neq \perp$ then $Y \leftarrow \mathtt{H}[P, R, M]$
  Ret $\mathtt{H}[P, R, M] \leftarrow Y$

  On query $\mathbf{LR}(\boldsymbol{\mathcal{M}})$:
  $c \leftarrow c + 1$
  $(\mathbf{m}_0^*, \mathbf{m}_1^*, \mathbf{r}^*) \leftarrow\$ \boldsymbol{\mathcal{M}}^{\mathbf{Hash}}(1^k)$
  $\mathbf{m}_c \leftarrow \mathbf{m}_b^*$ ; $\mathbf{r}_c \leftarrow \mathbf{r}^*$
  For $i = 1, \ldots, v$ do
      $\mathbf{Hash}(pk, \mathbf{r}_c[i], \mathbf{m}_c[i])$
      $\mathbf{m}_c[i] \leftarrow\$ \{0,1\}^n$
      $\mathbf{ctxt}[i] \leftarrow \mathbf{LR}_B(\mathbf{m}_c[i], \mathbf{m}_b^*[i])$
  Ret $\mathbf{ctxt}$

  On query $\mathbf{RevealPK}(j)$:
  Ret $pk$

When $A$ halts with output $b'$,
return 1 if $\mathsf{bad} = \mathsf{true}$ and 0 otherwise
```

Figure 8: Adversaries used to bound the G_2 to G_3 transition.

A only queries distinct sources $\boldsymbol{\mathcal{M}}$.) Moreover, the adversary A does not learn anything about the coins to run $\boldsymbol{\mathcal{M}}$ in the course of the game. Let "$\mathbf{m}_v[y], \mathbf{r}_v[y]$ $\mathsf{collides}$" be the event that bad_2 was set because of query $\mathbf{Hash}(pk, \mathbf{r}_u[y], \mathbf{m}_u[y])$ made by **Finalize**. Then

$$\Pr\left[\ \mathrm{G}_6^A \text{ sets } \mathsf{bad}_2\ \right] = \Pr\left[\bigvee_{u,y} \mathbf{m}_u[y], \mathbf{r}_u[y] \ \mathsf{collides}\right] \leq \sum_{u,y} \Pr\left[\ \mathbf{m}_u[y], \mathbf{r}_u[y] \ \mathsf{collides}\ \right]$$

where the or and sum are taken over $1 \leq u \leq q$ and $1 \leq y \leq v$. For any particular value u, y there are at most $h + (u-1)v$ points in the table $\mathtt{H}$ which it can collide with. Applying the min-entropy of each $\boldsymbol{\mathcal{M}}$, we therefore have that each probability in the sum is at most $(h + (u-1)v)2^{-\mu}$. Thus

$$\Pr\left[\ \mathrm{G}_6^A \text{ sets } \mathsf{bad}_2\ \right] \leq \sum_{1 \leq u \leq q} \frac{h + (u-1)v}{2^{\mu}} \leq \frac{h + q^2 v}{2^{\mu}}$$

Upper bound on setting bad_1. The setting of bad_1 happens only if A (before pk is revealed) or one of the $\mathcal{M}$ algorithms queries the hash function with $P = (pk, K)$. First consider the case where $q > 1$ (the adaptive setting). Relative to the event space defined by G_6^A, let "$\mathsf{Queries}\ K$" be the event that A queries the hash function on $P = (pk', K)$ for some pk' before the public key is revealed and let "$\mathsf{Queries}\ pk$" be the event that A queries the hash function on $P = (pk, K')$ for some K' before the public key is revealed. Then we will justify that

$$\Pr\left[\ \mathrm{G}_6^A \text{ sets } \mathsf{bad}_1\ \right] \leq \Pr\left[\ \mathsf{Queries}\ K \wedge \mathsf{Queries}\ pk\ \right] \tag{16}$$

in two different ways. First, we point out that the right hand side is less than or equal to both $\Pr[\mathsf{Queries}\ K]$ and $\Pr[\mathsf{Queries}\ pk]$. Since the choice of K is independent of all hash queries made before the public key is revealed, we have that

$$\Pr\left[\ \mathsf{Queries}\ K\ \right] \leq \frac{h}{2^{\kappa}} \tag{17}$$

```
procedure LR($\boldsymbol{\mathcal{M}}$):
$c \leftarrow c + 1$ ; $\boldsymbol{\mathcal{M}}_c \leftarrow \boldsymbol{\mathcal{M}}$
For $i = 1, \ldots, v$ do
    $\mathbf{r}'_c[i] \leftarrow\$ \{0,1\}^\rho$
    $\mathbf{m}_c[i] \leftarrow\$ \{0,1\}^n$
Ret $\mathcal{E}(pk_\mathrm{r}, \mathbf{m}_c \, ; \, \mathbf{r}'_c)$

procedure RevealPK:
For $j = 1, \ldots, c$ do
    $(\mathbf{m}^*_0, \mathbf{m}^*_1, \mathbf{r}^*) \leftarrow\$ \boldsymbol{\mathcal{M}}_j^{\mathbf{Hash}}(1^k)$
    $\mathbf{m}_j \leftarrow \mathbf{m}^*_b$ ; $\mathbf{r}_j \leftarrow \mathbf{r}^*$
pkout $\leftarrow$ true ; Ret $pk$

procedure Hash($P, R, M$):                                         $\mathrm{G}_4$
$Y \leftarrow\$ \{0,1\}^\rho$
If $P = pk \wedge \mathtt{H}[P,R,M] \neq \bot$ then bad $\leftarrow$ true
If $P \neq pk \wedge \mathtt{H}[P,R,M] \neq \bot$ then $Y \leftarrow \mathtt{H}[P,R,M]$
Ret $\mathtt{H}[P,R,M] \leftarrow Y$

procedure Finalize($b'$):
For $j = 1, \ldots, c$ do
    For $i = 1, \ldots, v$ do
        Hash($pk, \mathbf{r}_j[i], \mathbf{m}_j[i]$)
Ret $(b = b')$
```

```
procedure LR($\boldsymbol{\mathcal{M}}$):
$c \leftarrow c + 1$ ; $\boldsymbol{\mathcal{M}}_c \leftarrow \boldsymbol{\mathcal{M}}$
For $i = 1, \ldots, v$ do
    $\mathbf{r}'_c[i] \leftarrow\$ \{0,1\}^\rho$
    $\mathbf{m}_c[i] \leftarrow\$ \{0,1\}^n$
Ret $\mathcal{E}(pk_\mathrm{r}, \mathbf{m}_c \, ; \, \mathbf{r}'_c)$

procedure RevealPK:
For $j = 1, \ldots, c$ do
    $(\mathbf{m}^*_0, \mathbf{m}^*_1, \mathbf{r}^*) \leftarrow\$ \boldsymbol{\mathcal{M}}_j^{\mathbf{Hash}}(1^k)$
    $\mathbf{m}_j \leftarrow \mathbf{m}^*_b$ ; $\mathbf{r}_j \leftarrow \mathbf{r}^*$
pkout $\leftarrow$ true ; Ret $pk$

procedure Hash($P, R, M$):                                         $\mathrm{G}_5$ $\boxed{\mathrm{G}_6}$
$Y \leftarrow\$ \{0,1\}^\rho$
If pkout = false $\wedge P = pk$ then
    bad$_1$ $\leftarrow$ true $\boxed{; \mathrm{Ret}\ Y}$
    Ret $\mathtt{H}[pk,R,M] \leftarrow Y$
else if pkout = true $\wedge P = pk$ then
    If $\mathtt{H}[pk,R,M] \neq \bot$ then bad$_2$ $\leftarrow$ true
    Ret $\mathtt{H}[pk,R,M] \leftarrow Y$
If $P \neq pk \wedge \mathtt{H}[P,R,M] \neq \bot$ then $Y \leftarrow \mathtt{H}[P,R,M]$
Ret $\mathtt{H}[P,R,M] \leftarrow Y$

procedure Finalize($b'$):
For $j = 1, \ldots, c$ do
    For $i = 1, \ldots, v$ do
        Hash($pk, \mathbf{r}_j[i], \mathbf{m}_j[i]$)
Ret $(b = b')$
```

Figure 9: Games used to bound the setting of bad in game G_3 of Figure 7.

To bound Pr[Queries pk] we build a KR-UMA adversary C, as shown in Figure 10. Adversary C implements G_6^A except that: C halts after **RevealPK** is queried; C forwards the public-key portion of **Hash** queries to its **Check** oracle; and C uses it's **Enc** oracle to answer **LR** queries. Let "A queries pk" be the event that one of A's **Hash** queries included a value $P = (pk, K)$ where pk is the one chosen by the $\text{KR-UMA}_{\mathcal{AE}_\mathrm{r},k}$ experiment (the event being defined in the probability space defined by $\text{KR-UMA}^C_{\mathcal{AE}_\mathrm{r},k}$). Then we have that

$$\mathbf{Adv}^{\text{kr-uma}}_{\mathcal{AE}_\mathrm{r},k}(C) = \Pr\left[\,\text{KR-UMA}^C_{\mathcal{AE}_\mathrm{r},k} \Rightarrow \mathsf{true}\,\right] = \Pr\left[\,A \text{ queries } pk\,\right] = \Pr\left[\,\text{Queries } pk\,\right] . \tag{18}$$

These equalities are justified by: (1) by construction, the probability that adversary A queries the appropriate pk in $\text{KR-UMA}_{\mathcal{AE}_\mathrm{r},k}$ and the probability that it queries the appropriate pk in G_6^A are the same; and (2) such a query must occur for the event "Queries pk" to be occur in G_6.

Thus, combining Equation 17 with 16 and Equation 18 with 16 gives that

$$\Pr\left[\,\mathrm{G}_6^A \text{ sets bad}_1\,\right] \leq \min\left\{\frac{h}{2^\kappa}, \mathbf{Adv}^{\text{kr-uma}}_{\mathcal{AE}_\mathrm{r},C}(k)\right\} .$$

We now turn to the case in which $q = 1$, which means that A is non-adaptive. Recall that for non-adaptive security, we can assume without loss that A queries **RevealPK** immediately after its single query to **LR**. Thus no hash queries are affected by the output of the **LR** query, and, in turn, by the public key $\mathcal{AE}_\mathrm{r}$. The probability of any individual hash query (P, R, M) having $P = (pk, K)$ is thus at

```
adversary $C(1^k)$:
Run $A(1^k)$

  On query $\mathbf{Hash}(P, R, M)$:
  $Y \leftarrow_\$ \{0,1\}^\rho$
  $d \leftarrow \mathbf{Check}(pk^*, K^*)$
  If $d = 0 \wedge \mathtt{H}[P, R, M] \neq \bot$ then $Y \leftarrow \mathtt{H}[P, R, M]$
  Ret $\mathtt{H}[P, R, M] \leftarrow Y$

  On query $\mathbf{LR}(\mathcal{M})$:
  $c \leftarrow c + 1$ ; $\boldsymbol{\mathcal{M}}_c \leftarrow \boldsymbol{\mathcal{M}}$
  For $i = 1, \ldots, v$ do
      $\mathbf{c}[i] \leftarrow \mathbf{Enc}$
  Ret $\mathbf{c}$

  On query $\mathbf{RevealPK}$:
  For $j = 1, \ldots, c$ do
      $(\mathbf{m}_0^*, \mathbf{m}_1^*, \mathbf{r}^*) \leftarrow_\$ \boldsymbol{\mathcal{M}}_j^{\mathbf{Hash}}(1^k)$
  Halt with no output
```

Figure 10: Adversary used to bound the setting of bad_1 in game G_6 of Figure 9.

most $2^{-\kappa} \cdot \mathsf{maxpk}_{\mathcal{AE}_\mathrm{r}}(k)$. We therefore can conclude that

$$\Pr\left[\ \mathrm{G}_6^A \text{ sets } \mathsf{bad}_1\ \right] \leq \frac{h \cdot \mathsf{maxpk}_{\mathcal{AE}_\mathrm{r}}(k)}{2^{-\kappa}} \ .$$

▮

6.2 Hedging via Composition

For the following, let $\mathcal{AE}_\mathrm{r} = (\mathcal{P}_\mathrm{r}, \mathcal{K}_\mathrm{r}, \mathcal{E}_\mathrm{r}, \mathcal{D}_\mathrm{r})$ be a (randomized) PKE scheme with message length $n_\mathrm{r}(\cdot)$ and randomness length $\rho(\cdot)$. Let $\mathcal{AE}_\mathrm{d} = (\mathcal{P}_\mathrm{d}, \mathcal{K}_\mathrm{d}, \mathcal{E}_\mathrm{d}, \mathcal{D}_\mathrm{d})$ be a (deterministic) PKE scheme with message length $n_\mathrm{d}(\cdot)$ and randomness length always 0. Associate to $\mathcal{AE}_c$ for $c \in \{\mathrm{d}, \mathrm{r}\}$ the function $\mathsf{maxclen}_c(k)$ mapping any k to the maximum length (over all possible public keys, messages, and if applicable, randomness) of a ciphertext output by $\mathcal{E}_c$.

Deterministic-then-randomized. Our first attempt is to perform hedged encryption via applying deterministic encryption and then randomized. More formally let $\mathsf{DtR}[\mathcal{AE}_\mathrm{r}, \mathcal{AE}_\mathrm{d}] = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ with randomness length ρ and message length n_d be the scheme that works as follows. Parameter generation algorithm $\mathcal{P}$ runs $par_\mathrm{r} \leftarrow_\$ \mathcal{P}_\mathrm{r}(1^k)$ and $par_\mathrm{d} \leftarrow_\$ \mathcal{P}_\mathrm{d}(1^k)$ and outputs $par = (par_\mathrm{r}, par_\mathrm{d})$. Key generation $\mathcal{K}$ just runs $(pk_\mathrm{r}, sk_\mathrm{r}) \leftarrow_\$ \mathcal{K}_\mathrm{r}(par_\mathrm{r})$ and $(pk_\mathrm{d}, sk_\mathrm{d}) \leftarrow_\$ \mathcal{K}_\mathrm{d}(par_\mathrm{d})$ and outputs $pk = (pk_\mathrm{r}, pk_\mathrm{d})$ and $sk = (sk_\mathrm{r}, sk_\mathrm{d})$. We define encryption by

$$\mathcal{E}((pk_\mathrm{r}, pk_\mathrm{d}), m\ ;\ r) = \mathcal{E}_\mathrm{r}(pk_\mathrm{r}, c \parallel 10^\ell\ ;\ r)\ ,$$

where $c = \mathcal{E}_\mathrm{d}(pk_\mathrm{d}, m)$ and $\ell = n_\mathrm{r} - |c| - 1$. Here we need that $n_\mathrm{r}(k) > \mathsf{maxclen}_\mathrm{d}(k)$ for all k. Decryption is defined in the natural way.

The scheme will clearly inherit IND-CPA security from the application of $\mathcal{E}_\mathrm{r}$. If the deterministic encryption scheme is PRIV secure for min-entropy μ, then the composition will also be secure if the *message* has min-entropy at least μ. However, our strong notion of IND-CDA security requires that schemes be secure if the *joint* distribution on the message and randomness has high min-entropy. If the entropy is unfortuitously split between both the randomness and the message, then there is no guarantee that the composition will be secure.

Randomized-then-deterministic. We can instead apply randomized encryption first, and then deterministic encryption. Define $\mathsf{RtD}[\mathcal{AE}_{\rm r}, \mathcal{AE}_{\rm d}] = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ with randomness length ρ and message length $n_{\rm r}$ to work as follows. The parameter and key generation algorithms are as for scheme DtR. Encryption is defined by

$$\mathcal{E}((pk_{\rm r}, pk_{\rm d}), m\,;\, r) = \mathcal{E}_{\rm d}(pk_{\rm d}, c \,\|\, 10^{\ell})\ .$$

where $c = \mathcal{E}_{\rm r}(pk_{\rm r}, m\,;\, r)$ and $\ell = n_{\rm d} - |c| - 1$. Here we need that $n_{\rm d}(k) > \mathsf{maxclen}_{\rm r}(k)$ for all k. The decryption algorithm $\mathcal{D}$ works in the natural way. As we will see below, this construction avoids the security issues of the previous, as long as the randomized encryption scheme preserves the min-entropy of its inputs. (For example, if for all k, all $par_{\rm r} \in [\mathcal{P}_{\rm r}(1^k)]$, and all $(pk_{\rm r}, sk_{\rm r}) \in [\mathcal{K}_{\rm r}(par_{\rm r})]$, $\mathcal{E}_{\rm r}(pk_{\rm r}, \cdot)$ is injective in (m, r).) Many encryption schemes have this property; El Gamal [22] is one example.

Security of RtD. Intuitively, the hedged security of the RtD construction is inherited from the IND-CPA security of the underlying randomized scheme $\mathcal{AE}_{\rm r}$ and the PRIV security of the underlying deterministic scheme $\mathcal{AE}_{\rm d}$. As alluded to before, we have one technical requirement on $\mathcal{AE}_{\rm r}$ for the IND-CDA proof to work. We say $\mathcal{AE}_{\rm r} = (\mathcal{P}_{\rm r}, \mathcal{K}_{\rm r}, \mathcal{E}_{\rm r}, \mathcal{D}_{\rm r})$ with message length $n_{\rm r}(\cdot)$ and randomness length $\rho(\cdot)$ is one-to-one (1-1) if for any k, any $par_{\rm r} \in [\mathcal{P}_{\rm r}(1^k)]$, and any $(pk_{\rm r}, sk_{\rm r}) \in [\mathcal{K}_{\rm r}(par_{\rm r})]$ their do not exist $(m, r) \neq (m', r')$ such that $\mathcal{E}_{\rm r}(pk_{\rm r}, m\,;\, r) = \mathcal{E}_{\rm r}(pk_{\rm r}, m'\,;\, r')$. Being 1-1 gives us two properties needed by the scheme. First, it implies that the equality pattern of a vector of inputs to $\mathcal{E}_{\rm r}$ is equal to the equality pattern of the vector encrypted under the same key. Second, it implies that min-entropy is preserved, meaning for any k, any $par_{\rm r} \in [\mathcal{P}_{\rm r}(1^k)]$, any $(pk_{\rm r}, sk_{\rm r}) \in [\mathcal{K}_{\rm r}(par_{\rm r})]$, and for all $c \in \{0,1\}^*$ and any $(\mu, 1, n_{\rm r}, \rho)$-$\mathsf{mr}$-source $\boldsymbol{\mathcal{M}}$ it holds that $\Pr\left[\, c = \mathcal{E}_{\rm r}(pk_{\rm r}, m\,;\, r) \;:\; (m, r) \leftarrow_{\$} \boldsymbol{\mathcal{M}}(1^k) \,\right] = 2^{-\mu}$. We have the following theorem.

Theorem 6.2 [**RtD is H-IND secure**] Let $\mathcal{AE}_{\rm r} = (\mathcal{P}_{\rm r}, \mathcal{K}_{\rm r}, \mathcal{E}_{\rm r}, \mathcal{D}_{\rm r})$ be a 1-1 PKE scheme with message length $n_{\rm r}(\cdot)$ and randomness length $\rho(\cdot)$. Let $\mathcal{AE}_{\rm d} = (\mathcal{P}_{\rm d}, \mathcal{K}_{\rm d}, \mathcal{E}_{\rm d}, \mathcal{D}_{\rm d})$ be a (deterministic) encryption scheme with message length $n_{\rm d}(\cdot)$ so that $n_{\rm d}(\cdot) \geq \mathsf{maxclen}_{\rm r}(\cdot)$. Let $\mathcal{AE} = \mathsf{RtD}[\mathcal{AE}_{\rm r}, \mathcal{AE}_{\rm d}] = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be the PKE scheme defined above.

- (IND-CPA) Let A be an IND-CPA adversary. Then there exists an IND-CPA adversary B such that for any k

  $$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},A}(k) = \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}_{\rm r},B}(k)$$

  where B runs in time that of A plus the time to run $\mathcal{E}_{\rm d}$ once.
- (IND-CDA) Let A be a CDA adversary that makes at most q LR queries, each consisting of a distinct $(\mu, v, n_{\rm r}, \rho)$-$\mathsf{mmr}$-source (resp. block-source). Then there exists a PRIV adversary B such that for any k

  $$\mathbf{Adv}^{\text{cda}}_{\mathcal{AE},A}(k) \leq \mathbf{Adv}^{\text{priv}}_{\mathcal{AE}_{\rm d},B}(k)$$

  where B runs in time that of A plus the time to run at most $q(k) \cdot v(k)$ executions of $\mathcal{E}_{\rm r}$ and makes at most q LR queries each consisting of a distinct $(\mu, v, \mathsf{maxclen}_{\rm r})$-$\mathsf{mm}$-source (resp. block-source). □

Note that the second part of the theorem states the result for either sources or just block-sources. Before proving in more detail, we give a sketch. The first part of the theorem is immediate from the IND-CPA security of $\mathcal{AE}_{\rm r}$. For the second part, any mmr-source $\boldsymbol{\mathcal{M}}$ queried by A is converted into an mm-source $\boldsymbol{\mathcal{M}}'$ to be queried by B. This is done by having $\boldsymbol{\mathcal{M}}'$ run $\boldsymbol{\mathcal{M}}$ to get $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r})$ and then outputting the pair of vectors $(\mathcal{E}_{\rm r}(pk, \mathbf{m}_0\,;\, \mathbf{r}), \mathcal{E}_{\rm r}(pk, \mathbf{m}_1\,;\, \mathbf{r}))$. (The ciphertexts are the "messages" for $\mathcal{E}_{\rm d}$.) Because $\mathcal{AE}_{\rm r}$ is 1-1, $\boldsymbol{\mathcal{M}}'$ is a source of the appropriate type.

Proof of Theorem 6.2: We first show IND-CPA security. Let A be an IND-CPA adversary against $\mathcal{AE} = \mathsf{RtD}[\mathcal{AE}_{\rm r}, \mathcal{AE}_{\rm d}] = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where $\mathcal{AE}_{\rm r} = (\mathcal{P}_{\rm r}, \mathcal{K}_{\rm r}, \mathcal{E}_{\rm r}, \mathcal{D}_{\rm r})$ is a randomized PKE scheme and $\mathcal{AE}_{\rm d} = (\mathcal{P}_{\rm d}, \mathcal{K}_{\rm d}, \mathcal{E}_{\rm d}, \mathcal{D}_{\rm d})$ is a deterministic PKE scheme with plaintext length $n_{\rm d}$. We build an IND-CPA adversary B against $\mathcal{AE}_{\rm r}$ using A; the adversary is shown in Figure 11. Adversary B, on input $pk_{\rm r}$, runs

```
Adversary $B(1^k, pk_{\mathrm{r}})$
$par \leftarrow\$ \mathcal{P}_{\mathrm{d}}(1^k)$
$(pk_{\mathrm{d}}, sk_{\mathrm{d}}) \leftarrow\$ \mathcal{K}_{\mathrm{d}}(par)$ ; $pk \leftarrow (pk_{\mathrm{r}}, pk_{\mathrm{d}})$
Run $A(1^k, pk)$.
  On query $\mathbf{LR}(m_0, m_1)$:
  $c \leftarrow \mathbf{LR}_B(m_0, m_1)$
  $\ell \leftarrow n_{\mathrm{d}} - |c| - 1$
  $c' \leftarrow \mathcal{E}_{\mathrm{d}}(pk_{\mathrm{d}}, c \,\|\, 10^{\ell})$
  Ret. $c'$
When $A$ halts with output $b'$, halt and output $b'$.
```

```
Adversary $B(1^k)$
$par \leftarrow\$ \mathcal{P}_{\mathrm{r}}(1^k)$
$(pk_{\mathrm{r}}, sk_{\mathrm{r}}) \leftarrow\$ \mathcal{K}_{\mathrm{r}}(par)$
Run $A(1^k)$.
  On query $\mathbf{RevealPK}()$:
  $pk_{\mathrm{d}} \leftarrow \mathbf{RevealPK}_B()$
  $pk \leftarrow (pk_{\mathrm{r}}, pk_{\mathrm{d}})$
  Ret. $pk$
  On query $\mathbf{LR}(\boldsymbol{\mathcal{M}})$:
  $\mathbf{c} \leftarrow \mathbf{LR}_B(\boldsymbol{\mathcal{M}}^*(\boldsymbol{\mathcal{M}}))$
  Ret. $\mathbf{c}$
  $\boldsymbol{\mathcal{M}}^*(\boldsymbol{\mathcal{M}})$:
  $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ \boldsymbol{\mathcal{M}}$
  For $i$ in 1 to $|\mathbf{m}_0|$ do:
    $\ell_0 \leftarrow n_{\mathrm{d}} - |\mathcal{E}_{\mathrm{r}}(pk_{\mathrm{r}}, \mathbf{m}_0[i] \,;\, \mathbf{r}[i])| - 1$
    $\mathbf{x}_0[i] \leftarrow \mathcal{E}_{\mathrm{r}}(pk_{\mathrm{r}}, \mathbf{m}_0[i] \,;\, \mathbf{r}[i]) \,\|\, 10^{\ell_0}$
    $\ell_1 \leftarrow n_{\mathrm{d}} - |\mathcal{E}_{\mathrm{r}}(pk_{\mathrm{r}}, \mathbf{m}_1[i] \,;\, \mathbf{r}[i])| - 1$
    $\mathbf{x}_1[i] \leftarrow \mathcal{E}_{\mathrm{r}}(pk_{\mathrm{r}}, \mathbf{m}_1[i] \,;\, \mathbf{r}[i]) \,\|\, 10^{\ell_1}$
  Ret. $(\mathbf{x}_0, \mathbf{x}_1)$
When $A$ halts with output $b'$, halt and output $b'$.
```

Figure 11: Adversaries for the proof of Theorem 6.2.

$\mathcal{P}_{\mathrm{d}}$ and $\mathcal{K}_{\mathrm{d}}$ to generate a keypair $(pk_{\mathrm{d}}, sk_{\mathrm{d}})$ for the deterministic PKE scheme. It then runs adversary A with public key $(pk_{\mathrm{r}}, pk_{\mathrm{d}})$. When A queries **LR** with a pair of messages (m_0, m_1), B forwards the query to its own **LR** oracle. When B receives ciphertext c, it returns to adversary A the encryption $\mathcal{E}_{\mathrm{d}}(pk_{\mathrm{d}}, c \,\|\, 10^{\ell})$, where $\ell = n_{\mathrm{d}} - |c| - 1$ is the amount of padding necessary to make c fit in the plaintext space of $\mathcal{AE}_{\mathrm{d}}$. When A outputs a guess bit b', B also outputs this same guess. It is easy to see that the simulation is perfect and the advantages are equal.

We next show IND-CDA security. Let A be a CDA adversary making q **LR** queries, each a v-vector $(\mu, n_{\mathrm{r}}, \rho)$-source, and attacking $\mathcal{AE}$ constructed as in the theorem statement from $\mathcal{AE}_{\mathrm{r}}$ and $\mathcal{AE}_{\mathrm{d}}$. We will build PRIV adversary B against $\mathcal{AE}_{\mathrm{d}}$ as follows; the adversary is shown on the right side of Figure 11. Adversary B, at the start of the game, runs $\mathcal{P}_{\mathrm{r}}$ and $\mathcal{K}_{\mathrm{r}}$ to generate keys $(pk_{\mathrm{r}}, sk_{\mathrm{r}})$ for the randomized encryption scheme. It then runs adversary A and answers queries as follows. On query **RevealPK**, B queries its own **RevealPK** adversary to learn pk_{d} and then returns $(pk_{\mathrm{r}}, pk_{\mathrm{d}})$ to A. On query $\mathbf{LR}(\boldsymbol{\mathcal{M}})$ for mmr-source (resp. block-source) $\boldsymbol{\mathcal{M}}$, B constructs mm-source (resp. block-source) $\boldsymbol{\mathcal{M}}^*$ so as to run $\boldsymbol{\mathcal{M}}$ to get vector $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r})$ and then output the pair

$$(\mathcal{E}_{\mathrm{r}}(pk_{\mathrm{r}}, \mathbf{m}_0 \,;\, \mathbf{r}), \mathcal{E}_{\mathrm{r}}(pk_{\mathrm{r}}, \mathbf{m}_1 \,;\, \mathbf{r}))\ ,$$

where each component in each vector in the pair is padded out with a single 1 followed by the appropriate number of 0s to make it length n_{d}. The details of $\boldsymbol{\mathcal{M}}^*$ are shown in Figure 11. Since $\mathcal{AE}_{\mathrm{r}}$ is 1-1, it follows that $\boldsymbol{\mathcal{M}}^*$ is a distinct source (resp. block-source) with the appropriate min-entropy.

Finally, when A halts with guess bit b', B outputs the same guess. Again, it is easy to see the simulation is perfect. ▌

6.3 Hedging by Randomizing Deterministic Encryption

For the following, let $\mathcal{AE}_{\mathrm{r}} = (\mathcal{P}_{\mathrm{r}}, \mathcal{K}_{\mathrm{r}}, \mathcal{E}_{\mathrm{r}}, \mathcal{D}_{\mathrm{r}})$ be a (randomized) PKE scheme with message length $n_{\mathrm{r}}(\cdot)$ and randomness length $\rho(\cdot)$. Let $\mathcal{AE}_{\mathrm{d}} = (\mathcal{P}_{\mathrm{d}}, \mathcal{K}_{\mathrm{d}}, \mathcal{E}_{\mathrm{d}}, \mathcal{D}_{\mathrm{d}})$ be a (deterministic) PKE scheme with message length $n_{\mathrm{d}}(\cdot)$ and randomness length always 0. Associate to $\mathcal{AE}_c$ for $c \in \{\mathrm{d}, \mathrm{r}\}$ the function $\mathsf{maxclen}_c(k)$ mapping any k to the maximum length (over all possible public keys, messages, and if applicable, randomness) of a ciphertext output by $\mathcal{E}_c$.

Pad-then-Deterministic. Our final construction dispenses entirely with the need for a dedicated randomized encryption scheme, instead using simple padding to directly construct a (randomized) encryption scheme from a deterministic one. Scheme $\mathsf{PtD}[\mathcal{AE}_{\mathrm{d}}] = (\mathcal{P}_{\mathrm{d}}, \mathcal{K}_{\mathrm{d}}, \mathcal{E}, \mathcal{D})$ with randomness length ρ and message length n works as follows. Parameter and key generation are inherited form the underlying (deterministic) encryption scheme. Encryption is defined by

$$\mathcal{E}(pk_{\mathrm{d}}, m\ ;\ r) = \mathcal{E}_{\mathrm{d}}(pk_{\mathrm{d}}, r \parallel m)$$

where we require that $n_{\mathrm{d}}(k) \geq \rho(k) + n(k)$. Decryption proceeds by applying $\mathcal{D}_{\mathrm{d}}$, to retrieve $r \parallel m$, and then returning m.

Security. The IND-CDA security of PtD is inherited immediately from the PRIV security of the $\mathcal{AE}_{\mathrm{d}}$ scheme. The more challenging part is proving the IND-CPA security. For this we will need a stronger assumption on the underlying deterministic encryption scheme — that it is a u-LTDF.

Theorem 6.3 [PtD **is H-IND secure**] Let $\mathcal{AE}_{\mathrm{d}} = (\mathcal{P}_{\mathrm{d}}, \mathcal{K}_{\mathrm{d}}, \mathcal{E}_{\mathrm{d}}, \mathcal{D}_{\mathrm{d}})$ be a deterministic encryption scheme with message length $n_{\mathrm{d}}(\cdot)$. Let $\mathcal{AE} = \mathsf{PtD}[\mathcal{AE}_{\mathrm{d}}] = (\mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be the PKE scheme defined above with message length $n(\cdot)$ and randomness length $\rho(\cdot)$ such that $n(k) = n_{\mathrm{d}}(k) - \rho(k)$ for all k.

- (IND-CPA) Let $\mathcal{K}_l$ be a universal-inducing (n_{d}, ℓ)-lossy key generation algorithm for $\mathcal{AE}_{\mathrm{d}}$. Let A be an IND-CPA adversary. Then there exists a LOS adversary B such that for all k
$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},A}(k) \leq \mathbf{Adv}^{\text{los}}_{\mathcal{AE}_{\mathrm{d}},\mathcal{K}_l,B}(k) + \sqrt{2^{3n(k)-\ell(k)+2}}\ .$$
B runs in time that of A.
- (IND-CDA) Let A be a CDA adversary that makes at most q LR queries each consisting of a distinct (μ, v, n, ρ)-mmr-source (resp. block-source). Then there exists a PRIV adversary B such that for all k
$$\mathbf{Adv}^{\text{cda}}_{\mathcal{AE},A}(k) \leq \mathbf{Adv}^{\text{priv}}_{\mathcal{AE}_{\mathrm{d}},B}(k)$$
where B runs in time that of A and makes at most q LR queries each consisting of a distinct (μ, v, n_{d})-mm-source (resp. block-source). $\square$

One might think that concluding IND-CPA can be based just on PtD being IND-CDA secure, since the padded randomness provides high min-entropy. However, this approach does not work because an IND-CPA adversary expects knowledge of the public-key *before* making any **LR** queries, while a CDA adversary only learns the public-key *after* making its **LR** queries. This issue, which also arised in another context, is discussed in more detail in [7]. We use a different approach (which may be of independent interest) to prove this part of Theorem 6.3. Intuitively, our proof strategy corresponds to using the standard LHL $2^{n(k)}$ times, once for each possible message the IND-CPA adversary might query.

Proof of Theorem 6.3: We first briefly prove IND-CDA. Let A be a CDA adversary against $\mathcal{AE}$. We can easily construct a PRIV adversary B against $\mathcal{AE}_{\mathrm{d}}$. B runs A and on **LR** query $\boldsymbol{\mathcal{M}}$, a distinct $(\mu, v, n_{\mathrm{r}}, \rho)$-mmr source (resp. block-source) queries $\boldsymbol{\mathcal{M}}'$ that samples from $\boldsymbol{\mathcal{M}}$ to get $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r})$ and outputs $((\mathbf{m}_0, \mathbf{r}), (\mathbf{m}_1, \mathbf{r}))$. This results in a distinct (μ, v, n_{d}) mm-source (resp. block-source), where $n_{\mathrm{d}} = n_{\mathrm{r}} + \rho$. The simulation is perfect and security follows.

Next we show IND-CPA. Let A be an IND-CPA adversary against PtD. We will go through a series of game transitions to prove the theorem. Game G_0 is simply the IND-CPA game, so by definition $\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{PtD},A}(k) = 2 \cdot \Pr\left[\, G_0^A(k) \,\right] - 1$. Game G_1 is identical to G_0 except that **Initialize** uses the lossy key generation algorithm $\mathcal{K}_l$. We will define a LOS adversary B such that

$$\Pr\left[\, G_0^A(k) \Rightarrow \mathsf{true} \,\right] - \Pr\left[\, G_1^A(k) \Rightarrow \mathsf{true} \,\right] \leq \mathbf{Adv}^{\text{los}}_{\mathcal{AE},\mathcal{K}_l,B}(k)\ .$$

Adversary B, shown in Figure 12, when given a public key pk that is either from $\mathcal{K}_{\mathrm{d}}$ or $\mathcal{K}_l$, simply runs adversary A as in games G_0 and G_1 with pk; if there is a gap between A's success probability in games

```
Adversary $B(1^k, pk)$
$b \leftarrow_\$ \{0,1\}^*$
Run $A(1^k, pk)$.
  On query $\mathbf{LR}(m_0, m_1)$:
  $c \leftarrow_\$ \mathcal{E}(pk, m_b)$
  Ret. $c$
When $A$ halts with output $b'$, halt and output $(b = b')$.
```

```
Adversary $C(1^k, par)$
$b \leftarrow_\$ \{0,1\}$
For $m$ in $0^{n_\mathrm{r}}$ to $1^{n_\mathrm{r}}$:
    $\mathbf{c}[m] \leftarrow \mathbf{RoR}(\boldsymbol{\mathcal{M}}_m)$
$pk \leftarrow_\$ \mathbf{RevealPK}()$
Run $A(1^k)$.
  On query $\mathbf{LR}(m_0, m_1)$:
  $c \leftarrow \mathbf{c}[m_b]$
  Ret. $c$
  $\boldsymbol{\mathcal{M}}_m$:
  $r \leftarrow_\$ \{0,1\}^{n_\mathrm{d} - n_\mathrm{r}}$
  Ret. $r \,\|\, m$
When $A$ halts with output $b'$, halt and output $(b = b')$.
```

Figure 12: Adversaries for the proof of Theorem 6.3

G_0 and G_1, then B will be able to distinguish whether the key is lossy or not.

Game G_2 is the same as G_1 except that **LR** returns a uniform element from the range of the hash function (instead of returning the encryption of m_b). We claim that there is an unbounded ALH adversary C such that

$$\Pr\left[\, G_1^A(k) \Rightarrow \mathsf{true} \,\right] - \Pr\left[\, G_2^A(k) \Rightarrow \mathsf{true} \,\right] \le \mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},C}(k)\ ,$$

where $\mathcal{H} = (\mathcal{K}_l, \mathcal{E}_\mathrm{d})$ is a universal family of hash functions. The LH adversary C, shown in Figure 12, proceeds as follows. First, C makes $q = 2^n$ queries to its **RoR** oracle, where $\{0,1\}^n$ is the plaintext space of PtD. The ith **RoR** query is an m-source of vector length 1 consisting of a uniform string of $n_\mathrm{d} - n$ bits concatenated with m^i, the ith message in the plaintext space $\{0,1\}^n$ according to some known ordering on $\{0,1\}^n$ (i.e., lexigraphical order). It is easy to see that due to the padded uniform bits, each m-source has min-entropy of at least $n_\mathrm{d} - n$ bits, even conditioned on all the previous queries. Let the answers C receives to its q **RoR** queries be called $y_1, \ldots, y_q$. Next, C calls oracle **RevealPK** and learns pk'. At this point, C runs adversary A as in games G_1 and G_2, flipping a bit b and giving A the public key pk'. On oracle query $\mathbf{LR}(m_0, m_1)$ from A, C finds the j such that $m_b = m^j$, the jth message in the plaintext space according to the known ordering. Adversary C answers the **LR** query with y_j. When A finishes with output b', C outputs 1 if $b = b'$ and 0 otherwise.

Finally, we claim that $\Pr\left[\, G_2^A(k) \Rightarrow \mathsf{true} \,\right] = 1/2$. This is true since the answer to the **LR** query no longer depends on the bit b. Combining the above equations we get

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{PtD},A}(k) &\le \mathbf{Adv}^{\mathrm{los}}_{\mathcal{AE}_\mathrm{d},\mathcal{K}_l,B}(k) + 2\cdot \mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},C}(k) \\
&\le \mathbf{Adv}^{\mathrm{los}}_{\mathcal{AE}_\mathrm{d},\mathcal{K}_l,B}(k) + 2\cdot 2^{n(k)}\cdot \sqrt{2^{n_\mathrm{d}(k)-\ell(k)-(n_\mathrm{d}(k)-n(k))}} \\
&\le \mathbf{Adv}^{\mathrm{los}}_{\mathcal{AE}_\mathrm{d},\mathcal{K}_l,B}(k) + \sqrt{2^{3n(k)-\ell(k)+2}}\ ,
\end{aligned}$$

proving the theorem. ▌

6.4 RtD and PtD Without Universality

We have shown that RtD and PtD meet adaptive CDA security when instantiating the deterministic encryption scheme with a u-LTDF. One can also instantiate with LTDFs that are not necessarily universal. This allows a wider variety of instantiations, including several that are more efficient than the best known u-LTDFs (see [11] for a discussion).

We follow a strategy from [11] to replace u-LTDFs with the composition of a LTDF and a pairwise-independent hash. let $\mathcal{AE}_{\mathrm{d}} = (\mathcal{P}_{\mathrm{d}}, \mathcal{K}_{\mathrm{d}}, \mathcal{E}_{\mathrm{d}}, \mathcal{D}_{\mathrm{d}})$ be any deterministic PKE scheme and $\mathcal{H} = (\mathcal{P}, \mathcal{K}, F, F^{-1})$ be a family of efficiently invertible pairwise-independent permutations with the same input and output length as the message length of $\mathcal{AE}_{\mathrm{d}}$. We can build a new deterministic encryption scheme $\mathcal{AE}_{pw}[\mathcal{AE}_{\mathrm{d}}, \mathcal{H}] = (\mathcal{P}_{pw}, \mathcal{K}_{pw}, \mathcal{E}_{pw}, \mathcal{D}_{pw})$ by composing the two as follows. The parameter generation $\mathcal{P}_{pw}$ runs $par_{\mathrm{d}} \leftarrow\!\!\$\, \mathcal{P}_{\mathrm{d}}(1^k)$ as well as $par \leftarrow\!\!\$\, \mathcal{P}(1^k)$ and outputs parameters $par = (par_{\mathrm{d}}, par)$. To compute $\mathcal{K}_{pw}(par)$, first run $(pk, sk) \leftarrow\!\!\$\, \mathcal{K}_{\mathrm{d}}(par_{\mathrm{d}})$ and then randomly choose a function key K by running $K \leftarrow\!\!\$\, \mathcal{K}(par)$. The key pair output is $((pk, K), (sk, K))$. Encryption is defined by $\mathcal{E}_{pw}((pk, K), M) = \mathcal{E}_{\mathrm{d}}(pk, F(K, M))$ and decryption works in the natural way. Note that if $\mathcal{AE}_{\mathrm{d}}$ is an LTDF, then so too is $\mathcal{AE}_{pw}[\mathcal{AE}_{\mathrm{d}}, \mathcal{H}]$.

The resulting deterministic scheme will not in general be anonymous. However, one can provide a direct analysis of PtD and RtD using $\mathcal{AE}_{pw}[\mathcal{AE}_{\mathrm{d}}, \mathcal{H}]$ as the underlying deterministic scheme by applying the "crooked" leftover hash lemma from [11]. We omit the details.

Acknowledgements

We thank the Asiacrypt 2009 reviewers for detailed and thoughtful comments.

References

[1] M. Abdalla, M. Bellare, and P. Rogaway. The oracle diffie-hellman assumptions and an analysis of dhies. In *CT-RSA 2001*, volume 2020 of *LNCS*. Springer.

[2] P. Abeni, L. Bello, and M. Bertacchini. Exploiting DSA-1571: How to break PFS in SSL with EDH, July 2008. `http://www.lucianobello.com.ar/exploiting_DSA-1571/index.html`.

[3] O. Baudron, D. Pointcheval, and J. Stern. Extended notions of security for multicast public key cryptosystems. In *ICALP 2000*, volume 1853 of *LNCS*. Springer.

[4] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *ASIACRYPT 2001*, volume 2248 of *LNCS*. Springer.

[5] M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In *EUROCRYPT 2000*, volume 1807 of *LNCS*. Springer.

[6] M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In *CRYPTO 2007*, volume 4622 of *LNCS*. Springer.

[7] M. Bellare, M. Fischlin, A. O'Neill, and T. Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In *CRYPTO 2008*, volume 5157 of *LNCS*. Springer.

[8] M. Bellare and P. Rogaway. Code-based game-playing proofs and the security of triple encryption. In *EUROCRYPT 2006*, volume 4004 of *LNCS*. Springer.

[9] M. Bellare and P. Rogaway. Optimal asymmetric encryption – how to encrypt with RSA. In *EUROCRYPT 1994*, volume 950 of *LNCS*. Springer.

[10] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo random bits. In *FOCS 1982*. IEEE.

[11] A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In *CRYPTO 2008*, volume 5157 of *LNCS*. Springer.

[12] D. Boneh. Simplified OAEP for the RSA and Rabin functions. In *CRYPTO 2001*, volume 2139 of *LNCS*. Springer.

[13] C. Bosley and Y. Dodis. Does privacy require true randomness? In *TCC 2007*, volume 4392 of *LNCS*. Springer.

[14] D. R. Brown. A weak randomizer attack on RSA-OAEP with e=3. IACR ePrint Archive, 2005.

[15] K. Chung and S. P. Vadhan. Tight bounds for hashing block sources. In *APPROX-RANDOM*, pages 357–370, 2008.

[16] D. Coppersmith. Find a small root of a univariate modular equation. In *EUROCRYPT 1996*, volume 1070 of *LNCS*. Springer.

[17] Y. Dodis, S. J. Ong, M. Prabhakaran, and A. Sahai. On the (im)possibility of cryptography with imperfect randomness. In *FOCS 2004*. IEEE.

[18] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *SIAM Journal of Computing*, 38(1):97–139, 2008.

[19] Y. Dodis and A. Smith. Entropic security and the encryption of high entropy messages. In *TCC 2005*, volume 3378 of *LNCS*. Springer.

[20] L. Dorrendorf, Z. Gutterman, and B. Pinkas. Cryptanalysis of the windows random number generator. In *CCS 2007*. ACM.

[21] E. Fujisaki and T. Okamoto. How to enhance the security of public-key encryption at minimum cost. In *PKC 1999*, volume 1560 of *LNCS*. Springer.

[22] T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO 1984*, volume 196 of *LNCS*. Springer.

[23] I. Goldberg and D. Wagner. Randomness in the Netscape browser. Dr. Dobb's Journal, January 1996.

[24] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[25] Z. Gutterman and D. Malkhi. Hold your sessions: An attack on Java session-id generation. In *CT-RSA 2005*, volume 3376 of *LNCS*. Springer.

[26] Z. Gutterman, B. Pinkas, and T. Reinman. Analysis of the linux random number generator. In *IEEE Symposium on Security and Privacy*, pages 371–385, 2006.

[27] N. Howgrave-Graham. Finding small roots of univariate modular equations revisited. In M. Darnell, editor, *Proceedings of IMA Cryptography and Coding 1997*, pages 131–42. Springer-Verlag, Dec. 1997.

[28] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *STOC 1989*. ACM.

[29] S. Kamara and J. Katz. How to encrypt with a malicious random number generator. In *FSE 2008*, volume 5086 of *LNCS*. Springer.

[30] C. Lu. Encryption against storage-bounded adversaries from on-line strong extractors. *J. Cryptology*, 17(1):27–42, 2004.

[31] J. L. McInnes and B. Pinkas. On the impossibility of private key cryptography with weakly random keys. In *CRYPTO 1990*, volume 537 of *LNCS*. Springer.

[32] M. Mueller. Debian OpenSSL predictable PRNG bruteforce SSH exploit, May 2008. `http://milw0rm.com/exploits/5622`.

[33] K. Ouafi and S. Vaudenay. Smashing SQUASH-0. In *EUROCRYPT 2009*, volume 5479 of *LNCS*. Springer.

[34] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *STOC 2008*. ACM.

[35] P. Rogaway. Nonce-based symmetric encryption. In *FSE 2004*, volume 3017 of *LNCS*. Springer.

[36] P. Rogaway and T. Shrimpton. Deterministic authenticated-encryption: A provable-security treatment of the key-wrap problem. In *EUROCRYPT 2006*, volume 4004 of *LNCS*. Springer.

[37] A. Rosen and G. Segev. Efficient lossy trapdoor functions based on the composite residuosity assumption. Cryptology ePrint Archive, Report 2008/134, 2008.

[38] V. Shoup. Using hash functions as a hedge against chosen ciphertext attack. In *Advances in Cryptology – EUROCRYPT '00*, volume 1807 of *LNCS*, pages 275–288. Springer, 2000.

[39] B. Waters. Personal Communication to Hovav Shacham, December 2008.

[40] S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage. When private keys are public: Results from the 2008 Debian OpenSSL vulnerability. In *IMC 2009*. ACM. To appear.

[41] D. Zuckerman. Simulating BPP using a general weak random source. In *FOCS 1991*. IEEE.

A Adaptive Variants of the Leftover Hash Lemma

In this section we present a generalization of the leftover hash lemma. Informally, the leftover hash lemma (LHL) [28] states that a universal family of functions $\mathcal{H} = (\mathcal{P}, \mathcal{K}, F)$ is a strong extractor, meaning that $F(K, X)$ is statistically indistinguishable from a uniform point when X is drawn from a high min-entropy source. A well-known argument (c.f., [41, Lemma 6]) extends the LHL to block sources[1].

We generalize this lemma to an adaptive variant[2]. Game $\mathrm{ALH}_{\mathcal{H},k}$ is shown in Figure 13. Recall that as defined in Section 2, for every k, all $par \in [\mathcal{P}(1^k)]$ we let $R(par) = \{\, F(K, x) \,:\, K \in [\mathcal{K}(par)] \text{ and } x \in \{0,1\}^n \,\}$. We denote by $\mathbf{h} \leftarrow_{\$} (R(par))^{|\mathbf{m}|}$ selecting $|\mathbf{m}|$ range points independently at random from $R(par)$. An LH adversary may make multiple **RoR** queries, each being a vector m-source over $\{0,1\}^{n(k)}$. The setting is adaptive because each query can depend on replies to previous ones. The adversary makes a single **RevealPK** query and after that makes no **RoR** queries. In Lemma A.1 we bound the advantage of any adversary in this game, formally defined as

$$\mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},A}(k) = 2 \cdot \Pr\left[\, \mathrm{ALH}^{A}_{\mathcal{H}}(k) \Rightarrow \mathsf{true} \,\right] - 1 \,.$$

We also define games $\mathrm{ALH1}_{\mathcal{H},A}$ and $\mathrm{ALH0}_{\mathcal{H},A}$ to be the same as $\mathrm{ALH}_{\mathcal{H},A}$ with $b = 1$ and $b = 0$, respectively. A standard argument shows that

$$\mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},A}(k) = \Pr\left[\, \mathrm{ALH1}^{A}_{\mathcal{H}}(k) \Rightarrow \mathsf{true} \,\right] - \Pr\left[\, \mathrm{ALH0}^{A}_{\mathcal{H}}(k) \Rightarrow \mathsf{false} \,\right] \,.$$

[1] Chung and Vadhan [15] presented an improved analysis of the leftover hash lemma for block sources. Although their tight analysis can slightly improve the parameters of our resulting schemes, we chose for simplicity to follow the more basic approach of [41].

[2] We note that hashing of block sources in an adaptive setting was also considered by Lu [30] in the context of the bounded storage model.

```
Game $\mathrm{ALH}_{\mathcal{H},k}$

procedure Initialize($1^k$):
    $par \leftarrow\!\$\ \mathcal{P}(1^k)$
    $K \leftarrow\!\$\ \mathcal{K}(par)$
    $b \leftarrow\!\$\ \{0,1\}$
    Ret $par$

procedure RoR($\boldsymbol{\mathcal{M}}$):
    If $\mathsf{pkout} = \mathsf{true}$ then Ret $\perp$
    $\mathbf{m} \leftarrow\!\$\ \boldsymbol{\mathcal{M}}(1^k)$
    If $b = 1$ then $\mathbf{h} \leftarrow F(K, \mathbf{m})$
    Else $\mathbf{h} \leftarrow\!\$\ (R(par))^{|\mathbf{m}|}$
    Ret $\mathbf{h}$

procedure RevealPK:
    $\mathsf{pkout} \leftarrow \mathsf{true}$
    Ret $K$

procedure Finalize($b'$):
    Ret $(b = b')$
```

Figure 13: Game ALH (Adaptive Leftover Hash) associated to a family of functions $\mathcal{H} = (\mathcal{P}, \mathcal{K}, F)$.

Lemma A.1 Let $\mathcal{H} = (\mathcal{P}, \mathcal{K}, F)$ be a universal family of hash functions with associated message length $n(\cdot)$ and 2^t-bounded range. Let A be an LH adversary making q **RoR** queries, each being a (μ, v, n)-m-block-source ℓ-vector m-source. Then for all k

$$\mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},A}(k) \le q(k) \cdot v(k) \cdot \sqrt{2^{t(k)-\mu(k)}} \ . \ \square$$

Proof: Fix a $k \in \mathbb{N}$ and let $q = q(k)$. The proof proceeds by a hybrid argument. From A we build LH adversaries B_i that each only query a single time to the **RoR** oracle, see Figure 14. More specifically, the i^{th} query by A is answered via a **RoR** query. All queries before this are answered with random range points and all queries after are answered by simulating directly F. Note that A does not reveal the key until after it has completed all its **RoR** queries, and so each B_i is free to reveal K as indicated.

We also define games HYB_a for $0 \le a \le q$ as shown in Figure 14. By construction we have that HYB_0^A is equivalent to both $\mathrm{ALH1}_{\mathcal{H}}^A$ and $\mathrm{ALH1}_{\mathcal{H}}^{B_1}$ and that HYB_q^A is equivalent to both $\mathrm{ALH0}_{\mathcal{H}}^A$ and $\mathrm{ALH0}_{\mathcal{H}}^{B_q}$. This means in particular that

$$\begin{aligned} \Pr\left[\,\mathrm{ALH1}_{\mathcal{H}}^A \Rightarrow 1\,\right] &= \Pr\left[\,\mathrm{HYB}_0^A \Rightarrow 1\,\right] = \Pr\left[\,\mathrm{ALH1}_{\mathcal{H}}^{B_1} \Rightarrow 1\,\right] \quad \text{and} \\ \Pr\left[\,\mathrm{ALH0}_{\mathcal{H}}^A \Rightarrow 1\,\right] &= \Pr\left[\,\mathrm{HYB}_q^A \Rightarrow 1\,\right] = \Pr\left[\,\mathrm{ALH0}_{\mathcal{H}}^{B_q} \Rightarrow 1\,\right] . \end{aligned}$$

Moreover, we have that for $1 \le i \le q-1$ it holds that

$$\Pr\left[\,\mathrm{HYB}_i^A \Rightarrow 1\,\right] = \Pr\left[\,\mathrm{ALH0}_{\mathcal{H}}^{B_i} \Rightarrow 1\,\right] = \Pr\left[\,\mathrm{ALH1}_{\mathcal{H}}^{B_{i+1}} \Rightarrow 1\,\right] .$$

We thus have that

$$\begin{aligned} \mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},A}(k) &= \Pr\left[\,\mathrm{HYB}_0^A \Rightarrow 1\,\right] - \Pr\left[\,\mathrm{HYB}_q^A \Rightarrow 1\,\right] \\ &= \sum_{j=0}^{q-1} \left(\Pr\left[\,\mathrm{HYB}_j^A \Rightarrow 1\,\right] - \Pr\left[\,\mathrm{HYB}_{j+1}^A \Rightarrow 1\,\right]\right) \\ &= \sum_{j=1}^{q} \left(\Pr\left[\,\mathrm{ALH1}_{\mathcal{H}}^{B_i} \Rightarrow \mathsf{true}\,\right] - \Pr\left[\,\mathrm{ALH0}_{\mathcal{H}}^{B_i} \Rightarrow \mathsf{false}\,\right]\right) \\ &= \sum_{j=1}^{q} \left(\Pr\left[\,\mathrm{ALH1}_{\mathcal{H}}^{B} \Rightarrow \mathsf{true} \mid i = j\,\right] - \Pr\left[\,\mathrm{ALH0}_{\mathcal{H}}^{B} \Rightarrow \mathsf{false} \mid i = j\,\right]\right) \\ &= q \cdot \mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},B}(k) \ . \end{aligned}$$

where the event "$i = j$" occurs when adversary B's random choice of index i matches the specific value of j. The last equality comes from multiplying by $q \cdot \Pr\left[\,i = j\,\right]$, which is equal to one. By [41, Lemma 6] we have that

$$\mathbf{Adv}^{\mathrm{alh}}_{\mathcal{H},B}(k) \le v(k) \cdot \sqrt{2^{t(k)-\mu(k)}}$$

which, plugging into the equation above, completes the proof. ▌

Game HYB_i

```
**procedure Initialize**($1^k$):
$par \leftarrow\$ \mathcal{P}(1^k)$
$K \leftarrow\$ \mathcal{K}(par)$
$j \leftarrow 0$
$b \leftarrow\$ \{0,1\}$
Ret $par$

**procedure RoR**($\boldsymbol{\mathcal{M}}$):
$j \leftarrow j + 1$
$\mathbf{m} \leftarrow\$ \boldsymbol{\mathcal{M}}(1^k)$
If $j \leq i$ then $\mathbf{h} \leftarrow\$ (\mathsf{R}(par))^{|\mathbf{m}|}$
Else $\mathbf{h} \leftarrow F(K, \mathbf{m})$
Ret $\mathbf{h}$

**procedure RevealPK**:
Ret $K$

**procedure Finalize**($b'$):
Ret $b'$
```

```
$B_i^{\mathbf{RoR},\mathbf{RevealPK}}(par)$:
$j \leftarrow 0$
$b' \leftarrow\$ A^{\mathbf{RoR'},\mathbf{RevealPK'}}(par)$
Ret $b'$

**procedure RoR'**($\boldsymbol{\mathcal{M}}$):
$j \leftarrow j + 1$
If $j < i$ then
    $\mathbf{m} \leftarrow\$ \boldsymbol{\mathcal{M}}(1^k)$
    $\mathbf{h} \leftarrow\$ (\mathsf{R}(par))^{|\mathbf{m}|}$
If $j = i$ then
    $\mathbf{h} \leftarrow\$ \mathbf{RoR}(\boldsymbol{\mathcal{M}})$
    $K \leftarrow \mathbf{RevealPK}()$
If $j > i$ then
    $\mathbf{m} \leftarrow\$ \boldsymbol{\mathcal{M}}(1^k)$
    $\mathbf{h} \leftarrow F(K, \mathbf{m})$
Ret $\mathbf{h}$

**procedure RevealPK'**:
Ret $K$
```

Figure 14: Games used in proof of Lemma A.1.