A preliminary version of this paper appears in the proceedings of Eurocrypt 2015. This is the full version, appearing as IACR ePrint Archive Report 2014/876.

Resisting Randomness Subversion: Fast Deterministic and Hedged Public-key Encryption in the Standard Model

Mihir Bellare[1] Viet Tung Hoang[2]

November 13, 2015

Abstract

This paper provides the first *efficient*, *standard-model*, *fully-secure* schemes for some related and challenging forms of public-key encryption (PKE), namely deterministic and hedged PKE. These forms of PKE defend against subversion of random number generators, an end given new urgency by recent revelations on the nature and extent of such subversion. We resolve the (recognized) technical challenges in reaching these goals via a new paradigm that combines UCEs (universal computational extractors) with LTDFs (lossy trapdoor functions). Crucially, we rely only on a weak form of UCE, namely security for statistically (rather than computationally) unpredictable sources. We then define and achieve unique-ciphertext PKE as a way to defend against implementation subversion via algorithm-substitution attacks.

[1] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. Email: `mihir@eng.ucsd.edu`. URL: `http://cseweb.ucsd.edu/~mihir/`. Supported in part by NSF grants CNS-1116800 and CNS-1228890.

[2] Department of Computer Science, University of Maryland, College Park, and Department of Computer Science, Georgetown University, 37th and O Streets, NW, Washington, DC 20057, USA. Email: `tvhoang@umd.edu`. URL: `http://csiflabs.cs.ucdavis.edu/~tvhoang/`. Supported in part by NSF award CNS-1223623. Part of the work was done while Hoang was working at UCSD and supported in part by NSF grants CNS-1116800, and CNS-1228890.

Contents

1 Introduction

Recent revelations about the prevalence of mass-surveillance and subversion raise new challenges for cryptography. This paper is concerned with subversion of public-key encryption (PKE). We first consider randomness-subversion attacks, namely ones that undermine randomness-generation processes. Forms of PKE resisting these have in fact already been defined, namely deterministic public-key encryption (D-PKE) [4] and hedged public-key encryption (H-PKE) [5]. However, good schemes —we mean efficient ones providing full security in the standard model— are not only lacking but a recognized challenge [52]. With the new impetus and urgency arising from the subversion perspective, we revisit these goals to provide such schemes. We achieve our ends via a new PKE paradigm in which universal computational extractors (UCEs) [8] —of the weaker ilk requiring only statistical rather than computational unpredictability— are combined with lossy trapdoor functions (LTDFs) [47].

We then turn to defending against subversion of encryption implementations via algorithm-substitution attacks [11, 55]. Here we follow [11] to define the new goal of unique ciphertext public-key encryption (U-PKE) and then reach it generically and efficiently from D-PKE.

Deterministic PKE. Technically, conceptually and historically, D-PKE is the core goal in this domain, and we begin there. The encryption algorithm of a D-PKE scheme takes public encryption key ek and message m to deterministically return a ciphertext c. We use the IND formalization of [7] which they show equivalent to the PRIV formalization of [4]. These formalizations capture the best possible privacy, namely semantic security for unpredictable messages that do not depend on the public key.

The core IND requirement asks for privacy when messages are individually unpredictable but may be arbitrarily correlated. We call this *full IND security* for emphasis. Full security is important in practice. For example, I might upload an encrypted file, then make a small edit to the file, re-encrypt and re-upload, so that the messages underlying the successive ciphertexts are very similar. It is thus the desired goal.

The EwH —encrypt with hash— D-PKE scheme of [4] encrypts message m under a (any) randomized IND-CPA scheme RE with the coins set to a hash of m. When the hash function is a random oracle, they showed EwH achieves full IND security. Achieving full IND security in the standard model however seemed out of reach. Many standard-model D-PKE schemes, using sophisticated techniques [16, 7, 18, 32, 10, 48, 29], have been proposed, but the security they achieve is not full. They only achieve security for *block sources*, where each message is assumed unpredictable *even given prior ones*, which is not realistic in practice.

The elusiveness of full security in the standard model was explained by Wichs [52], who showed that it could not be achieved under any single-stage assumption. To achieve full security one thus needs a multi-stage assumption. However most assumptions are single stage and it was not immediately clear what would even be a candidate for a suitable multi-stage assumption.

Such a candidate emerged with the UCE class of assumptions of security for hash functions of BHK1 [8]. The latter showed that the RO in EwH could be securely instantiated with a function family H that is $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$ —UCE-secure for *computationally* unpredictable sources— to yield a standard model, fully IND secure D-PKE scheme. Unfortunately, soon after, Brzuska, Farshim and Mittelbach (BFM) [20] showed that $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$-security is not achievable if indistinguishability obfuscation (iO) [3, 33, 34] is possible. BFM [20] and BHK1 [8] independently proposed to instead use $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$— UCE-security for *statistically* unpredictable sources. BFM [20] give some evidence that their attacks will not extend to $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ and that this assumption is weaker.

This raises several questions. Can one show that EwH is secure under $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$? If not, can one provide a new, different D-PKE scheme that achieves full IND-security under $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$?

Results for D-PKE. Our first result is negative. We show that if iO is possible then the RO in EwH is not universally instantiable. In more detail, given *any* family of functions H —in particular a $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ one— we build a (pathological and H-dependent) randomized PKE scheme RE such that (1) RE is IND-

CPA secure, but (2) An attack shows that the D-PKE scheme $\mathsf{EwH}[\mathsf{H},\mathsf{RE}]$ given by the EwH transform is not IND-secure. The starting point is ideas of BFM [20], but several new ideas are needed, including several applications of a variable-output-length PRF to allocate randomness for the iO and a base PKE scheme in such a way that both (1) and (2) are possible. We note that the same negative result was obtained independently and concurrently by [21]. A general framework to obtain RO un-instantiability results via iO is given in [37] but it applies to single-stage games and thus doesn't yield a result for D-PKE.

Let H be a $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ function family. Then our negative result rules out showing an analogue of BHK1 [8], namely that $\mathsf{EwH}[\mathsf{H},\mathsf{RE}]$ is fully IND secure for *any* IND-CPA RE. But there is a loophole, namely that the negative result does not preclude showing this for a *particular* choice of RE. We exploit this loophole to arrive at the desired goal of a fully IND secure D-PKE scheme under $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$, as follows. We take the ROM BR93 PKE scheme [12], instantiate its trapdoor function with a *lossy* trapdoor function (LTDF) [47, 31], and instantiate its RO with H, to get a standard-model PKE scheme RE. Next, we take the D-PKE scheme $\mathsf{EwH}[\mathsf{H},\mathsf{RE}]$, which has two uses of H, under two independent keys. Our D-PKE scheme $\mathsf{DE1}$ is obtained by implementing these two uses of H with a single key. We prove that $\mathsf{DE1}$ is fully IND secure assuming the LTDF is secure and H is $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$. We remark that using a single H key is important to prove security under $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$, not just an efficiency optimization.

The connection of LTDFs to D-PKE was first made by Boldyreva, Fehr and O'Neill (BFO) [16]. Their LTDF-based D-PKE schemes however only achieve security for block sources, not full IND security. The block source restriction seems quite inherent in their methods, and indeed due to Wichs [52] we do not expect to achieve fully IND secure D-PKE using LTDFs alone. Our approach combines LTDFs with $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ to surmount this obstacle.

$\mathsf{DE1}$ is the first D-PKE scheme that is fully IND secure in the standard model. Beyond that, however, it has the following important practical attributes: it is competitive on short messages, very fast on long messages, and supports variable-length messages directly. These practical attributes are a first for standard-model D-PKE schemes.

LTDFs and $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ are a productive and (in retrospect) natural match. Intuitively, LTDFs allow us to move to a game with information-theoretic guarantees, at which point it becomes possible to exploit UCE under statistical unpredictability. We view $\mathsf{DE1}$ as a relatively simple illustration of the power of the UCE+LTDF method. H-PKE brings new challenges, which we surmount via non-trivial extensions of the basic method. We believe the UCE+LTDF method will have applications beyond this as well.

Hedged PKE. The encryption algorithm of a H-PKE scheme takes public encryption key ek, message m and randomness r to deterministically return a ciphertext c. The H-IND requirement of BBNRSS [5] has two parts: (1) standard IND-CPA security if r is good, meaning uniform and independent across encryptions, and (2) semantic security of m if the pair (m, r) is unpredictable and does not depend on the public key. This second requirement is formalized as indistinguishability under chosen-distribution attack (IND-CDA) [5].

H-IND-secure PKE aims to provide the best possible privacy in the face of untrusted randomness. If the randomness is good, it does as well as standard IND-CPA encryption. But, whereas schemes providing only IND-CPA can fail spectacularly under poor randomness [19, 45, 5], H-IND PKE will not. It will compensate for poor randomness by also exploiting any available entropy in the message, protecting the latter as long as the message and randomness *together* are unpredictable. This is as good as it can get, since if the message-randomness pair is predictable, trial re-encryption on candidate pairs will recover the message underlying a target ciphertext. IND-CDA is an extension of IND that coincides with the latter if the randomness has no entropy at all.

In practice the most desirable form of IND-CDA is, again, full, meaning privacy when message-randomness pairs, although individually unpredictable, may be arbitrarily correlated. By full H-IND, we mean IND-CPA plus full IND-CDA. In the ROM, fully H-IND PKE is achieved by an extension of EwH called REwH that encrypts m under an IND-CPA scheme with the coins set to the hash of $m \| r$ [5]. In the standard model, things are more difficult. Providing a fully IND-CDA PKE scheme is harder than providing a fully IND D-PKE scheme because the unpredictability pertains to (m, r) not just m and also, more importantly, because IND-CDA is formalized in [5] as an *adaptive* requirement. Additionally, while IND-CPA is easy in isolation, it is *not* in combination with IND-CDA. The reason is subtle, namely that IND-CDA breaks when m depends on the public key, but IND-CPA must remain secure in this case. This butting of heads of the IND-CPA and IND-CDA conditions doubles the challenge of achieving fully H-IND PKE compared to fully IND D-PKE.

These technical difficulties are reflected in the landscape of standard-model schemes, where fully H-IND PKE has not been achieved under *any* assumption. BBNRSS [5] build standard-model H-IND PKE schemes by composition of standard-model D-PKE and IND-CPA schemes, and also directly via anonymous LTDFs, but these schemes achieve IND-CDA only for block sources. (The latter now means that message-randomness pairs are assumed to be unpredictable even given prior ones.) It is instructive that full H-IND PKE has not even been achieved under $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$. To elaborate, recall that BHK1 [8] showed that $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$-instantiating the RO in EwH results in a fully IND secure standard-model D-PKE scheme. We can correspondingly $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$-instantiate the RO in REwH. But, even if the resulting scheme can be shown fully IND-CDA, there seems no reason it is IND-CPA. The reason is the difficulty alluded to above. Namely, a UCE hash function may not provide security on messages that are a function of the hashing key, but the latter is part of the public key and IND-CPA requires security for messages depending on the public key.

But the bar for us is even higher: due to the BFM attacks [20] on $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$, we want to use the weaker $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ assumption, just as we did for $\mathsf{DE1}$. We thus face at least two difficulties. The first is to achieve full IND-CDA under $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$. Here the main challenge is handling adaptivity. But beyond that the fundamental above-mentioned difficulty of achieving IND-CPA in the same scheme remains, because no form of UCE guarantees security for messages that depend on the hashing key.

Results for H-PKE. We surmount the technical difficulties discussed above to provide the first standard-model, fully H-IND PKE schemes. We specify three schemes, $\mathsf{HE1}, \mathsf{HE2}$ and $\mathsf{HE3}$. All efficiently achieve our security goals, the second and third handle variable-length messages, and the third further adds better concrete security.

Recall that we obtained $\mathsf{DE1}$ as $\mathsf{EwH}[\mathsf{H}, \mathsf{BR93}[\mathsf{LT}, \mathsf{H}]]$, where H is $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ and LT is a LTDF. A natural idea is to similarly get H-PKE as $\mathsf{REwH}[\mathsf{H}, \mathsf{BR93}[\mathsf{LT}, \mathsf{H}]]$. (In both cases we use one hash key rather than two.) We are able to show this achieves full IND-CDA. This is significant since handling adaptivity required anonymous LTDFs in [5] which we do not need. But we then hit the problem above, namely $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ security of H may not be enough to provide IND-CPA. We resolve this by building a *particular*, suitable $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ family H. We first build a particular family U of AU (almost universal) hash functions and then obtain H by applying the $\mathsf{AU\text{-}then\text{-}Hash}$ transform of BHK2 [9] to a fixed-input-length $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ family $\overline{\mathsf{H}}$ and our U. We refer to the resulting PKE scheme as $\mathsf{HE1}$. We are able to show that it is full IND-CDA as well as IND-CPA assuming $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ security of $\overline{\mathsf{H}}$ and security of the LTDF.

This achieves, for the first time, the security goal of fully H-IND PKE in the standard model, which we consider already significant. But in terms of practicality, $\mathsf{HE1}$ is not ideal because it can only handle fixed-length messages. $\mathsf{HE2}$ efficiently encrypts variable and arbitrary length messages while retaining full H-IND security. It uses a variable-output-length PRF in addition to the primitives used by $\mathsf{HE1}$. Finally, $\mathsf{HE3}$ exploits some combinatorial techniques to obtain better security bounds, as a result of which it offers security for lower values of the message min-entropy than the other schemes.

Speed. Our D-PKE and H-PKE schemes are the first to achieve full security in the standard model, which we believe is a significant theoretical contribution. However, beyond that, they have important practical attributes, expanded on below and in Section 5.

It is well known that asymmetric primitives are orders of magnitude less efficient than symmetric ones. Central to making standard IND-CPA encryption efficient is hybrid encryption as represented by the KEM-DEM paradigm [24]. Encryption generates a random asymmetrically-protected per-message symmetric key and then symmetrically encrypts the message under the latter, leading to cheap encryption of long messages. But for standard model D-PKE and H-PKE the hybrid encryption paradigm breaks down, because, with the constraint of being deterministic or not trusting the randomness, it is not clear how to even pick the per-message key. This difficulty is recognized and seems quite fundamental and hard to bypass. As a result, prior standard-model D-PKE and H-PKE schemes fix the message length and rely only on asymmetric operations. Their cost in asymmetric operations becomes exorbitant on long messages and they also cannot encrypt variable-length messages.

Our methods break these efficiency bottlenecks to recover hybrid-encryption like performance. Our $\mathsf{DE1}, \mathsf{HE2}$ and $\mathsf{HE3}$ schemes handle messages of variable and arbitrary length, and the asymmetric cost is fixed independent of the message length, so that we pay only in hashing as the message length grows. Placing us in a particularly good position to exploit this is the speed of $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ functions. Direct constructions based on HMAC-SHA-256 [8, 44] are already efficient, but in fact still more efficient and even parallelizable constructions are given in BHK2 [9], along with software implementations and cost comparisons. Meanwhile LTDFs can be efficiently instantiated in a variety of ways [47, 31, 42, 39], making the asymmetric component competitive. This leads overall to performance comparable to existing IND-CPA schemes while providing protection against randomness subversion.

In practice concrete security is important to know how to set parameters. Good bounds are important so that one may use smaller parameters. (The cost of the asymmetric operations is usually cubic in the key length so cutting the latter by one-half yields a factor eight speedup.) For this reason we not only state in our theorems the concrete security bounds of the reductions but also try to obtain good ones.

Unique-ciphertext PKE. In an algorithm-substitution attack (ASA) [11, 55], the prescribed encryption algorithm is replaced with a malicious one that may attempt to leak information about the message to "big brother" based on a shared key. BPR [11] formalize the attacker goal in an ASA as compromising privacy without detection. BPR [11] and ACMPS [1] indicate that randomized encryption will be subject to successful attack. In the symmetric setting, BPR [11] show that ASAs can be protected against by a form of deterministic encryption they call unique-ciphertext symmetric encryption.

We analogously define unique-ciphertext PKE. U-PKE requires that for every key pair (ek, dk) and message m, there is at most one ciphertext c that decrypts to m under dk. A U-PKE scheme is thus deterministic, but not every D-PKE scheme is U-PKE. For example, appending to a D-PKE ciphertext a zero bit ignored by decryption leaves D-PKE intact but violates U-PKE. In Section 6 we show however how to achieve U-PKE in a simple and generic way from D-PKE. Combining this with our efficient D-PKE scheme above yields efficient U-PKE, allowing us to better defend against ASAs.

Discussion and related work. In a world of subversion, there are no panaceas. As with BPR [11], our goals are deliberately restricted in scope. We aim to provide better (not perfect) security in the face of some (not all) subversion threats. Thus, we restrict attention to randomness-subversion attacks and algorithm-substitution attacks. We assume that key-generation, being one-time, can leverage good randomness.

We might view IND-CPA as the optimistic view (the randomness is excellent, use it), D-PKE as the pessimistic view (the randomness may be bad so, to be safe, ignore it) and H-IND PKE as the pragmatic view (I don't know how good the randomness is but I will just get the best out of it that I can). We would expect the extent and nature of randomness subversion to vary rather than be ubiquitous and total, in part because subversion will aim to evade detection. In this light H-IND PKE emerges as the best defense in the face of randomness subversion.

Failures of randomness-generation processes [23, 40, 43, 27, 28, 38] have in the past been attributed to error. Now we know better, namely that some should be attributed to subversion. This makes practical defenses more urgent and increases the motivation for work like ours that delivers such defenses.

At SXSW 2014, Snowden said "... we know that the encryption algorithms we are using today work ... it is the random number generators that are attacked as opposed to the encryption algorithms themselves ... ". We aim, in some sense, to turn this on its head. We suggest that the encryption algorithms *don't* work because they are not robust in the face of poor randomness. We pursue practical hedged encryption as a counter-measure.

We do not expect or aim to maintain, under subversion, the high level of security we can achieve in its absence. Security will unavoidably degrade. Our goal with H-IND PKE is for it to degrade as little as possible rather than disappear. This philosophy sets us apart from most of the related work on randomness subversion we will discuss in the next paragraph, which either aims to understand under what limitations on the class of attacks one can achieve the same security one would under perfect randomness, or shows that such security is not possible.

Yilek [54] studies randomness-reset attacks, where the randomness is uniform but the adversary can force its re-use across different encryptions. Paterson, Schuldt and Sibborn [46] introduce related-randomness attacks, where encryption is under adversary-specified functions of some initial uniform randomness, providing negative results, as well as positive results for some classes of attacks. Birrell, Chung, Pass and Telang [14] and Hemenway and Ostrovsky [39] study the encryption of randomness-dependent messages. Austrin, Chung, Mahmoody, Pass and Seth [1] show that encryption is insecure under even quite weak adversarial tampering of randomness. Authenticated key-exchange with bad randomness is studied in [53, 30]. Negative results for cryptography with imperfect randomness are provided by [26, 17, 25]. Kamara and Katz [41] study symmetric encryption providing semantic security under good coins in the face of chosen-plaintext attacks involving bad coins.

Ristenpart and Yilek [49] study the use of H-IND PKE in real systems. Brakerski and Segev [18] study D-PKE security in the presence of auxiliary information about messages. Raghunathan, Segev and Vadhan [48] study security of D-PKE when the message may depend on the public key. Vergnaud and Xiao [51] study IND-CDA when the message and randomness may depend on the public key. In the symmetric setting, Rogaway and Shrimpton's misuse-resistant authenticated encryption [50] represents a form of hedging.

2 Preliminaries

We review basic notation and definitions including games, function families, VOL PRFs, LTDFs and UCE.

Notation. By $\lambda \in \mathbb{N}$ we denote the security parameter. If $n \in \mathbb{N}$ then 1^n denotes its unary representation. We denote the size of a finite set X by $|X|$, the number of coordinates of a vector $\mathbf{x}$ by $|\mathbf{x}|$, and the length of a string $x \in \{0,1\}^*$ by $|x|$. We let ε denote the empty string. If x is a string then $x[i]$ is its i-th bit and $x[1,\ell] = x[1] \ldots x[\ell]$. By $x \| y$ we denote the concatenation of strings x, y. If X is a finite set, we let $x \leftarrow\$ X$ denote picking x uniformly at random from X. Algorithms may be randomized unless otherwise indicated. Running time is worst case. "PT" stands for "polynomial-time," whether for randomized algorithms or deterministic ones. If A is an algorithm, we let $y \leftarrow A(x_1, \ldots; r)$ denote running A with randomness r on inputs $x_1, \ldots$ and assigning the output to y. We let $y \leftarrow\$ A(x_1, \ldots)$ be the resulting of picking r at random and letting $y \leftarrow A(x_1, \ldots; r)$. We let $[A(x_1, \ldots)]$ denote the set of all possible outputs of A when invoked with inputs $x_1, \ldots$. We say that $f : \mathbb{N} \to \mathbb{R}$ is negligible if for every positive polynomial p, there exists $n_p \in \mathbb{N}$ such that $f(n) < 1/p(n)$ for all $n > n_p$. An adversary is an algorithm or a tuple of algorithms.

Games. We use the code based game playing framework of [13]. (See Fig. 1 for an example.) By $\mathrm{G}^A(\lambda) \Rightarrow y$ we denote the event that the execution of game G with adversary A and security parameter λ results in output y, the game output being what is returned by the game. We abbreviate $\mathrm{G}^A(\lambda) \Rightarrow \mathsf{true}$ by $\mathrm{G}^A(\lambda)$, the occurrence of this event meaning that A wins the game.

For concrete security assessments, we adopt the notation of [9]. Let the *number of queries* of A to an oracle Proc be the function $\mathbf{Q}_A^{\textsc{Proc}}$ that on input λ returns the maximum number of queries that A makes to Proc when executed with security parameter λ, the maximum over all coins and all possible replies to queries to all oracles of A. Time assessments are simplified by the convention that running time is

```
GAME $\mathrm{CPA}^{A}_{\mathsf{PKE}}(\lambda)$
$(ek, dk) \leftarrow\$ \mathsf{PKE.Kg}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
$(m_0, m_1, t) \leftarrow\$ A(1^\lambda, ek)$
$c \leftarrow\$ \mathsf{PKE.Enc}(ek, m_b)$
$b' \leftarrow\$ A(1^\lambda, t, c)$
Return $(b = b')$
```

```
GAME $\mathrm{PRF}^{A}_{\mathsf{F}}(\lambda)$
$b \leftarrow\$ \{0,1\}$ ; $fk \leftarrow\$ \{0,1\}^{\mathsf{F.kl}(\lambda)}$
$b' \leftarrow\$ A^{\mathrm{RR}}(1^\lambda)$ ; Return $(b = b')$

$\mathrm{RR}(x, 1^\ell)$
If $b = 1$ then $y \leftarrow \mathsf{F.Ev}(1^\lambda, fk, x, 1^\ell)$
Else $y \leftarrow\$ \{0,1\}^\ell$
Return $y$
```

```
GAME $\mathrm{Lossy}^{A}_{\mathsf{LT}}(\lambda)$
$(ek, dk) \leftarrow\$ \mathsf{LT.EKg}(1^\lambda)$
$lk \leftarrow\$ \mathsf{LT.LKg}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
If $b = 1$ then $K \leftarrow ek$
Else $K \leftarrow lk$
$b' \leftarrow A(1^\lambda, K)$
Return $(b' = b)$
```

Figure 1: **Left:** Game CPA defining IND-CPA security of a PKE scheme PKE. **Middle:** Game PRF defining the PRF security of a variable-output-length function family F. **Right:** Game Lossy defining the security of a lossy trapdoor function LT.

---

that of the game rather than merely the adversary, and we let $\mathbf{T}(\mathrm{G}^{A_1,A_2,\dots})$ denote the function of λ that returns the maximum execution time of game G with adversaries $A_1, A_2, \dots$ and security parameter λ, the maximum over all coins, and the time being all inclusive, meaning the time taken by game procedures to compute replies is included.

Function families. Our syntax for function families follows [8], in particular allowing variable output lengths. This is important in our applications to encrypt messages of variable length, which in turn is important in practice. A family of functions H specifies the following. On input the unary representation 1^λ of the security parameter $\lambda \in \mathbb{N}$, key generation algorithm $\mathsf{H.Kg}$ returns a key $hk \in \{0,1\}^{\mathsf{H.kl}(\lambda)}$, where $\mathsf{H.kl}\colon \mathbb{N} \to \mathbb{N}$ is the key length function associated to H. The deterministic, PT evaluation algorithm $\mathsf{H.Ev}$ takes 1^λ, key hk an input $x \in \{0,1\}^*$ with $|x| \in \mathsf{H.IL}(\lambda)$, and a unary encoding 1^ℓ of an output length $\ell \in \mathsf{H.OL}(\lambda)$ to return $\mathsf{H.Ev}(1^\lambda, hk, x, 1^\ell) \in \{0,1\}^\ell$. Here $\mathsf{H.IL}$ is the input-length function associated to H, so that $\mathsf{H.IL}(\lambda) \subseteq \mathbb{N}$ is the set of allowed input lengths, and similarly $\mathsf{H.OL}$ is the output-length function associated to H, so that $\mathsf{H.OL}(\lambda) \subseteq \mathbb{N}$ is the set of allowed output lengths. The latter allows us to cover functions of variable output length. If H has fixed input length then let $\mathsf{H.il}$ denote the function such that $\mathsf{H.IL}(\lambda) = \{\mathsf{H.il}(\lambda)\}$ for every $\lambda \in \mathbb{N}$. If H has fixed output length, define $\mathsf{H.ol}$ likewise.

Variable output length PRFs. A variable output length (VOL) PRF is a function family F such that $\mathsf{F.Kg}$ returns a uniformly distributed key in $\{0,1\}^{\mathsf{F.kl}}$ and $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},A}(\lambda) = 2\Pr[\mathrm{PRF}^{A}_{\mathsf{F}}(\lambda)] - 1$ is negligible for every PT adversary A, where game $\mathrm{PRF}^{A}_{\mathsf{F}}$ is defined in the middle panel of Fig. 1. In this game the adversary is given an oracle RR that either implements a random oracle or $\mathsf{F.Ev}(1^\lambda, fk, \cdot, \cdot)$, where $fk \leftarrow\$ \{0,1\}^{\mathsf{F.kl}(\lambda)}$ is a random key. We assume that A doesn't repeat a prior RR query, and any RR query $(x, 1^\ell)$ must satisfy $x \in \mathsf{F.IL}(\lambda)$ and $\ell \in \mathsf{F.OL}(\lambda)$. This extends [35] to VOL families. A practical construction of a VOL PRF from a blockcipher is given in [15].

Public-key encryption. A PKE scheme PKE defines PT algorithms $\mathsf{PKE.Kg}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec}$, the last deterministic. Algorithm $\mathsf{PKE.Kg}$ takes as input 1^λ and outputs a public encryption key $ek \in \{0,1\}^{\mathsf{PKE.ekl}(\lambda)}$ and a secret decryption key dk, where $\mathsf{PKE.ekl}\colon \mathbb{N} \to \mathbb{N}$ is the public-key length of PKE. Algorithm $\mathsf{PKE.Enc}$ takes as input $1^\lambda, ek$ and a message m with $|m| \in \mathsf{PKE.IL}(\lambda)$ to return a ciphertext c, where $\mathsf{PKE.IL}$ is the input-length function associated to PKE, so that $\mathsf{PKE.IL}(\lambda) \subseteq \mathbb{N}$ is the set of allowed input (message) lengths. Algorithm $\mathsf{PKE.Dec}$ takes $1^\lambda, dk, c$ and outputs $m \in \{0,1\}^* \cup \{\bot\}$. Correctness requires that $\mathsf{PKE.Dec}(1^\lambda, dk, c) = m$ for all $\lambda \in \mathbb{N}$, all $(ek, dk) \in [\mathsf{PKE.Kg}(1^\lambda)]$ all m with $|m| \in \mathsf{PKE.IL}(\lambda)$ and all $c \in [\mathsf{PKE.Enc}(1^\lambda, ek, m)]$. Scheme PKE is IND-CPA secure [36] if $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{PKE},A}(\lambda) = 2[\mathrm{CPA}^{A}_{\mathsf{PKE}}(\lambda)] - 1$ is negligible for every PT adversary A, where game CPA is defined in the left panel of Fig. 1. We require that the messages m_0, m_1 output by A have the same length $|m_0| = |m_1| \in \mathsf{PKE.IL}(\lambda)$. Let $\mathsf{PKE.rl}\colon \mathbb{N} \to \mathbb{N}$ denote the randomness-length function of PKE, meaning $\mathsf{PKE.Enc}(1^\lambda, \cdot, \cdot)$ draws its coins at random from $\{0,1\}^{\mathsf{PKE.rl}(\lambda)}$. We say that PKE has input length $\mathsf{PKE.il}\colon \mathbb{N} \to \mathbb{N}$ if $\mathsf{PKE.IL}(\lambda) = \{\mathsf{PKE.il}(\lambda)\}$ for all

```
GAME $\mathrm{UCE}_{\mathsf{H}}^{S,D}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$ ; $hk \leftarrow_{\$} \mathsf{H.Kg}(1^\lambda)$
$L \leftarrow_{\$} S^{\mathrm{HASH}}(1^\lambda)$
$b' \leftarrow_{\$} D(1^\lambda, hk, L)$
Return $(b' = b)$

HASH$(x, 1^\ell)$
If $T[x,\ell] = \bot$ then
   If $b = 0$ then $T[x,\ell] \leftarrow_{\$} \{0,1\}^\ell$
   Else $T[x,\ell] \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^\ell)$
Return $T[x,\ell]$
```

```
GAME $\mathrm{Pred}_S^P(\lambda)$
$Q \leftarrow \emptyset$ ; $L \leftarrow_{\$} S^{\mathrm{HASH}}(1^\lambda)$
$Q' \leftarrow_{\$} P(1^\lambda, L)$
Return $(Q' \cap Q \neq \emptyset)$

HASH$(x, 1^\ell)$
If $T[x,\ell] = \bot$ then
   $T[x,\ell] \leftarrow_{\$} \{0,1\}^\ell$
$Q \leftarrow Q \cup \{x\}$
Return $T[x,\ell]$
```

```
GAME $\mathrm{Reset}_S^R(\lambda)$
$\mathsf{Dom} \leftarrow \emptyset$ ; $L \leftarrow_{\$} S^{\mathrm{HASH}}(1^\lambda)$ ; $b \leftarrow_{\$} \{0,1\}$
If $b = 0$ then   // reset the array $T$
   For all $(x,\ell) \in \mathsf{Dom}$ do
      $T[x,\ell] \leftarrow_{\$} \{0,1\}^\ell$
$b' \leftarrow R^{\mathrm{HASH}}(1^\lambda, L)$ ; Return $(b' = b)$

HASH$(x, 1^\ell)$
If $T[x,\ell] = \bot$ then $T[x,\ell] \leftarrow_{\$} \{0,1\}^\ell$
$\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{(x,\ell)\}$ ; Return $T[x,\ell]$
```

Figure 2: **Games** UCE **(left),** Pred **(middle), and** Reset **(right) to define UCE security.**

$\lambda \in \mathbb{N}$, and refer to this as a PKE scheme that only allows fixed length messages. Our goal will be to allow variable and arbitrary-length messages, ideally $\mathsf{PKE.IL}(\cdot) = \mathbb{N}$, but at least some large subset thereof.

Lossy trapdoor functions. A lossy trapdoor function [47] LT specifies PT algorithms $\mathsf{LT.EKg}, \mathsf{LT.LKg}, \mathsf{LT.Ev}, \mathsf{LT.Inv}$, the last two deterministic, as well as an input length $\mathsf{LT.il}\colon \mathbb{N} \to \mathbb{N}$ and an output length $\mathsf{LT.ol}\colon \mathbb{N} \to \mathbb{N}$. Key-generation algorithm $\mathsf{LT.EKg}$ takes 1^λ and returns an "injective" key ek and a decryption key dk. Evaluation algorithm $\mathsf{LT.Ev}$ takes $1^\lambda, ek$ and $x \in \{0,1\}^{\mathsf{LT.il}(\lambda)}$ to return an $\mathsf{LT.ol}(\lambda)$-bit string. Inversion algorithm $\mathsf{LT.Inv}$ takes $1^\lambda, dk$ and $y \in \{0,1\}^{\mathsf{LT.ol}(\lambda)}$ to return a $\mathsf{LT.il}(\lambda)$-bit string. The *correctness requirement* demands that $\mathsf{LT.Inv}(1^\lambda, dk, \mathsf{LT.Ev}(1^\lambda, ek, x)) = x$ for every $\lambda \in \mathbb{N}$, every $(ek, dk) \in [\mathsf{LT.EKg}(1^\lambda)]$ and every $x \in \{0,1\}^{\mathsf{LT.il}(\lambda)}$. Algorithm $\mathsf{LT.LKg}$, given 1^λ, returns a "lossy" key lk. Let $\tau : \mathbb{N} \to \mathbb{N}$ be a function such that $2^{-\tau(\cdot)}$ is negligible. We say that LT is τ-lossy if the size of the set $\{\mathsf{LT.Ev}(1^\lambda, lk, x) \mid x \in \{0,1\}^{\mathsf{LT.il}(\lambda)}\}$ is at most $2^{\mathsf{LT.il}(\lambda) - \tau(\lambda)}$ for every $\lambda \in \mathbb{N}$ and every $lk \in [\mathsf{LT.LKg}(1^\lambda)]$. Security of an LTDF demands two things. First, lossy and injective keys are indistinguishable. Formally, $\mathsf{Adv}_{\mathsf{LT},A}^{\mathsf{ltdf}}(\lambda) = 2\Pr[\mathrm{Lossy}_{\mathsf{LT}}^{A}(\cdot)] - 1$ must be negligible for every PT adversary A, where game Lossy is defined in the right panel of Fig. 1. Second, LTDF is τ-lossy for some τ such that $2^{-\tau(\cdot)}$ is negligible. To simplify concrete security analyses, we assume that $\mathsf{LT.LKg}$'s worst-case running time is at most that of $\mathsf{LT.EKg}$.

There are by now many constructions of LTDFs known [47, 31, 42, 39]. As an example, RSA is shown to be lossy [42] under the Φ-hiding assumption of [22]. For a 2048-bit modulus, one may choose $\tau = 430$ for 80-bit security.

UCE. We recall the Universal Computational Extractor (UCE) framework of BHK1 [8]. Let H be a family of functions as defined above. Let S be an adversary called the *source* and D an adversary called the *distinguisher*. We associate to them and H the game $\mathrm{UCE}_{\mathsf{H}}^{S,D}(\lambda)$ at the left panel of Fig. 2. The source has access to an oracle HASH and we require that any query $(x, 1^\ell)$ made to this oracle satisfy $|x| \in \mathsf{H.IL}(\lambda)$ and $\ell \in \mathsf{H.OL}(\lambda)$. When the challenge bit b is 1 (the "real" case) the oracle responds via $\mathsf{H.Ev}$ under a key hk that is chosen by the game and *not* given to the source. When $b = 0$ (the "random" case) it responds as a random oracle. The source then leaks a string L to its accomplice distinguisher. The latter *does* get the key hk as input and must now return its guess $b' \in \{0,1\}$ for b. The game returns true iff $b' = b$, and the uce-advantage of (S, D) is defined for $\lambda \in \mathbb{N}$ via $\mathsf{Adv}_{\mathsf{H},S,D}^{\mathsf{uce}}(\lambda) = 2\Pr[\mathrm{UCE}_{\mathsf{H}}^{S,D}(\lambda)] - 1$. If $\mathcal{S}$ is a class (set) of sources, we say that H is $\mathsf{UCE}[\mathcal{S}]$-secure if $\mathsf{Adv}_{\mathsf{H},S,D}^{\mathsf{uce}}(\cdot)$ is negligible for all sources $S \in \mathcal{S}$ and all PT distinguishers D. Trivial attacks from [8] show that $\mathsf{UCE}[\mathcal{S}]$-security is not achievable if $\mathcal{S}$ is the class of all PT sources. To obtain meaningful notions of security, BHK1 [8] impose restrictions on the source. There are many ways to do this; below we'll focus on what they call statistically unpredictable and reset-secure sources.

A source is unpredictable if it is hard to guess the source's HASH queries even given the leakage, in the

```
Game $\mathrm{IND}^{A}_{\mathsf{DE}}(\lambda)$
$b \leftarrow\!\!\$\ \{0,1\}$
$(ek, dk) \leftarrow\!\!\$\ \mathsf{DE.Kg}(1^\lambda)$
$(\mathbf{m}_0, \mathbf{m}_1) \leftarrow\!\!\$\ A_1(1^\lambda)$
For $i = 1$ to $|\mathbf{m}_0|$ do
    $\mathbf{c}[i] \leftarrow \mathsf{DE.Enc}(1^\lambda, ek, \mathbf{m}_b[i])$
$b' \leftarrow\!\!\$\ A_2(1^\lambda, ek, \mathbf{c})$
Return $(b = b')$
```

```
$\mathsf{DE.Kg}(1^\lambda)$
$(ek, dk) \leftarrow\!\!\$\ \mathsf{RE.Kg}(1^\lambda)$ ; $hk \leftarrow\!\!\$\ \mathsf{H.Kg}(1^\lambda)$
Return $((ek, hk), dk)$

$\mathsf{DE.Enc}(1^\lambda, (ek, hk), m)$
$r \leftarrow \mathsf{H.Ev}(1^\lambda, hk, ek \,\|\, m, 1^{\mathsf{RE.rl}(\lambda)})$
$c \leftarrow \mathsf{RE.Enc}(1^\lambda, ek, m; r)$ ; Return $c$

$\mathsf{DE.Dec}(1^\lambda, dk, c)$
$m \leftarrow \mathsf{RE.Dec}(1^\lambda, dk, c)$ ; Return $m$
```

```
Game $\mathrm{IO}^{A}_{\mathsf{G}}(\lambda)$
$(C_0, C_1, t) \leftarrow\!\!\$\ A(1^\lambda)$
$b \leftarrow\!\!\$\ \{0,1\}$
$P \leftarrow\!\!\$\ \mathsf{G.Ob}(1^\lambda, C_b)$
$b' \leftarrow\!\!\$\ A(t, P)$
Return $(b = b')$
```

Figure 3: **Left:** Game defining IND security of D-PKE scheme DE. **Middle:** D-PKE scheme $\mathsf{DE} = \mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$. **Right:** Game defining iO security of an indistinguishability obfuscator G.

random case of UCE game. Formally, let S be a source and P an adversary called a predictor. Consider game $\mathrm{Pred}^{P}_{S}(\lambda)$ in the middle panel of Fig. 2. Given the leakage, P outputs a set Q'; we require that $|Q'|$ is polynomially bounded. The predictor wins if this set contains a HASH-query of the source. For $\lambda \in \mathbb{N}$ we let $\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\lambda) = \Pr[\mathrm{Pred}^{P}_{S}(\lambda)]$. We say that S is statistically unpredictable if $\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot)$ is negligible for all (even computationally unbounded) predictors P. We say that H is $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$-secure if $\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot)$ is negligible for all statistically unpredictable PT sources and all PT distinguishers.

The second restriction on sources from [8] is reset security. Let S be a source and R an adversary called a reset adversary. The source again is executed with its HASH being a random oracle. The reset adversary is either given access to the same random oracle or to an *independent* one. The requirement is that it should not be able to tell which. Consider game $\mathrm{Reset}^{R}_{S}(\lambda)$ at the right panel of Fig. 2; we require that R make only polynomial number of queries to HASH. For $\lambda \in \mathbb{N}$ we let $\mathsf{Adv}^{\mathsf{reset}}_{S,R}(\lambda) = 2\Pr[\mathrm{Reset}^{R}_{S}(\lambda)] - 1$. We say S is statistically reset-secure if $\mathsf{Adv}^{\mathsf{reset}}_{S,R}(\cdot)$ is negligible for all reset adversaries R. We say that H is $\mathsf{UCE}[\mathcal{S}^{\mathrm{srs}}]$-secure if $\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot)$ is negligible for all statistically reset-secure PT sources and all PT distinguishers.

BHK1 [8] show that $\mathsf{UCE}[\mathcal{S}^{\mathrm{srs}}]$-security of H implies $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$-security of H. BFM [20] show that if indistinguishability obfuscation for all circuits is possible then $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$ —UCE for *computationally* unpredictable sources— is not achievable in the standard model. However $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ and $\mathsf{UCE}[\mathcal{S}^{\mathrm{srs}}]$ are not subject to their attack and emerge as weaker and plausible assumptions. Moving to the statistical versions was independently suggested by BHK1 [8] and BFM [20]. These statistical assumptions will be the basis of our constructs.

While $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ and $\mathsf{UCE}[\mathcal{S}^{\mathrm{srs}}]$ may seem like strong assumptions, we know that multi-stage assumptions are necessary to reach our goals [52]. There are very few candidate multi-stage assumptions and amongst them the ones we use are the more plausible.

$\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ and $\mathsf{UCE}[\mathcal{S}^{\mathrm{srs}}]$ families may be efficiently instantiated via HMAC-SHA-256 [8, 44] or super-efficiently via [9], which we will exploit for efficient schemes.

3 Efficient, fully IND secure D-PKE

This section begins with a negative result —that assuming iO the random oracle (RO) in EwH is not *universally* instantiable— and then provides a complementary positive result —that there is a *particular* instantiation of the RO and IND-CPA scheme in EwH that results in a fully IND secure D-PKE scheme. The latter, which is the main result of this section, showcases our UCE+LTDF method and brings a new D-PKE scheme with two attributes: (1) On the theoretical front, it is the first D-PKE scheme shown *fully* IND secure in the standard model, and (2) On the practical front, it encrypts variable-input length messages and achieves hybrid-encryption like efficiency on long messages.

D-PKE and EwH. We say that a PKE scheme DE is a deterministic public-key encryption (D-PKE) [4] if the encryption algorithm $\mathsf{DE.Enc}$ is deterministic. We use the IND formalization of security of BFOR [7], which they show equivalent to the PRIV formalization of [4]. Game IND defining the IND notion is shown in the left panel of Fig. 3. An IND adversary $A = (A_1, A_2)$ is a pair of PT algorithms, where A_1 on input 1^λ returns a pair of message vectors $(\mathbf{m}_0, \mathbf{m}_1)$. We require that (i) there be a polynomial v such that $|\mathbf{m}_0| = |\mathbf{m}_1| \leq v(\lambda)$ and $|\mathbf{m}_0[i]| = |\mathbf{m}_1[i]| \in \mathsf{DE.IL}(\lambda)$, for every $i \leq |\mathbf{m}_0|$, and (ii) messages $\mathbf{m}_0[1], \ldots, \mathbf{m}_0[|\mathbf{m}_0|]$ are distinct and also messages $\mathbf{m}_1[1], \ldots, \mathbf{m}_1[|\mathbf{m}_1|]$ are distinct. The guessing probability $\mathrm{Guess}_A(\cdot)$ of A is the function that on input $\lambda \in \mathbb{N}$ returns the maximum, over all b, m, i, of $\Pr[\mathbf{m}_b[i] = m]$, the probability over $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow\!\$\, A_1(1^\lambda)$. We say that A has *high min-entropy* if $\mathrm{Guess}_A(\cdot)$ is negligible. We let $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DE},A}(\lambda) = 2\Pr[\mathrm{IND}^A_{\mathsf{DE}}(\lambda)] - 1$ and say that DE is IND-secure if $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DE},A}(\cdot)$ is negligible for all PT A of high min-entropy.

We stress that this definition captures *full* security because the messages in the message vectors may be arbitrarily correlated. This is what is needed in practice. In contrast, security for block sources [16] requires that each message in each vector has high min entropy even given prior ones. This is often not true in practice and security only for block sources is quite weak, yet prior standard-model schemes have only been able to achieve this.

EwH [4] is a simple and natural transform that takes a family of functions H and a randomized PKE scheme RE to return the D-PKE scheme $\mathsf{DE} = \mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$ whose algorithms are shown in the middle panel of Fig. 3. We let $\mathsf{DE.IL} = \mathsf{RE.IL}$. We require that $\mathsf{RE.rl}(\lambda) \in \mathsf{H.OL}(\lambda)$ and $\mathsf{RE.ekl}(\lambda) + \ell \in \mathsf{H.IL}(\lambda)$ for all $\lambda \in \mathbb{N}$ and all $\ell \in \mathsf{RE.IL}(\lambda)$.

Indistinguishability obfuscation. We recall the definition of [33], which extends that of [3] to allow auxiliary information. We say that circuits C_0 and C_1 are *functionally equivalent*, denoted $C_0 \equiv C_1$, if they have the same size, the same number n of inputs, and $C_0(x) = C_1(x)$ for every input $x \in \{0,1\}^n$. An indistinguishability obfuscator (iO) G defines PT algorithms $\mathsf{G.Ob}, \mathsf{G.Ev}$ and a randomness length function $\mathsf{G.rl}\colon \mathbb{N} \to \mathbb{N}$. Algorithm $\mathsf{G.Ob}$ takes as input 1^λ and a circuit C, and outputs a string P using randomness of length $\mathsf{G.rl}(\lambda)$. Deterministic algorithm $\mathsf{G.Ev}$ takes as input strings P, x and returns $y \in \{0,1\}^* \cup \{\bot\}$. We require that for any circuit C, any input x for C any $\lambda \in \mathbb{N}$, and any $P \in [\mathsf{G.Ob}(1^\lambda, C)]$, it holds that $\mathsf{G.Ev}(P, x) = C(x)$. An adversary A is *well-formed* if $\Pr[C_0 \not\equiv C_1 \colon (C_0, C_1, t) \leftarrow\!\$\, A(1^\lambda)]$ is negligible. We say that G is iO-secure if $\mathsf{Adv}^{\mathsf{io}}_{\mathsf{G},A}(\lambda) = 2\Pr[\mathrm{IO}^A_{\mathsf{G}}(\lambda)] - 1$ is negligible for every PT well-formed adversary A, where game IO is defined at the right panel of Fig. 3.

Implausibility of universal instantiation of EwH. BBO [4] showed that if H is implemented via a RO then $\mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$ is IND-secure for *any* IND-CPA RE. A basic theoretical and practical question is whether the RO in this result can be securely instantiated. The most desirable instantiation is *universal*, by which we mean there is a function family H such that $\mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$ is IND-secure for any IND-CPA RE. Here we show that if iO exists then there is no such universal instantiation. Given any function family H we build an IND-CPA PKE scheme RE such that $\mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$ is not IND-secure. We stress that this does not preclude providing specific H, RE such that $\mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$ is IND-secure, and indeed it is in this way that we will later obtain our positive result.

Our findings strengthen, and are consistent with, prior work. BHK1 [8] showed that a $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$ family will provide a universal instantiation of EwH, but $\mathsf{UCE}[\mathcal{S}^{\mathrm{cup}}]$ is ruled out under iO by BFM [20], so there is no contradiction. However, following BFM, it remained possible that some other class of function families might be able to universally instantiate EwH. Under iO, we rule this out.

We let H be a function family with input length $\mathsf{H.il}$ and output length $\mathsf{H.ol}$. We will build the counter-example PKE scheme RE from H and the following auxiliary primitives: an arbitrary, base IND-CPA scheme $\overline{\mathsf{RE}}$, a VOL PRF F and an iO scheme G. The result is as follows.

Proposition 3.1 Let H be a function family with input length $\mathsf{H.il}$ and output length $\mathsf{H.ol}$. Let F be a VOL PRF with $\mathsf{F.IL} = \mathsf{F.OL} = \mathbb{N}$. Assume $\mathsf{F.kl} \leq \mathsf{H.ol}$. Let $\overline{\mathsf{RE}}$ be an IND-CPA PKE scheme with fixed input length $\overline{\mathsf{RE}}.\mathsf{il}$ and public key length $\overline{\mathsf{RE}}.\mathsf{pkl}$ satisfying $\overline{\mathsf{RE}}.\mathsf{il} + \overline{\mathsf{RE}}.\mathsf{pkl} = \mathsf{H.il}$. Let G be an iO-secure iO

```
Circuit $C_{1^\lambda,x,y}(hk)$
// Input length is $\mathsf{H.kl}(\lambda)$
// Output length is $|x|$
$r \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{H.ol}(\lambda)})$
$fk \leftarrow r[1, \mathsf{F.kl}(\lambda)]$
$u \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
If $y = u$ then return $x$
Return $0^{|x|}$
```

```
$\mathsf{RE.Enc}(1^\lambda, ek, m; r)$
$fk \leftarrow r[1, \mathsf{F.kl}(\lambda)]$
$y \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
$r_1 \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0 \,\|\, 1^{\mathsf{F.il}(\lambda)-1}, 1^{\mathsf{G.rl}(\lambda)})$
$r_2 \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 1^{\mathsf{F.il}(\lambda)}, 1^{\overline{\mathsf{RE}}.\mathsf{rl}(\lambda)})$
$x \leftarrow ek \,\|\, m$ ; $P \leftarrow \mathsf{G.Ob}(1^\lambda, C_{1^\lambda,x,y}; r_1)$
$c' \leftarrow \overline{\mathsf{RE}}.\mathsf{Enc}(1^\lambda, ek, m; r_2)$
$c \leftarrow (c', P)$ ; Return $c$
```

```
$\mathsf{RE.Dec}(1^\lambda, dk, c)$
$(c', P) \leftarrow c$
Return $\overline{\mathsf{RE}}.\mathsf{Dec}(1^\lambda, dk, c')$
```

Figure 4: **Middle, Right:** Encryption and decryption algorithm of the counter-example PKE scheme RE for Proposition 3.1. **Left:** Circuit constructed and obfuscated in $\mathsf{RE.Enc}$.

scheme. Define PKE scheme RE as follows. Let $\mathsf{RE.il} = \overline{\mathsf{RE}}.\mathsf{il}$. Let $\mathsf{RE.Kg} = \overline{\mathsf{RE}}.\mathsf{Kg}$. Let the encryption and decryption algorithms of RE be as shown in Fig. 4. Then (1) $\mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$ is not IND-secure, but (2) RE is IND-CPA secure. ▌

The proof of Proposition 3.1 is in Appendix A. Here we will sketch the ideas. An encryption $c = (c', P)$ of a message m under RE with public key ek will have two parts. The first, c', is an encryption of m under $\overline{\mathsf{RE}}$ with ek. The second, P, is an obfuscated circuit that will (1) help attack $\mathsf{DE} = \mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$ yet (2) not compromise IND-CPA security of RE. The question is how to construct RE to ensure *both* properties. (Ensuring either alone is trivial.)

The starting idea, inspired by BFM [20], is to have $\mathsf{RE.Enc}$, given $1^\lambda, ek, m$ and coins r, create the following circuit:

$$\underline{C_{1^\lambda,ek,m,r}(hk)} \;:\; \text{If } \mathsf{H}(1^\lambda, hk, ek\|m, 1^{\mathsf{RE.rl}(\lambda)}) = r \text{ then return } m \text{ else return } 0^{|m|}.$$

The input to the circuit is a key hk for H, and the hardwired values $1^\lambda, ek, m, r$ are the inputs to the algorithm $\mathsf{RE.Enc}$ that creates the circuit. Now $\mathsf{RE.Enc}$ lets P be an obfuscation of this circuit. Pretend for now that the obfuscation process is deterministic, which of course is not true, and also that no coins are used to create c', which is also not true. Under these assumptions, if an attacker has an EwH ciphertext $(c', P) = \mathsf{DE.Enc}(1^\lambda, (ek, hk), m)$, and also has the public key (ek, hk) of DE, then it can run P on hk which, due to the structure of EwH and the construction of $C_{1^\lambda,ek,m,r}$, returns m, breaking the IND-security of DE. But there are a number of difficulties. One is that there seems no reason that this RE retains IND-CPA security assuming only iO security of the obfuscation. Another is that the obfuscation and $\overline{\mathsf{RE}}$ are randomized, and RE has to provide coins for both from r yet be able to create P to allow the attack when r is produced via the hash in EwH.

We will use the VOL PRF F to allocate pseudorandom coins for the obfuscation process and $\overline{\mathsf{RE}}$. The key for F will be a prefix $fk \leftarrow r[1, \mathsf{F.kl}(\lambda)]$ of the coins r provided to $\mathsf{RE.Enc}$. Recall that in our definition of a VOL PRF, the key generation always samples $fk \leftarrow_\$ \{0,1\}^{\mathsf{F.kl}(\lambda)}$, so if r is truly random then we give F a correctly generated key. Instead of hardwiring r to the circuit, we hardwire $y \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^\ell)$ for an appropriate ℓ. We also hardwire $x = ek\|m$ rather than ek, m separately. Our circuit $C_{1^\lambda,x,y}$ is shown in the left panel of Fig. 4. We need (1) an attack on $\mathsf{DE} = \mathsf{EwH}[\mathsf{H}, \mathsf{RE}]$ and (2) a proof that RE is IND-CPA. For (1) our claim is that if $C_{1^\lambda,ek\|m,y}$ is produced by $\mathsf{RE.Enc}$ within DE then $C_{1^\lambda,ek\|m,y}(hk)$ will return $ek\|m$, and thus running an obfuscation P of $C_{1^\lambda,ek\|m,y}$ on hk will return the same. For (2), r is truly random so $C_{1^\lambda,ek\|m,y}$ as produced during encryption is indistinguishable from $C_{1^\lambda,ek\|m,u}$ with u a random ℓ-bit string, by PRF security of F. To use iO security, we want that when u is random the probability that there exists a $\mathsf{H.kl}(\lambda)$-bit z such that $C_{1^\lambda,ek\|m,u}(z) \neq 0^{|ek\,\|\,m|}$ is negligible. This is established via a counting argument which relies on having set ℓ to be large enough. See Appendix A for details.

<u>The DE1 scheme.</u> We now provide our positive result on D-PKE, namely an efficient, fully IND stan-

$\mathsf{DE1.Kg}(1^\lambda)$	$\mathsf{DE1.Enc}(1^\lambda,(ek,hk),m)$	$\mathsf{DE1.Dec}(1^\lambda,(dk,hk),(trap,c))$
$(ek,dk) \leftarrow\!\!\$\ \mathsf{LT.EKg}(1^\lambda)$ $hk \leftarrow\!\!\$\ \mathsf{H.Kg}(1^\lambda)$ Return $((ek,hk),(dk,hk))$	$r \leftarrow \mathsf{H.Ev}(1^\lambda,hk,m,1^{\mathsf{LT.il}(\lambda)})$ $trap \leftarrow \mathsf{LT.Ev}(1^\lambda,ek,r)$ $c \leftarrow m \oplus \mathsf{H.Ev}(1^\lambda,hk,r,1^{\lvert m\rvert})$ Return $(trap,c)$	$r \leftarrow \mathsf{LT.Inv}(1^\lambda,dk,trap)$ Return $c \oplus \mathsf{H.Ev}(1^\lambda,hk,r,1^{\lvert c\rvert})$

Figure 5: The algorithms of our $\mathsf{DE1}$ D-PKE scheme.

dard model scheme under $\mathsf{UCE}[\mathcal{S}^{\sup}]$. Let H be a $\mathsf{UCE}[\mathcal{S}^{\sup}]$ function family with $\mathsf{H.IL}(\cdot) = \mathsf{H.OL}(\cdot) = \mathbb{N}$. From the above we know that $\mathsf{EwH}[\mathsf{H},\mathsf{RE}]$ will not be IND for all IND-CPA RE. We consider instead a particular choice of IND-CPA RE. Recall that BR93 [12] present a simple TDF-based PKE scheme proven IND-CPA in the ROM. We instantiate their TDF with a LTDF and then instantiate the RO with H to get a standard-model PKE scheme we denote $\mathsf{RE} = \mathsf{BR93}[\mathsf{LT},\mathsf{H}]$. We now consider the standard-model D-PKE scheme $\mathsf{EwH}[\mathsf{H},\mathsf{RE}]$. In this scheme, H is used twice, with two independent keys. Our final $\mathsf{DE1}$ D-PKE scheme is obtained by using the same key for both invocations of H. The algorithms of this scheme are shown in Fig. 5. Importantly, $\mathsf{DE1.IL}(\cdot) = \mathsf{H.OL}(\cdot) = \mathbb{N}$, meaning we can encrypt messages of arbitrary and varying length. We note that using a single H key is not only an optimization in key size but also avoids using multi-key variants of UCE [8] and is important to prove security under $\mathsf{UCE}[\mathcal{S}^{\sup}]$. The following says that $\mathsf{DE1}$ is IND-secure.

Theorem 3.2 Let LT be a lossy trapdoor function and H a $\mathsf{UCE}[\mathcal{S}^{\sup}]$ function family with $\mathsf{H.IL}(\cdot) = \mathsf{H.OL}(\cdot) = \mathbb{N}$. Let $\mathsf{DE1}$ be constructed as in Fig. 5. Then

Asymptotic result: $\mathsf{DE1}$ is IND-secure.

Concrete result: Let A be an adversary and P a predictor. We can construct an adversary B, a source S, and a distinguisher D such that

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DE1},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) + \frac{3v^2}{2^{\mathsf{LT.il}}} \tag{1}$$

$$\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot) \le \frac{1.5v^2}{2^{\mathsf{LT.il}}} + qv \cdot \mathrm{Guess}_A(\cdot) + \frac{qv}{2^\tau} \tag{2}$$

where q is the maximum of the size of P's output in the execution of Pred^P_S, v is the maximum of the size of A's message vector in the execution of $\mathrm{IND}^A_{\mathsf{DE}}$, and τ is the lossiness of LT. Furthermore, $\mathbf{T}(\mathrm{UCE}^{S,D}_{\mathsf{H}}) \le \mathbf{T}(\mathrm{IND}^A_{\mathsf{DE1}})$; $\mathbf{Q}^{\mathrm{HASH}}_S \le v$; and $\mathbf{T}(\mathrm{Lossy}^B_{\mathsf{LT}}) \le \mathbf{T}(\mathrm{IND}^A_{\mathsf{DE1}})$. ▌

The proof is in Appendix B. Here we discuss some of the ideas. To construct a source S and a distinguisher D, a naive method is to let them run A to simulate game $\mathrm{IND}^A_{\mathsf{DE1}}$. However this won't produce a statistically unpredictable source. The key idea is to let our source generate a *lossy* key lk. instead of an injective key ek as in game $\mathrm{IND}^A_{\mathsf{DE1}}$. The statistical unpredictability of S then follows from the lossiness of LT, as represented by Equation (2). On the other hand, game $\mathrm{UCE}^{S,D}_{\mathsf{H}}$ for challenge bit $b = 1$ no longer coincides with game $\mathrm{IND}^A_{\mathsf{DE1}}$. Still, this gap can be bounded by constructing B attacking LT, so that Equation (1) holds.

In Section 5 we discuss how, under appropriate instantiations of the $\mathsf{UCE}[\mathcal{S}^{\sup}]$ family, $\mathsf{DE1}$ is extremely efficient compared to prior standard-model D-PKE schemes.

BFOR [7] originally defined an IND adversary as a triple (A_0, A_1, A_2), where A_0 specifies state information that is passed on to A_1, A_2. Results from [6] indicate this is important to ensure that security in the standard model implies security in the ROM. For notational simplicity, here we omit A_0. Our construction and proof work for the original IND definition with the following modification. One first needs to redefine Guess_A as the conditional min-entropy of the messages, given the state, and then include the state as a part of the leakage of S.

```
Game $\mathrm{CDA}^{A}_{\mathsf{HE}}(\lambda)$
$(ek, dk) \leftarrow\$ \mathsf{HE.Kg}(1^\lambda)$ ; $b \leftarrow\$ \{0,1\}$ ; $t \leftarrow\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow\$ A_2(t, ek)$ ; Return $(b = b')$

LR($d$)
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do $\mathbf{c}[i] \leftarrow \mathsf{HE.Enc}(1^\lambda, ek, \mathbf{m}_b[i]; \mathbf{r}[i])$
Return $\mathbf{c}$
```

Figure 6: Game defining IND-CDA security of PKE scheme HE.

4 Fully secure Hedged PKE

In this section we provide the first fully H-IND PKE schemes in the standard model. Additionally our schemes are efficient. HE1 is our base scheme encrypting fixed-length messages; HE2 encrypts variable-length messages; HE3 has a tighter security analysis. Our schemes provide pragmatic and effective defense against subversion of encryption randomness.

Hedged PKE. To achieve standard IND-CPA security, PKE schemes demand truly random coins. Many well-known PKE schemes fail spectacularly, allowing message recovery from the ciphertext, if the latter is created with even somewhat weak coins [19, 45, 5]. BBNRSS [5] introduce security under chosen-distribution attack (IND-CDA) to provide meaningful security when bad randomness is used. A secure hedged PKE scheme must provide IND-CPA security when the coins are truly random, and fall back to IND-CDA security when bad coins are provided. Formally, for a PKE scheme HE, we say that HE is H-IND secure if (1) HE is IND-CPA secure, and (2) HE is IND-CDA secure. Game CDA defining the IND-CDA notion is given in Fig. 6. An IND-CDA adversary $A = (A_1, A_2)$ is a pair of algorithms. In the first part of the attack, A_2 can adaptively query oracle LR, each query taking a distribution-specifier string d and returning a challenge ciphertext vector $\mathbf{c}$. In this phase A_2 does not get ek. Once this stage ends, it gets ek and must then render its decision. Algorithm A_1 defines a distribution over triples $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r})$ that is a function of d. We require that (i) there be a polynomial v such that $|\mathbf{m}_0| = |\mathbf{m}_1| = |\mathbf{r}| \le v(\lambda)$, (ii) $|\mathbf{m}_0[i]| = |\mathbf{m}_1[i]| \in \mathsf{HE.IL}(\lambda)$ and $|\mathbf{r}[i]| = \mathsf{HE.rl}(\lambda)$ for every $i \le |\mathbf{r}|$, and (iii) for each $b \in \{0,1\}$ the $|\mathbf{r}|$ pairs $(\mathbf{m}_b[i], \mathbf{r}[i])$ are distinct, where $1 \le i \le |\mathbf{r}|$. Let $\mathrm{Guess}_A(\cdot)$ be the function that on input $\lambda \in \mathbb{N}$ returns the maximum, over all b, i, m, r, d, of $\Pr[(\mathbf{m}_b[i], \mathbf{r}[i]) = (m, r)]$, the probability over $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ A_1(1^\lambda, d)$. We say that A has high min-entropy if $\mathrm{Guess}_A(\cdot)$ is negligible. We say that HE is IND-CDA-secure if $\mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE},A}(\cdot) = 2\Pr[\mathrm{CDA}^{A}_{\mathsf{HE}}(\cdot)] - 1$ is negligible for every PT adversary A of high min-entropy. We stress that this captures full IND-CDA since the messages in the message vectors may be arbitrarily correlated.

The HE1 scheme. Recall we obtained our D-PKE scheme DE1 via a BR93-based instantiation of EwH. In analogy it is natural to try to obtain an H-IND scheme via a similar BR93-based instantiation of the REwH transform of BBNRSS [5]. This results in the candidate scheme $\mathsf{Hedge}[\mathsf{H}, \mathsf{LT}]$, associated to a function family H and LTDF LT, whose algorithms are shown in the left panel of Fig. 7. Here $\mathsf{Hedge}[\mathsf{H}, \mathsf{LT}].\mathsf{IL}(\cdot) = \mathsf{H.OL}(\cdot)$, meaning we can encrypt messages of length matching the allowed output lengths of H.

We first ask if one can show IND-CDA security of $\mathsf{Hedge}[\mathsf{H}, \mathsf{LT}]$ assuming $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ security of H. This involves two new difficulties relative to Theorem 3.2. The first, more minor, is the presence of the randomness. The second is more major, namely that the IND-CDA notion is adaptive. To address this, BBNRSS [5] needed quite involved techniques including anonymous LTDFs and an adaptive LHL, and yet only achieved security for block sources, not the full IND-CDA security that we target. However we are able to show that $\mathsf{Hedge}[\mathsf{H}, \mathsf{LT}]$ does achieve (full) IND-CDA assuming only that LT is a (standard) LTDF and H is $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$.

```
$\mathsf{Hedge}[\mathsf{H},\mathsf{LT}].\mathsf{Kg}(1^\lambda)$
$hk \leftarrow_\$ \mathsf{H}.\mathsf{Kg}(1^\lambda)$
$(ek, dk) \leftarrow_\$ \mathsf{LT}.\mathsf{EKg}(1^\lambda)$
Return $((ek, hk), (dk, hk))$

$\mathsf{Hedge}[\mathsf{H},\mathsf{LT}].\mathsf{Enc}(1^\lambda, (ek, hk), m; r)$
$x \leftarrow \mathsf{H}.\mathsf{Ev}(1^\lambda, hk, r \,\|\, m, 1^{\mathsf{LT.il}(\lambda)})$
$trap \leftarrow \mathsf{LT}.\mathsf{Ev}(1^\lambda, ek, x)$
$c \leftarrow \mathsf{H}.\mathsf{Ev}(1^\lambda, hk, x, 1^{|m|}) \oplus m$
Return $(trap, c)$

$\mathsf{Hedge}[\mathsf{H},\mathsf{LT}].\mathsf{Dec}(1^\lambda, (dk, hk), (trap, c))$
$x \leftarrow \mathsf{LT}.\mathsf{Inv}(1^\lambda, dk, trap)$
$m \leftarrow \mathsf{H}.\mathsf{Ev}(1^\lambda, hk, x, 1^{|c|}) \oplus c$
Return $m$
```

```
$\mathsf{H}.\mathsf{Kg}(1^\lambda)$
$uk \leftarrow_\$ \mathsf{U}.\mathsf{Kg}(1^\lambda)$
$\overline{hk} \leftarrow_\$ \overline{\mathsf{H}}.\mathsf{Kg}(\lambda)$
$hk \leftarrow (\overline{hk}, uk)$
Return $hk$

$\mathsf{H}.\mathsf{Ev}(1^\lambda, hk, x, 1^\ell)$
$(\overline{hk}, uk) \leftarrow hk$
$u \leftarrow \mathsf{U}.\mathsf{Ev}(1^\lambda, uk, x)$
$y \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, u, 1^\ell)$
Return $y$
```

```
$\mathsf{U}.\mathsf{Kg}(1^\lambda)$
$\overline{uk} \leftarrow_\$ \overline{\mathsf{U}}.\mathsf{Kg}(1^\lambda)$
$mk \leftarrow_\$ \{0,1\}^{\mathsf{U.ol}(\lambda)}$
$rk \leftarrow_\$ \mathrm{GF}(2^{\mathsf{U.ol}(\lambda)}) \setminus \{0^{\mathsf{U.ol}(\lambda)}\}$
Return $(\overline{uk}, rk, mk)$

$\mathsf{U}.\mathsf{Ev}(1^\lambda, (\overline{uk}, rk), x)$
If $|x| < \mathsf{U.ol}(\lambda)$ then
    Return $mk \oplus (x \,\|\, 10^{\mathsf{U.ol}(\lambda)-|x|})$
$x_1 \leftarrow x[1, \mathsf{U.ol}(\lambda)]$
$x_2 \leftarrow x[\mathsf{U.ol}(\lambda)+1, |x|]$
$y \leftarrow \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, x_2) \oplus (x_1 \times rk)$
Return $y$
```

Figure 7: **Left:** The PKE scheme $\mathsf{Hedge}[\mathsf{H},\mathsf{LT}]$ associated to function family H and LTDF LT. **Middle:** The $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$ VIL $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ family built from an AU hash U and a FIL $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ family $\overline{\mathsf{H}}$. **Right:** The $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$ AU family built from an AU family $\overline{\mathsf{U}}$. The operator $\times$ is multiplication in the finite field $\mathrm{GF}(2^{\mathsf{U.ol}(\lambda)})$ and the string $0^{\mathsf{U.ol}(\lambda)}$ encodes the zero element of $\mathrm{GF}(2^{\mathsf{U.ol}(\lambda)})$. **HE1:** Our $\mathsf{HE1}$ PKE scheme is obtained from an LTDF LT, a FIL $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ family $\overline{\mathsf{H}}$ and an AU family $\overline{\mathsf{U}}$ as $\mathsf{HE1} = \mathsf{Hedge}[\mathsf{H},\mathsf{LT}]$ with $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$ and $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$.

But recall that H-IND requires also that $\mathsf{Hedge}[\mathsf{H},\mathsf{LT}]$ is IND-CPA. But it is quite unclear why this would be true under $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ security of H. The reason is that UCE guarantees nothing for inputs depending on *hk* but messages in IND-CPA can depend on the public key, which contains *hk*. This difficulty is quite fundamental and at first seemed impossible to circumvent within the UCE framework. We resolve it by using a *particular* $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ family H. Let $\overline{\mathsf{H}}$ be a fixed input length $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ family. Recall that the $\mathsf{AU\text{-}then\text{-}Hash}$ transform of BHK2 [9] takes an AU (almost universal) family U and $\overline{\mathsf{H}}$ to return a variable input length family $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$ that they show is itself $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$. We will take an (arbitrary) AU family $\overline{\mathsf{U}}$ and construct another, special AU family $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$ via a transform called $\mathsf{Hash\text{-}then\text{-}Mask}$ that we introduce. Then our $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$ family is $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$. With this choice we will be able to show that $\mathsf{HE1} = \mathsf{Hedge}[\mathsf{H},\mathsf{LT}]$ —this is our scheme— is IND-CPA. In conjunction with our prior claim, $\mathsf{HE1}$ is then H-IND as desired.

We now detail this. We recall some definitions from BHK2 [9]. Let V be a fixed output length (FOL) function family. Let $\lambda, m \in \mathbb{N}$. Let

$$\mathbf{Coll1}_{\mathsf{V}}(\lambda, m) = \max\{\, \Pr[y = \mathsf{V}.\mathsf{Ev}(1^\lambda, vk, x)] \;:\; |y| = \mathsf{V.ol}(\lambda) \text{ and } |x| \le m \,\}$$

$$\mathbf{Coll2}_{\mathsf{V}}(\lambda, m_0, m_1) = \max\{\, \Pr[\mathsf{V}.\mathsf{Ev}(1^\lambda, vk, x_0) = \mathsf{V}.\mathsf{Ev}(1^\lambda, vk, x_1)] \;:\; |x_0| \le m_0,\ |x_1| \le m_1 \text{ and } x_0 \ne x_1 \,\}$$

$$\mathbf{Coll}_{\mathsf{V}}(\lambda, m_0, m_1) = \max\{\, \mathbf{Coll2}_{\mathsf{V}}(\lambda, m_0, m_1),\ \mathbf{Coll1}_{\mathsf{V}}(\lambda, \min\{m_0, m_1\}) \,\}\ .$$

In the first and second equations, the probability is over $vk \leftarrow_\$ \mathsf{V}.\mathsf{Kg}(1^\lambda)$. A FOL family V is *almost universal* (AU) if for all polynomials $M_0, M_1\colon \mathbb{N} \to \mathbb{N}$ the function f_{M_0,M_1} is negligible, where for $\lambda \in \mathbb{N}$ we let $f_{M_0,M_1}(\lambda) = \mathbf{Coll}_{\mathsf{V}}(\lambda, M_0(\lambda), M_1(\lambda))$.

Now let $\overline{\mathsf{U}}$ be a (FOL) AU family having $\overline{\mathsf{U}}.\mathsf{IL} = \mathbb{N}$. We introduce a transform called $\mathsf{Hash\text{-}then\text{-}Mask}$ that given $\overline{\mathsf{U}}$ returns the family $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$ defined in the right panel of Fig. 7. It has $\mathsf{U.ol} = \overline{\mathsf{U}}.\mathsf{ol}$ and $\mathsf{U.IL} = \mathbb{N}$. Lemma 4.1 below shows that U is itself an AU family.

Lemma 4.1 Let $\overline{\mathsf{U}}$ be a (FOL) AU hash of $\overline{\mathsf{U}}.\mathsf{IL} = \mathbb{N}$. Let $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$. Then for any $\lambda, m, m' \in \mathbb{N}$ we have (a) $\mathbf{Coll1}_{\mathsf{U}}(\lambda, m) \le \mathbf{Coll1}_{\overline{\mathsf{U}}}(\lambda, m) + 2^{-\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ and (b) $\mathbf{Coll2}_{\mathsf{U}}(\lambda, m, m') \le \mathbf{Coll2}_{\overline{\mathsf{U}}}(\lambda, m, m') + 2/2^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$. ▌

The proof of Lemma 4.1 is in Appendix C. Note that BHK2 [9] provide an extremely fast construction of an AU family $\overline{\mathsf{U}}$, running at 0.4 cycles per byte. Our $\mathsf{Hash\text{-}then\text{-}Mask}$ does not degrade speed much, and thus the family $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$ used in our scheme is also fast.

Now let $\overline{\mathsf{H}}$ be a function family with FIL $\overline{\mathsf{H}}.\mathsf{il}$ and with $\overline{\mathsf{H}}.\mathsf{OL} = \mathbb{N}$. Let U be a FOL AU function family with $\mathsf{U}.\mathsf{ol} = \overline{\mathsf{H}}.\mathsf{il}$ and with $\mathsf{U}.\mathsf{IL} = \overline{\mathsf{H}}.\mathsf{OL} = \mathbb{N}$. The $\mathsf{AU\text{-}then\text{-}Hash}$ transform of BHK2 [9] takes $\mathsf{U}, \overline{\mathsf{H}}$ and returns the family $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$ shown in the middle panel of Fig. 7. It has $\mathsf{H}.\mathsf{OL} = \mathsf{H}.\mathsf{IL} = \mathbb{N}$. BHK2 [9] show that if $\overline{\mathsf{H}}$ is $\mathsf{UCE}[\mathcal{S}^{\sup}]$ then so is H.

We are finally ready to define our $\mathsf{HE1}$ scheme. Let $\overline{\mathsf{H}}$ be a function family with FIL $\overline{\mathsf{H}}.\mathsf{il}$ and with $\overline{\mathsf{H}}.\mathsf{OL} = \mathbb{N}$. Let $\overline{\mathsf{U}}$ be a (FOL) AU family having $\overline{\mathsf{U}}.\mathsf{IL} = \mathbb{N}$. Let LT be an LTDF. Let $\ell\colon \mathbb{N} \to \mathbb{N}$ be a polynomial. Then let $\mathsf{HE1} = \mathsf{Hedge}[\mathsf{H}, \mathsf{LT}]$ with $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$ and $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$. A subtle point is that we set $\mathsf{HE1}.\mathsf{il} = \ell$, meaning $\mathsf{HE1}$ is restricted to encrypt messages of length ℓ. Why this is needed is not evident from the scheme description but will be needed in the proof of security. We also set $\mathsf{HE1}.\mathsf{rl} = \overline{\mathsf{U}}.\mathsf{ol}$. Theorem 4.2 below shows that $\mathsf{HE1}$ is H-IND secure. The concrete security statements refer to

$$\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\lambda, p, \sigma) = \max\left\{ \sum_{i=1}^{k}\sum_{j=1}^{k'} \mathbf{Coll}_{\mathsf{U}}(\lambda, m_i, m'_j) \;:\; k \le p, k' \le p, \sum_{i=1}^{k} m_i \le \sigma, \sum_{i=1}^{k'} m'_i \le \sigma \right\} .$$

Theorem 4.2 Let $\overline{\mathsf{H}}$ be a $\mathsf{UCE}[\mathcal{S}^{\sup}]$ function family with FIL $\overline{\mathsf{H}}.\mathsf{il}$ and with $\overline{\mathsf{H}}.\mathsf{OL} = \mathbb{N}$. Let $\overline{\mathsf{U}}$ be a (FOL) AU family having $\overline{\mathsf{U}}.\mathsf{IL} = \mathbb{N}$. Let LT be an LTDF. Let $\ell\colon \mathbb{N} \to \mathbb{N}$ be a polynomial. Let $\mathsf{HE1}$ be defined from $\overline{\mathsf{H}}, \overline{\mathsf{U}}, \mathsf{LT}, \ell$ as above.

Asymptotic result: HE1 is H-IND secure.

Concrete IND-CPA result: Let A be an adversary and $\overline{P}$ be a predictor. We can construct a source $\overline{S}$, a distinguisher $\overline{D}$ and an adversary B such that

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE1},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2^{1-\overline{\mathsf{U}}.\mathsf{ol}}$$

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le \frac{\sqrt{q}}{2^{\tau/2}} + \sqrt{q \cdot \mathbf{Coll2}_{\overline{\mathsf{U}}}(\cdot, \mathsf{LT}.\mathsf{il})} + \frac{2\sqrt{q}}{2^{\overline{\mathsf{U}}.\mathsf{ol}/2}}$$

where q is the maximum of the size of $\overline{P}$'s output in the execution of $\mathrm{Pred}^{\overline{P}}_{\overline{S}}$ and τ is the lossiness of LT. Furthermore, $\mathbf{T}(\mathrm{Lossy}^{B}_{\mathsf{LT}}), \mathbf{T}(\mathrm{UCE}^{\overline{S},\overline{D}}_{\overline{\mathsf{H}}})$; and $\mathbf{Q}^{\mathrm{HASH}}_{\overline{S}} = 2$.

Concrete IND-CDA result: Let A be an adversary and $\overline{P}$ be a predictor. We can construct a source $\overline{S}$, a distinguisher $\overline{D}$ and an adversary B such that

$$\mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE1},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + 2\mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 2p, s) + 3p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{19p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}}}$$

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le \sqrt{2q \cdot \mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 2p, s)} + 2p\sqrt{q \cdot \mathrm{Guess}_A(\cdot)} + \frac{6p\sqrt{q}}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \tau\}/2}}$$

where p is the maximum of the total number of messages that A produces in the execution of $\mathrm{CDA}^{A}_{\mathsf{HE1}}$, $s = p \cdot (\overline{\mathsf{U}}.\mathsf{ol} + \mathsf{LT}.\mathsf{il} + \ell)$, q is the maximum of the size of $\overline{P}$'s output in the execution of $\mathrm{Pred}^{\overline{P}}_{\overline{S}}$, and τ is the lossiness of LT. Moreover, $\mathbf{T}(\mathrm{Lossy}^{B}_{\mathsf{LT}}), \mathbf{T}(\mathrm{UCE}^{\overline{S},\overline{D}}_{\overline{\mathsf{H}}}) \le \mathbf{T}(\mathrm{CDA}^{A}_{\mathsf{HE1}})$; and $\mathbf{Q}^{\mathrm{HASH}}_{\overline{S}} \le 2p$. ▮

The proof of Theorem 4.2 is in Appendix D. Here we discuss some of the ideas. For IND-CPA security, recall that the adversary A makes only a single LR query. The transform $\mathsf{Hash\text{-}then\text{-}Mask}$ ensures that, for any string m, if r is a random $\overline{\mathsf{U}}.\mathsf{ol}(\lambda)$-bit string and $uk \leftarrow\!\!\$\; \mathsf{U}.\mathsf{Kg}(1^\lambda)$ then $u \leftarrow \mathsf{U}(1^\lambda, uk, r \,\|\, m)$ is also uniformly random, independent of m. Therefore, one doesn't need to know m to sample $r \leftarrow\!\!\$\; \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ and compute $x \leftarrow \mathsf{H}.\mathsf{Ev}(1^\lambda, hk, r \,\|\, m, 1^{\mathsf{LT}.\mathsf{il}(\lambda)})$, because one can instead sample $u \leftarrow\!\!\$\; \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ and compute $x \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, u, 1^{\mathsf{LT}.\mathsf{il}(\lambda)})$. The source will leak $\mathsf{H}.\mathsf{Ev}(1^\lambda, hk, x, 1^{|m|})$ so that the distinguisher can run A to get m and xor the two strings to complete the ciphertext. Still, computing $\mathsf{H}.\mathsf{Ev}(1^\lambda, hk, x, 1^{|m|})$

```
$\underline{\mathsf{HE2.Enc}(1^\lambda, (ek, hk), m; r)}$
$x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, r \,\|\, m, 1^{\mathsf{LT.il}(\lambda)})$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, ek, x)$
$seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda)+\overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
$y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m|}) \oplus mask \oplus m$
Return $(trap, c)$

$\underline{\mathsf{HE2.Dec}(1^\lambda, (dk, hk), (trap, c))}$
$x \leftarrow \mathsf{LT.Inv}(1^\lambda, dk, trap)$
$seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda)+\overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
$y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|c|})$
$m \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|c|}) \oplus mask \oplus c$
Return $m$
```

Figure 8: Encryption and decryption algorithms of HE2, where $\overline{\mathsf{U}}$ is an AU family, $\overline{\mathsf{H}}$ is a FIL $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ family, F is a VOL PRF, LT is a LTDF. Here $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$ and $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$.

requires knowing $|m|$; it's why HE1 can only handle fixed-length messages. For IND-CDA security, we can actually prove that $\mathsf{Hedge}[\mathsf{H}, \mathsf{LT}]$ is IND-CDA secure for *any* $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ H. The source will run A_1 and the first phase of A_2 to create the ciphertexts via the HASH oracle. Note that during the first phase, A_2 only receives what the source sees, and therefore doesn't get to learn the hash key hk. UCE then allows us to switch to a game in which the adversary has to fight an RO-based scheme, and thus its adaptivity is futile. Moreover, it can only specify *distributions*, and thus despite the adaptivity, the chance that the source repeats a HASH query is about $p^2 \cdot \mathrm{Guess}_A$. We again exploit the lossiness of LT to allow statistical unpredictability.

The HE2 scheme. With HE1 we reach our goal of the first fully H-IND secure PKE scheme in the standard model. Additionally it is more efficient than prior standard-model schemes that only achieved non-full security. However, like prior standard-model schemes, it is FIL, meaning only encrypts messages of a fixed length. We now provide the HE2 scheme that retains the security properties of HE1 but additionally can encrypt messages of variable and arbitrary length. Furthermore it can do this with hybrid-encryption like performance, meaning the asymmetric cost is fixed as message length grows.

The additional tool that we need is a VOL PRF F —this means $\mathsf{F.OL}(\cdot) = \mathbb{N}$— such that $\lambda \in \mathsf{F.IL}(\lambda)$ for every $\lambda \in \mathbb{N}$. As before let $\overline{\mathsf{H}}$ be a function family with FIL $\overline{\mathsf{H}}.\mathsf{il}$ and with $\overline{\mathsf{H}}.\mathsf{OL}(\cdot) = \mathbb{N}$. Let $\overline{\mathsf{U}}$ be a (FOL) AU family having $\overline{\mathsf{U}}.\mathsf{IL}(\cdot) = \mathbb{N}$. Let LT be an LTDF. Let $\mathsf{U} = \mathsf{Hash\text{-}then\text{-}Mask}[\overline{\mathsf{U}}]$ and $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$. The encryption and decryption algorithms of HE2 are specified in Fig. 8. The key-generation algorithm HE2.Kg is the same as HE1.Kg. We let $\mathsf{HE2.rl} = \overline{\mathsf{U}}.\mathsf{ol}$. But this time $\mathsf{HE2.IL}(\cdot) = \mathbb{N}$, meaning we can encrypt messages of any length. Theorem 4.3 below formally confirms that HE2 is H-IND secure.

Theorem 4.3 Let F be a PRF with $\mathsf{F.OL}(\cdot) = \mathbb{N}$ and $\lambda \in \mathsf{F.IL}(\lambda)$ for every $\lambda \in \mathbb{N}$. Let $\overline{\mathsf{H}}$ be a $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ function family with FIL $\overline{\mathsf{H}}.\mathsf{il}$ and with $\overline{\mathsf{H}}.\mathsf{OL}(\cdot) = \mathbb{N}$. Let $\overline{\mathsf{U}}$ be a (FOL) AU family having $\overline{\mathsf{U}}.\mathsf{IL}(\cdot) = \mathbb{N}$. Let LT be an LTDF. Let HE2 be defined from $\mathsf{F}, \overline{\mathsf{H}}, \overline{\mathsf{U}}, \mathsf{LT}$ as above.

Asymptotic result: HE2 is H-IND secure.

Concrete IND-CPA result: Let A be an adversary and $\overline{P}$ be a predictor. We can construct a source $\overline{S}$, a distinguisher $\overline{D}$, adversaries B and C such that

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE2},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},C}(\cdot) + 2^{1-\overline{\mathsf{U}}.\mathsf{ol}}$$

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le \frac{\sqrt{q}}{2^{\tau/2}} + \sqrt{q \cdot \mathbf{Coll2}_{\overline{\mathsf{U}}}(\cdot, \mathsf{LT.il})} + \frac{2\sqrt{q}}{2^{\overline{\mathsf{U}}.\mathsf{ol}/2}}$$

where q is the maximum of the size of $\overline{P}$'s output in the execution of $\mathrm{Pred}^{\overline{P}}_{\overline{S}}$ and τ is the lossiness of LT. Furthermore, $\mathbf{T}(\mathrm{Lossy}^{B}_{\mathsf{LT}}), \mathbf{T}(\mathrm{UCE}^{\overline{S},\overline{D}}_{\overline{\mathsf{H}}}), \mathbf{T}(\mathrm{PRF}^{C}_{\mathsf{F}}) \le \mathbf{T}(\mathrm{CPA}^{A}_{\mathsf{HE3}})$; $\mathbf{Q}^{\mathrm{RR}}_{C} = 1$; and $\mathbf{Q}^{\mathrm{HASH}}_{S} = 2$.

Concrete IND-CDA result: Let A be an adversary and $\overline{P}$ be a predictor. We can construct a source $\overline{S}$, a distinguisher $\overline{D}$, adversary B such that

$$\mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},\overline{S},\overline{D}}(\cdot) + 2\mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 3p, s) + 5p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{44p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol},\mathsf{LT}.\mathsf{il}\}}}$$

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le \sqrt{2q\mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 3p, s)} + 2.5p\sqrt{q \cdot \mathrm{Guess}_A(\cdot)} + \frac{9.5p\sqrt{q}}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol},\tau\}/2}}$$

where p is the maximum of the total number of messages that A produces in the execution of $\mathrm{CDA}^A_{\mathsf{HE2}}$, s is $3p \cdot \max\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}$ plus the maximum of the total length of messages that A produces in the execution of $\mathrm{CDA}^A_{\mathsf{HE2}}$, q is the maximum of the size of $\overline{P}$'s output in the execution of $\mathrm{Pred}^{\overline{P}}_{\overline{S}}$, and τ is the lossiness of LT. Moreover, $\mathbf{T}(\mathrm{Lossy}^B_{\mathsf{LT}}), \mathbf{T}(\mathrm{UCE}^{\overline{S},\overline{D}}_{\mathsf{H}}) \le \mathbf{T}(\mathrm{CDA}^A_{\mathsf{HE2}})$; and $\mathbf{Q}^{\mathrm{HASH}}_{\overline{S}} \le 3p$. ▌

The proof of Theorem 4.3 is in Appendix E. Here we give some intuition about why $\mathsf{HE2}$ can securely handle variable-length messages. We'll only discuss the IND-CPA case, in which the message length may depend on the public key. The source will be responsible for producing a PRF key fk, whose length is independent of the public key, and will leak it along with some other information. The UCE security is only used to ensure that fk looks random to the distinguisher. The task of generating the two pads $\mathsf{F}.\mathsf{Ev}(1^\lambda, fk, 0^{\mathsf{F}.\mathsf{il}(\lambda)}, 1^{|m|})$ and $\mathsf{H}.\mathsf{Ev}(1^\lambda, hk, y, 1^{|m|})$ is left to the distinguisher who runs A to get m. Note that the distinguisher always creates $\mathsf{H}.\mathsf{Ev}(1^\lambda, hk, y, 1^{|m|})$ regardless of the challenge bit of game UCE. We then use the PRF security of F to ensure that the first pad looks random to A. Consequently, in the string $(trap, c)$ that A receives, the first component is independent of the message, and the second component is indistinguishable from a random string.

The $\mathsf{HE3}$ scheme. Consider the $p\sqrt{q \cdot \mathrm{Guess}_A(\cdot)}$ term in the concrete bound for IND-CDA security in Theorem 4.3. This is worse than the "optimal" bound $p(q+p) \cdot \mathrm{Guess}_A(\cdot)$ if one uses a random oracle. Why does this gap matter? Asymptotically, we know that $\mathrm{Guess}_A(\cdot)$ is negligible, and hence this entire term is negligible too, under either of the two bounds. But concretely, the first bound means that we must have more min-entropy in the messages to get security. This is not desirable in practice. For example if we encrypt passwords, their min-entropy may be borderline. Thus it would be desirable to have a better bound. Moreover, it would also be desirable to give a simple construction based on a *generic* UCE-secure hash. We achieve both goals with our $\mathsf{HE3}$ scheme.

The only ingredients we need this time are a PRF F (with fixed input length $\mathsf{F}.\mathsf{il}$ and $\mathsf{F}.\mathsf{OL}(\cdot) = \mathbb{N}$), a $\mathsf{UCE}[\mathcal{S}^{\mathrm{srs}}]$ family H (with $\mathsf{H}.\mathsf{IL}(\cdot) = \mathsf{H}.\mathsf{OL}(\cdot) = \mathbb{N}$) and a LTDF LT. We let $\rho\colon \mathbb{N} \to \mathbb{N}$ be a polynomial that is a parameter of the scheme. The encryption and decryption algorithms of $\mathsf{HE3}$ are shown in Fig. 9 and the key-generation algorithm $\mathsf{HE3}.\mathsf{Kg}$ is the same as $\mathsf{HE1}.\mathsf{Kg}$. We let $\mathsf{HE3}.\mathsf{rl} = \rho$. We also let $\mathsf{HE3}.\mathsf{IL}(\cdot) = \mathbb{N}$, meaning the scheme encrypts variable and arbitrary length messages. While the scheme is quite simple it's challenging to find an analysis to match the desired bound $p(q+p) \cdot \mathrm{Guess}_A(\cdot)$ for the reset-advantage in the IND-CDA setting. A naive analysis will end up in an inferior bound $q^2 p \cdot \mathrm{Guess}_A(\cdot)$. Let $(m_1, r_1), \ldots, (m_p, r_p)$ be the message-coin pairs specified by A's IND-CDA queries. The reset adversary R is given a random oracle RO that on input (x, ℓ), returns a random string of length ℓ. Let Bad be the event that R queries $y \leftarrow \mathsf{RO}(m_k, \rho(\lambda))$ and then queries $\mathsf{RO}(y \oplus r_k, \mathsf{F}.\mathsf{kl}(\lambda) + \lambda)$ for some $k \le p$. For $\mathsf{HE3}$ to be IND-CDA secure, Bad must not occur. Suppose that R queries $\mathsf{RO}(x_1, \rho(\lambda)), \ldots, \mathsf{RO}(x_{\lfloor q/2 \rfloor}, \rho(\lambda))$, and then queries $\mathsf{RO}(z_1, \mathsf{F}.\mathsf{kl}(\lambda) + \lambda), \ldots, \mathsf{RO}(z_{\lfloor q/2 \rfloor}, \mathsf{F}.\mathsf{kl}(\lambda) + \lambda)$. If there are $i, j \le \lfloor q/2 \rfloor$ and $k \le p$ such that $x_i = m_k$ and $\mathsf{RO}(x_i, \rho(\lambda)) \oplus z_j = r_k$ then Bad occurs. This seems to happen with probability $\frac{q^2 p}{4} \mathrm{Guess}_A(\cdot)$, because R can *adaptively* choose z_j after seeing $\mathsf{RO}(x_1, \rho(\lambda)), \ldots, \mathsf{RO}(x_{\lfloor q/2 \rfloor}, \rho(\lambda))$.

To tackle this problem, we exploit a combinatorial technique on the coin length ρ—a parameter that we fully control. From Lemma 4.4 below, the chance that Bad occurs is at most $qp \cdot \mathrm{Guess}_A(\cdot) + q^2 p \cdot 2^{-\rho(\lambda)/3}$. If ρ is large enough, say $\rho(\lambda) \ge 4.5\lambda$ for every $\lambda \in \mathbb{N}$, then this matches the optimal bound. The proof of Lemma 4.4 is in Appendix F.

Lemma 4.4 Let U, V be random variables over $\{0,1\}^*$ and $\{0,1\}^\ell$, respectively. Assume that the

```
HE3.Enc($1^\lambda, (ek, hk), m; r$)
    $w \leftarrow \mathsf{H.Ev}(1^\lambda, hk, m, 1^{|r|}) \oplus r$
    $x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, w, 1^{\mathsf{LT.il}(\lambda)})$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, ek, x)$
    $seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
    $y \leftarrow seed[1, \lambda]$ ; $fk \leftarrow seed[\lambda+1, |seed|]$
    $mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m|})$
    $c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m|}) \oplus mask \oplus m$
    Return $(trap, c)$

HE3.Dec($1^\lambda, (dk, hk), (trap, c)$)
    $x \leftarrow \mathsf{LT.Inv}(1^\lambda, dk, trap)$
    $seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
    $y \leftarrow seed[1, \lambda]$ ; $fk \leftarrow seed[\lambda+1, |seed|]$
    $mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|c|})$
    $m \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|c|}) \oplus mask \oplus c$
    Return $m$
```

Figure 9: Encryption and decryption algorithms of HE3, where H is a UCE[$\mathcal{S}^{\rm srs}$] family, F is a VOL PRF and LT is a LTDF.

maximum, over all u, v, of $\Pr[(U,V) = (u,v)]$, is at most ϵ. Let RO be a random oracle and let $W = \mathsf{RO}(U, \ell) \oplus V$. For any adversary A that makes at most q queries to RO, the probability that the first component of one of A's RO queries is W is at most $q\epsilon + q^2 \cdot 2^{-\ell/3}$. ▮

Theorem 4.5 below confirms that HE3 is H-IND secure with very good concrete security bounds. While UCE[$\mathcal{S}^{\rm sup}$] is enough for IND-CPA security, IND-CDA requires the stronger UCE[$\mathcal{S}^{\rm srs}$] assumption. The proof is in Appendix G.

Theorem 4.5 Let F be a PRF with $\mathsf{F.OL}(\cdot) = \mathbb{N}$ and fixed input length F.il. Let H be a UCE[$\mathcal{S}^{\rm srs}$] function family with $\mathsf{H.IL}(\cdot) = \mathsf{H.OL}(\cdot) = \mathbb{N}$. Let LT be an LTDF such that $\mathsf{LT.il}(\lambda) \geq \lambda$ for all $\lambda \in \mathbb{N}$. Let ρ: $\mathbb{N} \to \mathbb{N}$ be a polynomial such that $\rho(\lambda) \geq \lambda$ for all $\lambda \in \mathbb{N}$. Let HE3 be defined from $\mathsf{F}, \mathsf{H}, \mathsf{LT}, \rho$ as above.

Asymptotic result: HE3 is H-IND secure.

Concrete IND-CPA result: Let A be an adversary and P be a predictor. We can construct a source S, a distinguisher D, adversaries B and C such that

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE3},A}(\cdot) \leq 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) + 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},C}(\cdot) + 2^{1-\rho}$$

$$\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot) \leq \frac{2q}{2^\rho} + \frac{q}{2^\tau}$$

where q is the maximum of the size of P's output in the execution of Pred^P_S and τ is the lossiness of LT. Furthermore, $\mathbf{T}(\mathrm{Lossy}^B_{\mathsf{LT}}), \mathbf{T}(\mathrm{UCE}^{S,D}_{\mathsf{H}}), \mathbf{T}(\mathrm{PRF}^C_{\mathsf{F}}) \leq \mathbf{T}(\mathrm{CPA}^A_{\mathsf{HE3}})$; $\mathbf{Q}^{\mathrm{RR}}_C = 1$; and $\mathbf{Q}^{\mathrm{HASH}}_S = 2$.

Concrete IND-CDA result: Let A be an adversary and R be a predictor. We can construct a source S, a distinguisher D, adversary B such that

$$\mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE},A}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\lambda) + 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\lambda) + p^2 \cdot \mathrm{Guess}_A(\lambda) + \frac{8p^2}{2^\lambda} + \frac{12p^2}{2^{\min\{\tau(\lambda),\rho(\lambda)\}}}$$

$$\mathsf{Adv}^{\mathsf{reset}}_{S,R}(\lambda) \leq p(p+q) \cdot \mathrm{Guess}_A(\lambda) + \frac{5p^2}{2^\lambda} + \frac{6.5p^2}{2^{\min\{\tau(\lambda),\rho(\lambda)\}}} + \frac{pq^2}{2^{\rho(\lambda)/3}}$$

where p is the maximum of the total number of messages that A produces in the execution of $\mathrm{CDA}^A_{\mathsf{HE3}}$, $q = \mathbf{Q}^{\mathrm{HASH}}_R$, and τ is the lossiness of LT. Furthermore, $\mathbf{T}(\mathrm{Lossy}^B_{\mathsf{LT}}), \mathbf{T}(\mathrm{UCE}^{S,D}_{\mathsf{H}}) \leq \mathbf{T}(\mathrm{CDA}^A_{\mathsf{HE3}})$; and $\mathbf{Q}^{\mathrm{HASH}}_S \leq 3p$. ▮

5 Efficiency and comparisons with prior schemes

Our schemes improve on prior work on both the theoretical and practical fronts. On the theoretical front, DE1 is the first standard-model D-PKE scheme that is *fully* IND secure and HE1, HE2, HE3 are the first standard-model PKE schemes achieving full H-IND, meaning IND-CPA plus *full* IND-CDA. Prior standard-model D-PKE (resp. PKE) schemes only achieved IND (resp. IND-CDA) for block sources, which assumes messages (resp. message-randomness pairs) are unpredictable even given prior ones, which is unlikely to be true in applications.

On the practical front, prior standard-model schemes fix a message length, create keys depending on it, and use only asymmetric operations, making them inflexible and inefficient. Our schemes handle variable input length messages with hybrid-encryption like efficiency, meaning the asymmetric cost is fixed and one pays only in hashing as message length grows. Exploiting fast instantiations of $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ and $\mathsf{UCE}[\mathcal{S}^{\rm srs}]$ functions [44, 9], this yields high performance.

To elaborate, recall that asymmetric primitives are orders of magnitude more expensive than symmetric ones. Crucial to making IND-CPA PKE efficient is the hybrid encryption paradigm as represented by the KEM-DEM framework [24]. Here, $\mathsf{PKE.Enc}(1^\lambda, ek, m)$ uses its coins to generate a random symmetric key K along with an encapsulation c_a of K under ek, and returns ciphertext (c_a, c_s) where c_s is a symmetric encryption of m under K. The asymmetric cost is thus fixed regardless of message length and is amortized out for long messages. Ideally, we would like a similar generic hybrid encryption paradigm for D-PKE and H-PKE. But, despite interest and search, this has not been found. The reason in part is the apparently crucial use of randomness in the choice of K. As a result, prior standard-model D-PKE and H-PKE schemes have used only asymmetric operations. This has resulted not only in fixed message lengths but in costs that are exorbitant for long messages.

Our methods and schemes change this. Although we do not provide a generic hybrid encryption paradigm for these domains, our $\mathsf{DE1}, \mathsf{HE2}$ and $\mathsf{HE3}$ schemes achieve hybrid-encryption like performance, meaning the asymmetric cost is fixed regardless of message length, and one pays only in symmetric operations — in our case this means hashing via the $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ or $\mathsf{UCE}[\mathcal{S}^{\rm srs}]$ functions— as the message length grows.

To capitalize on this for performance, good and careful instantiation of the UCE hash functions is needed. We need UCE functions H that are both VIL —variable input length, $\mathsf{H.IL}(\cdot) = \mathbb{N}$— and VOL —variable output length, $\mathsf{H.OL}(\cdot) = \mathbb{N}$. We now discuss how best to obtain these.

A simple instantiation of a UCE family is based on HMAC-SHA-256, as suggested in [8] and justified in [44]. While this yields a VIL family, it is FOL (fixed output length). A method to turn FOL UCE families into VOL ones is given in [8], but is slow. A better and faster transform is provided in Appendix H. With this we get $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ and $\mathsf{UCE}[\mathcal{S}^{\rm srs}]$ families with very good performance. These suffice for $\mathsf{DE1}, \mathsf{HE1}$ and $\mathsf{HE3}$.

But one can do even better. BHK2 [9] provide a fast FIL, VOL $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ function $\overline{\mathsf{H}}$ based on AES. They also provide a fast AU family $\overline{\mathsf{U}}$. Applying their $\mathsf{AU\text{-}then\text{-}Hash}$ transform will return a VIL, VOL $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ family H that is significantly faster than the HMAC-SHA-256 based instantiation. This suffices for $\mathsf{DE1}$ and $\mathsf{HE1}$.

Recall $\mathsf{HE2}$ needs a $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ family H of a special form, but it is based on $\mathsf{AU\text{-}then\text{-}Hash}$ and thus amenable to an efficient instantiation. Start again from $\overline{\mathsf{H}}, \overline{\mathsf{U}}$ from BHK2 as above. This time turn $\overline{\mathsf{U}}$ into U via our $\mathsf{Hash\text{-}then\text{-}Mask}$ transform —this preserves performance— and apply $\mathsf{AU\text{-}then\text{-}Hash}$ to this to get H. This $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$ family is again exceptionally fast and of the special form required for $\mathsf{HE2}$.

6 Unique-ciphertext PKE

In an algorithm-substitution attack (ASA) [11], the prescribed encryption algorithm is replaced with a subverted one that may attempt to leak information about the message to "big brother." The latter and the subverted algorithm may even share a key based on which they communicate. BPR [11] formalize the attacker goal in an ASA as compromising privacy while evading detection, the latter meaning that subverted ciphertexts are indistinguishable from real ones even given the decryption key. They focus on the symmetric setting. They give attacks showing that randomized, stateless schemes will succumb

```
$\underline{\mathsf{UE.Kg}(1^\lambda)}$
$(ek, dk) \leftarrow_\$ \mathsf{DE.Kg}(1^\lambda)$
Return $(ek, (ek, dk))$
```

```
$\underline{\mathsf{UE.Enc}(1^\lambda, ek, m)}$
$c \leftarrow \mathsf{DE.Enc}(ek, m)$
Return $c$
```

```
$\underline{\mathsf{UE.Dec}(1^\lambda, (ek, dk), c)}$
$m \leftarrow \mathsf{DE.Dec}(dk, c)$
If $m \neq \bot$ then
    $c' \leftarrow \mathsf{DE.Enc}(ek, m)$
    If $c' \neq c$ then return $\bot$
Return $m$
```

Figure 10: U-PKE scheme $\mathsf{UE} = \mathsf{UniqueCtx}[\mathsf{DE}]$ constructed from D-PKE scheme DE.

to attack. They show however that security against ASAs may be achieved by what they call unique ciphertext symmetric encryption schemes.

BPR [11] initiate the study of ASAs for PKE. Continuing that theme, we define unique ciphertext PKE. We say that a PKE scheme PKE has unique ciphertexts, or is a U-PKE scheme, if for every $\lambda \in \mathbb{N}$, every $(ek, dk) \in [\mathsf{PKE.Kg}(1^\lambda)]$, and every message m, there is at most one $c \in \{0,1\}^*$ such that $\mathsf{PKE.Dec}(1^\lambda, dk, c) = m$. Coupled with correctness, this means that for every $\lambda \in \mathbb{N}$, every $(ek, dk) \in [\mathsf{PKE.Kg}(1^\lambda)]$ and every $m \in \{0,1\}^*$ with $|m| \in \mathsf{PKE.IL}(\lambda)$ the set $[\mathsf{PKE.Enc}(1^\lambda, ek, m)]$ has size exactly one. The latter means that a unique ciphertext scheme is deterministic, meaning a D-PKE scheme.

We now ask how to design a U-PKE scheme. The natural thought is that any D-PKE scheme is a U-PKE scheme. This is not true. As an example, take any IND D-PKE scheme, and modify it so that encryption pre-pends a bit to the ciphertext that is ignored by decryption. This is still an IND D-PKE scheme, but it does not have unique ciphertexts, because if c is the encryption of m under $1^\lambda, ek$ in the starting D-PKE scheme then both $0 \,\|\, c$ and $1 \,\|\, c$ are valid ciphertexts in the new D-PKE scheme.

However, we show that one can transform any given D-PKE scheme DE into a U-PKE scheme UE. The U-PKE public key is the same as the D-PKE one, but the secret key is the pair (ek, dk) consisting of the D-PKE public key and matching secret key. Encryption is as in D-PKE. U-PKE decryption of ciphertext c first recovers the candidate message m via D-PKE decryption of c under dk and then checks that re-encrypting m under ek yields c, rejecting otherwise. $\mathsf{UE} = \mathsf{UniqueCtx}[\mathsf{DE}]$ is is formally specified in Fig. 10.

The security requirement for U-PKE contains to be IND, meaning a U-PKE scheme is treated just as a D-PKE scheme in the context of security. Applying our $\mathsf{UniqueCtx}$ to $\mathsf{DE1}$ thus yields a very efficient IND U-PKE scheme.

In the symmetric setting, unique-ciphertext encryption could be stateful and thus attain IND-CPA type security [11]. Here, a synchronized state is shared between sender and receiver. In the PKE setting, however, it is does not seem practical to assume that the sender and receiver share a synchronized state. Indeed, this would go against the spirit of public-key cryptography. As a consequence, for the benefit of unique ciphertexts, security must drop compared to IND-CPA, meaning we pay in security to protect against ASAs.

Acknowledgments

Bellare is supported in part by NSF grants CNS-1116800 and CNS-1228890. Hoang is supported in part by NSF grant 1223623. Part of the work was done when Hoang was working at UCSD, supported in part by NSF grants CNS-1116800 and CNS-1228890.

References

[1] P. Austrin, K.-M. Chung, M. Mahmoody, R. Pass, and K. Seth. On the impossibility of cryptography with tamperable randomness. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 462–479. Springer, Aug. 2014. 6, 7

[2] B. Barak, Y. Dodis, H. Krawczyk, O. Pereira, K. Pietrzak, F.-X. Standaert, and Y. Yu. Leftover hash lemma, revisited. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 1–20. Springer, Aug. 2011. 29

[3] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Aug. 2001. 3, 11

[4] M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552. Springer, Aug. 2007. 3, 11

[5] M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, and S. Yilek. Hedged public-key encryption: How to protect against bad randomness. In M. Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 232–249. Springer, Dec. 2009. 3, 4, 5, 14

[6] M. Bellare, R. Dowsley, and S. Keelveedhi. How secure is deterministic encryption? In J. Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 52–73. Springer, Mar. / Apr. 2015. 13

[7] M. Bellare, M. Fischlin, A. O'Neill, and T. Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 360–378. Springer, Aug. 2008. 3, 11, 13

[8] M. Bellare, V. T. Hoang, and S. Keelveedhi. Instantiating random oracles via UCEs. Cryptology ePrint Archive, Report 2013/424, 2013. Preliminary version in CRYPTO 2013. 3, 4, 5, 6, 8, 9, 10, 11, 13, 20, 43

[9] M. Bellare, V. T. Hoang, and S. Keelveedhi. Cryptography from compression functions: The UCE bridge to the ROM. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 169–187. Springer, Aug. 2014. 5, 6, 7, 10, 15, 16, 20, 32, 43

[10] M. Bellare, E. Kiltz, C. Peikert, and B. Waters. Identity-based (lossy) trapdoor functions and applications. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 228–245. Springer, Apr. 2012. 3

[11] M. Bellare, K. G. Paterson, and P. Rogaway. Security of symmetric encryption against mass surveillance. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 1–19. Springer, Aug. 2014. 3, 6, 20, 21

[12] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73. ACM Press, Nov. 1993. 4, 13

[13] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, May / June 2006. 7

[14] E. Birrell, K.-M. Chung, R. Pass, and S. Telang. Randomness-dependent message security. In A. Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 700–720. Springer, Mar. 2013. 7

[15] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill. Order-preserving symmetric encryption. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 224–241. Springer, Apr. 2009. 8

[16] A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 335–359. Springer, Aug. 2008. 3, 4, 11

[17] C. Bosley and Y. Dodis. Does privacy require true randomness? In S. P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 1–20. Springer, Feb. 2007. 7

[18] Z. Brakerski and G. Segev. Better security for deterministic public-key encryption: The auxiliary-input setting. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 543–560. Springer, Aug. 2011. 3, 7

[19] D. R. L. Brown. A weak-randomizer attack on RSA-OAEP with e = 3. Cryptology ePrint Archive, Report 2005/189, 2005. `http://eprint.iacr.org/2005/189`. 4, 14

[20] C. Brzuska, P. Farshim, and A. Mittelbach. Indistinguishability obfuscation and UCEs: The case of computationally unpredictable sources. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 188–205. Springer, Aug. 2014. 3, 4, 5, 10, 11, 12

[21] C. Brzuska, P. Farshim, and A. Mittelbach. Random-oracle uninstantiability from indistinguishability obfuscation. Cryptology ePrint Archive, Report 2014/867, 2014. `http://eprint.iacr.org/2014/867`. 4

[22] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In J. Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 402–414. Springer, May 1999. 9

[23] S. Checkoway, R. Niederhagen, A. Everspaugh, M. Green, T. Lange, T. Ristenpart, D. J. Bernstein, J. Maskiewicz, H. Shacham, and M. Fredrikson. On the practical exploitability of dual EC in TLS implementations. In *Proceedings of the 23rd USENIX Security Symposium*, pages 319–335, August 2014. 6

[24] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003. 5, 20

[25] Y. Dodis, A. López-Alt, I. Mironov, and S. P. Vadhan. Differential privacy with imperfect randomness. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 497–516. Springer, Aug. 2012. 7

[26] Y. Dodis, S. J. Ong, M. Prabhakaran, and A. Sahai. On the (im)possibility of cryptography with imperfect randomness. In *45th FOCS*, pages 196–205. IEEE Computer Society Press, Oct. 2004. 7

[27] Y. Dodis, D. Pointcheval, S. Ruhault, D. Vergnaud, and D. Wichs. Security analysis of pseudo-random number generators with input: /dev/random is not robust. Cryptology ePrint Archive, Report 2013/338, 2013. `http://eprint.iacr.org/2013/338`. 6

[28] L. Dorrendorf, Z. Gutterman, and B. Pinkas. Cryptanalysis of the windows random number generator. In P. Ning, S. D. C. di Vimercati, and P. F. Syverson, editors, *ACM CCS 07*, pages 476–485. ACM Press, Oct. 2007. 6

[29] A. Escala, J. Herranz, B. Libert, and C. Ràfols. Identity-based lossy trapdoor functions: New definitions, hierarchical extensions, and implications. In H. Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 239–256. Springer, Mar. 2014. 3

[30] M. Feltz and C. Cremers. On the limits of authenticated key exchange security with an application to bad randomness. Cryptology ePrint Archive, Report 2014/369, 2014. `http://eprint.iacr.org/2014/369`. 7

[31] D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev. More constructions of lossy and correlation-secure trapdoor functions. *Journal of Cryptology*, 26(1):39–74, Jan. 2013. 4, 6, 9

[32] B. Fuller, A. O'Neill, and L. Reyzin. A unified approach to deterministic encryption: New constructions and a connection to computational entropy. In R. Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 582–599. Springer, Mar. 2012. 3

[33] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, Oct. 2013. 3, 11

[34] C. Gentry, A. Lewko, A. Sahai, and B. Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. Cryptology ePrint Archive, Report 2014/309, 2014. `http://eprint.iacr.org/2014/309`. 3

[35] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, Oct. 1986. 8

[36] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. 8

[37] M. D. Green, J. Katz, A. J. Malozemoff, and H.-S. Zhou. A unified approach to idealized model separations via indistinguishability obfuscation. Cryptology ePrint Archive, Report 2014/863, 2014. `http://eprint.iacr.org/2014/863`. 4

[38] Z. Gutterman, B. Pinkas, and T. Reinman. Analysis of the linux random number generator. In *2006 IEEE Symposium on Security and Privacy*, pages 371–385. IEEE Computer Society Press, May 2006. 6

[39] B. Hemenway and R. Ostrovsky. Building lossy trapdoor functions from lossy encryption. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 241–260. Springer, Dec. 2013. 6, 7, 9

[40] N. Heninger, Z. Durumeric, E. Wustrow, and J. A. Halderman. Mining your Ps and Qs: Detection of widespread weak keys in network devices. In *Proceedings of the 21st USENIX Security Symposium*, pages 205–220, August 2012. 6

[41] S. Kamara and J. Katz. How to encrypt with a malicious random number generator. In K. Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 303–315. Springer, Feb. 2008. 7

[42] E. Kiltz, A. O'Neill, and A. Smith. Instantiability of RSA-OAEP under chosen-plaintext attack. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 295–313. Springer, Aug. 2010. 6, 9

[43] A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung, and C. Wachter. Public keys. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 626–642. Springer, Aug. 2012. 6

[44] A. Mittelbach. Salvaging indifferentiability in a multi-stage setting. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 603–621. Springer, May 2014. 6, 10, 20

[45] K. Ouafi and S. Vaudenay. Smashing SQUASH-0. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 300–312. Springer, Apr. 2009. 4, 14

[46] K. G. Paterson, J. C. N. Schuldt, and D. L. Sibborn. Related randomness attacks for public key encryption. In H. Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 465–482. Springer, Mar. 2014. 7

[47] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In R. E. Ladner and C. Dwork, editors, *40th ACM STOC*, pages 187–196. ACM Press, May 2008. 3, 4, 6, 9

[48] A. Raghunathan, G. Segev, and S. P. Vadhan. Deterministic public-key encryption for adaptively chosen plaintext distributions. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 93–110. Springer, May 2013. 3, 7

[49] T. Ristenpart and S. Yilek. When good randomness goes bad: Virtual machine reset vulnerabilities and hedging deployed cryptography. In *NDSS 2010*. The Internet Society, Feb. / Mar. 2010. 7

[50] P. Rogaway and T. Shrimpton. A provable-security treatment of the key-wrap problem. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 373–390. Springer, May / June 2006. 7

[51] D. Vergnaud and D. Xiao. Public-key encryption with weak randomness: Security against strong chosen distribution attacks. Cryptology ePrint Archive, Report 2013/681, 2013. `http://eprint.iacr.org/2013/681`. 7

[52] D. Wichs. Barriers in cryptography with weak, correlated and leaky sources. In R. D. Kleinberg, editor, *ITCS 2013*, pages 111–126. ACM, Jan. 2013. 3, 4, 10

[53] G. Yang, S. Duan, D. S. Wong, C. H. Tan, and H. Wang. Authenticated key exchange under bad randomness. Cryptology ePrint Archive, Report 2011/688, 2011. `http://eprint.iacr.org/2011/688`. 7

[54] S. Yilek. Resettable public-key encryption: How to encrypt on a virtual machine. In J. Pieprzyk, editor, *CT-RSA 2010*, volume 5985 of *LNCS*, pages 41–56. Springer, Mar. 2010. 7

[55] A. Young and M. Yung. Kleptography: Using cryptography against cryptography. In W. Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 62–74. Springer, May 1997. 3, 6

A Proof of Proposition 3.1

For part (1), consider the following adversary $A = (A_1, A_2)$. Adversary A_1 samples $m_0 \leftarrow_\$ \{0,1\}^\lambda 0$ and $m_1 \leftarrow_\$ \{0,1\}^\lambda 1$. Adversary A_2, given the public key (ek, hk) and the ciphertext $c = (c', P)$, will run $z \leftarrow \mathsf{G.Ev}(P, hk)$ and output the last bit of z. Let b be the challenge bit of game $\mathrm{IND}^A_{\mathsf{DE1}}$. Note that P is the obfuscated circuit of $C_{1^\lambda,x,y}$, with $x = ek \,\|\, m_b$, $fk = \mathsf{H.Ev}(1^\lambda, hk, m_b, 1^{\mathsf{H.ol}(\lambda)})[1, \mathsf{F.kl}(\lambda)]$, and $y = \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{\mathsf{F.kl}(\lambda)+\lambda})$. Then $C_{1^\lambda,x,y}(hk)$ will return x, due to the construction of $C_{1^\lambda,x,y}$. The

```
Game $H_1^A(\lambda)$, $\boxed{H_2^A(\lambda)}$
    $(ek, dk) \leftarrow\$ \overline{\mathsf{RE}}.\mathsf{Kg}(1^\lambda)$ ; $(m_0, m_1, t) \leftarrow\$ A(1^\lambda, ek)$
    $b \leftarrow\$ \{0,1\}$ ; $fk \leftarrow\$ \{0,1\}^{\mathsf{F.kl}(\lambda)}$ ; $x \leftarrow ek \,\|\, m_b$
    $\ell \leftarrow \mathsf{F.kl}(\lambda) + \lambda$ ; $y \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^\ell)$ ; $\boxed{y \leftarrow\$ \{0,1\}^\ell}$
    $r_1 \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0 \,\|\, 1^{\mathsf{F.il}(\lambda)-1}, 1^{\mathsf{G.rl}(\lambda)})$ ; $\boxed{r_1 \leftarrow\$ \{0,1\}^{\mathsf{G.rl}(\lambda)}}$
    $r_2 \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 1^{\mathsf{F.il}(\lambda)}, 1^{\overline{\mathsf{RE}}.\mathsf{rl}(\lambda)})$ ; $\boxed{r_2 \leftarrow\$ \{0,1\}^{\overline{\mathsf{RE}}.\mathsf{rl}(\lambda)}}$
    $P \leftarrow \mathsf{G.Ob}(1^\lambda, C_{\lambda,x,y}; r_1)$
    $c' \leftarrow \overline{\mathsf{RE}}.\mathsf{Enc}(1^\lambda, ek, m_b; r_2)$ ; $c \leftarrow (c', P)$
    $b' \leftarrow\$ A(1^\lambda, t, c)$ ; Return $(b = b')$

Game $H_3^A(\lambda)$
    $(ek, dk) \leftarrow\$ \overline{\mathsf{RE}}.\mathsf{Kg}(1^\lambda)$ ; $(m_0, m_1, t) \leftarrow\$ A(1^\lambda, ek)$
    $b \leftarrow\$ \{0,1\}$ ; $c' \leftarrow\$ \overline{\mathsf{RE}}.\mathsf{Enc}(1^\lambda, ek, m_b)$
    $P \leftarrow\$ \mathsf{G.Ob}(1^\lambda, R_{1^\lambda})$ ; $c \leftarrow (c', P)$
    $b' \leftarrow\$ A(1^\lambda, t, c)$ ; Return $(b = b')$
```

Figure 11: **Games for the proof of Proposition 3.1**. Games H_2 contains the corresponding boxed statement, but games H_1 doesn't. In game H_3, circuit R_{1^λ} has the same size as $C_{1^\lambda,x,y}$ constructed in games H_1 and H_2, but always outputs $0^{\mathsf{H.il}(\lambda)}$.

```
$B_1^{\mathrm{RR}}(1^\lambda)$
    $b \leftarrow\$ \{0,1\}$ ; $(ek, dk) \leftarrow\$ \overline{\mathsf{RE}}.\mathsf{Kg}(1^\lambda)$ ; $(m_0, m_1, t) \leftarrow\$ A(1^\lambda, ek)$ ; $x \leftarrow ek \,\|\, m_b$
    $y \leftarrow \mathrm{RR}(0^{\mathsf{F.il}(\lambda)}, 1^{\mathsf{F.kl}(\lambda)+\lambda})$ ; $r_1 \leftarrow \mathrm{RR}(0 \,\|\, 1^{\mathsf{F.il}(\lambda)-1}, 1^{\mathsf{G.rl}(\lambda)})$ ; $r_2 \leftarrow \mathrm{RR}(1^{\mathsf{F.il}(\lambda)}, 1^{\overline{\mathsf{RE}}.\mathsf{rl}(\lambda)})$
    $P \leftarrow \mathsf{G.Ob}(1^\lambda, C_{1^\lambda,x,y}; r_1)$ ; $c' \leftarrow \overline{\mathsf{RE}}.\mathsf{Enc}(1^\lambda, ek, m_b; r_2)$
    $c \leftarrow (c', P)$ ; $b' \leftarrow\$ A(1^\lambda, t, c)$
    If $(b = b')$ then return 1 else return 0

$B_2(1^\lambda)$
    $b \leftarrow\$ \{0,1\}$ ; $(ek, dk) \leftarrow\$ \overline{\mathsf{RE}}.\mathsf{Kg}(1^\lambda)$
    $(m_0, m_1, t) \leftarrow\$ A(1^\lambda, ek)$
    $x \leftarrow ek \,\|\, m_b$ ; $y \leftarrow\$ \{0,1\}^{\mathsf{F.kl}(\lambda)+\lambda}$
    $c' \leftarrow\$ \overline{\mathsf{RE}}.\mathsf{Enc}(1^\lambda, ek, m_b)$ ; $t' \leftarrow (1^\lambda, b, c', t)$
    $b' \leftarrow\$ A(1^\lambda, t, c)$ ; Return $(R_{1^\lambda}, C_{1^\lambda,x,y}, t')$

$B_2(t', P)$
    $(1^\lambda, b, c', t) \leftarrow t'$ ; $c \leftarrow (c', P)$ ; $b' \leftarrow\$ A(1^\lambda, t, c)$
    If $(b' = b)$ then return 1 else return 0

$B_3(1^\lambda, ek)$
    $b \leftarrow\$ \{0,1\}$ ; $(m_0, m_1, t) \leftarrow\$ A(1^\lambda, ek)$
    $P \leftarrow\$ \mathsf{G.Ob}(1^\lambda, R_{1^\lambda})$
    $t' \leftarrow (t, P, b)$ ; Return $(m_0, m_1, t')$

$B_3(1^\lambda, t', c')$
    $(t, P, b) \leftarrow t'$ ; $c \leftarrow (c', P)$
    $b' \leftarrow\$ A(1^\lambda, t, c)$
    If $(b = b')$ then return 1 else return 0
```

Figure 12: Adversaries B_1, B_2, and B_3 in the proof of Proposition 3.1 that attack F, G, and $\overline{\mathsf{RE}}$ respectively. Circuit R_{1^λ} has the same size as $C_{1^\lambda,x,y}$ but always outputs $0^{\mathsf{H.il}(\lambda)}$.

obfuscated circuit P is created by running $\mathsf{G.Ob}$ on deterministic coins, but the correctness of iO holds for any coins, so we're still able to evaluate P, and thus $\mathsf{G.Ev}(P, hk)$ also returns x. Hence the evaluation result z that A_2 obtains is $ek \,\|\, m_b$ whose last bit is exactly b. The adversary then wins with advantage 1.

For part (2), let A be an adversary attacking RE. Consider the games H_1–H_3 in Fig. 11. Game $H_1^A(\lambda)$ corresponds to game $\mathrm{CPA}_{\mathsf{RE}}^A(\lambda)$. Game $H_2^A(\lambda)$ is identical to game $H_1^A(\lambda)$, except that instead of using pseudorandom strings y, r_1, r_2, we sample them at random. Consider the adversary B_1 in Fig. 12 that attacks F. It simulates game H_1^A, but calls to $\mathsf{F}(1^\lambda, fk, \cdot, \cdot)$ are replaced by corresponding queries to B's oracle. Let a_1 be the challenge bit of game $\mathrm{PRF}_{\mathsf{F}}^{B_1}$. Then

$$\Pr[\,\mathrm{PRF}_{\mathsf{F}}^{B_1}(\cdot) \Rightarrow \mathsf{true} \,|\, a_1 = 1\,] = \Pr[H_1^A(\cdot)] \quad \text{and} \quad \Pr[\,\mathrm{PRF}_{\mathsf{F}}^{B_1}(\cdot) \Rightarrow \mathsf{false} \,|\, a_1 = 0\,] = \Pr[H_2^A(\cdot)] \ .$$

Hence $\mathsf{Adv}_{\mathsf{F},B_1}^{\mathsf{prf}}(\cdot) = \Pr[H_1^A(\cdot)] - \Pr[H_2^A(\cdot)]$. Next, game $H_3^A(\lambda)$ is identical to game $H_2^A(\lambda)$, except that instead of obfuscating circuit $C_{1^\lambda,x,y}$, we obfuscate a constant circuit R_{1^λ} of the same size that always outputs $0^{\mathsf{H.il}(\lambda)}$. Consider the adversary B_2 attacking G in Fig. 12. It first simulates game $H_2^A(\lambda)$ to obtain $C_{1^\lambda,x,y}$, and outputs $(R_{1^\lambda}, C_{1^\lambda,x,y})$. It then continues simulating $H_2^A(\lambda)$ but instead of obfuscating

```
$\underline{S^{\textsc{Hash}}(1^\lambda)}$
$lk \leftarrow\$ \mathsf{LT.LKg}(1^\lambda)$ ; $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow\$ A_1(1^\lambda)$ ; $b \leftarrow\$ \{0,1\}$
For $i = 1$ to $|\mathbf{m}_0|$ do
    $r \leftarrow \textsc{Hash}(1^\lambda, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$ ; $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, r)$
    $c \leftarrow \textsc{Hash}(1^\lambda, r, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $(b, lk, \mathbf{c})$

$\underline{D(1^\lambda, hk, L)}$
$(b, lk, \mathbf{c}) \leftarrow L$
$b' \leftarrow\$ A_2(1^\lambda, (lk, hk), \mathbf{c})$
Return $(b = b')$
```

```
$\underline{B(1^\lambda, K)}$
$(\mathbf{m}_0, \mathbf{m}_1) \leftarrow\$ A_1(1^\lambda)$ ; $hk \leftarrow\$ \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow\$ \{0,1\}$
For $i = 1$ to $|\mathbf{m}_0|$ do
    $r \leftarrow \mathsf{H.Ev}(1^\lambda, hk, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$ ; $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, r)$
    $c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, r, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
$b' \leftarrow\$ A_2(1^\lambda, (K, hk), \mathbf{c})$ ; Return $(b = b')$
```

Figure 13: **Top:** Source S and distinguisher D in the proof of Theorem 3.2. **Bottom:** Adversary B attacking LT in the proof of Theorem 3.2.

```
$\underline{\text{Game } G_1^A(\lambda), \boxed{G_2^A(\lambda)}}$
$(\mathbf{m}_0, \mathbf{m}_1) \leftarrow\$ A_1(1^\lambda)$ ; $hk \leftarrow\$ \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow\$ \{0,1\}$
$(ek, dk) \leftarrow\$ \mathsf{LT.EKg}(1^\lambda)$ ; $\boxed{ek \leftarrow\$ \mathsf{LT.LKg}(1^\lambda)}$
For $i = 1$ to $|\mathbf{m}_0|$ do
    $r \leftarrow \mathsf{H.Ev}(1^\lambda, hk, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$
    $c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, r, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, ek, r)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
$b' \leftarrow\$ A_2(1^\lambda, (ek, hk), \mathbf{c})$ ; Return $(b = b')$

$\underline{\text{Game } \boxed{G_3^A(\lambda)}, G_4^A(\lambda)}$
$(\mathbf{m}_0, \mathbf{m}_1) \leftarrow\$ A_1(1^\lambda)$ ; $hk \leftarrow\$ \mathsf{H.Kg}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$ ; $lk \leftarrow\$ \mathsf{LT.LKg}(1^\lambda)$
For $i = 1$ to $|\mathbf{m}_0|$ do
    $r \leftarrow \mathsf{RO}(\mathbf{m}_b[i], \mathsf{LT.il}(\lambda))$ ; $c \leftarrow \mathsf{RO}(r, |\mathbf{m}_b[i]|) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, r)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
$b' \leftarrow\$ A_2(1^\lambda, (lk, hk), \mathbf{c})$ ; Return $(b = b')$

$\underline{\mathsf{RO}(x, \ell)}$
$y \leftarrow\$ \{0,1\}^\ell$
If $H[x, \ell] \neq \bot$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{y \leftarrow H[x, \ell]}$
$H[x, \ell] \leftarrow y$ ; Return $y$
```

Figure 14: **Games G_1–G_4 in the proof of Theorem 3.2**. Games G_2 and G_3 contain the corresponding boxed statements, but games G_1 and G_4 do not.

$C_{1^\lambda,x,y}$, it uses the obfuscated circuit P that it receives. We claim that B_2 is well-formed. Since F is deterministic, the set

$$\text{Set} = \{\, z \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{\mathsf{F.kl}(\lambda)+\lambda}) \;:\; fk \in \{0,1\}^{\mathsf{F.kl}(\lambda)} \,\}$$

contains at most $2^{\mathsf{F.kl}(\lambda)}$ elements. If $y \leftarrow\$ \{0,1\}^{\mathsf{F.kl}(\lambda)+\lambda}$ then the chance that $y \in \text{Set}$ is at most $2^{-\lambda}$. Hence for $y \leftarrow\$ \{0,1\}^{\mathsf{F.kl}(\lambda)+\lambda}$, the chance that there exists $z \in \{0,1\}^{\mathsf{H.kl}(\lambda)}$ such that $C_{1^\lambda,x,y}(z) \neq 0^{|x|}$ is at most $2^{-\lambda}$. In other words, $\Pr[C_{1^\lambda,x,y} \not\equiv R_{1^\lambda}] \leq 2^{-\lambda}$, where the probability is taken over $y \leftarrow\$ \{0,1\}^{\mathsf{F.kl}(\lambda)+\lambda}$, and thus B_2 is well-formed. Let a_2 be the challenge bit of game $\mathrm{IO}_{\mathsf{G}}^{B_4}(\lambda)$. Then

$$\Pr[\,\mathrm{IO}_{\mathsf{G}}^{B_2}(\cdot) \Rightarrow \mathsf{true} \,|\, a_2 = 1\,] = \Pr[H_2^A(\cdot)] \quad \text{and} \quad \Pr[\,\mathrm{IO}_{\mathsf{G}}^{B_2}(\cdot) \Rightarrow \mathsf{false} \,|\, a_2 = 0\,] = \Pr[H_3^A(\cdot)] \ .$$

Consider adversary B_3 in Fig. 12 that attacks $\overline{\mathsf{RE}}$. Game $\mathrm{CPA}_{\overline{\mathsf{RE}}}^{B_3}$ coincides with game H_3^A. Summing up,

$$\begin{aligned} \mathsf{Adv}_{\mathsf{RE},A}^{\mathsf{ind\text{-}cpa}}(\cdot) = 2\Pr[\mathrm{CPA}_{\mathsf{RE}}^A(\cdot)] - 1 &= 2\big(\Pr[H_1^A(\cdot)] - \Pr[H_3^A(\cdot)]\big) \\ &= 2\mathsf{Adv}_{\mathsf{F},B_1}^{\mathsf{prf}}(\cdot) + 2\mathsf{Adv}_{\mathsf{G},B_2}^{\mathsf{io}}(\cdot) + 2\mathsf{Adv}_{\overline{\mathsf{RE}},B_3}^{\mathsf{ind\text{-}cpa}}(\cdot) \ . \end{aligned}$$

The proposition then follows.

B Proof of Theorem 3.2

Let $A = (A_1, A_2)$ be an adversary attacking $\mathsf{DE1}$. Consider the source S and distinguisher D in Fig. 13. They simulate game $\mathrm{IND}^A_{\mathsf{DE1}}$, but use a lossy key lk instead of an injective key ek, and calls to $\mathsf{H}(1^\lambda, hk, \cdot, \cdot)$ are replaced by corresponding queries to $\textsc{Hash}$. Consider games G_1–G_4 in Fig. 14. Game G_1^A coincides with game $\mathrm{IND}^A_{\mathsf{DE1}}$. We explain the game chain up to the terminal one. Game G_2^A is identical to game G_1^A, except that instead of using an injective key for LT, we use a lossy key. Consider the adversary B in Fig. 13 that attacks LT. It's given a key K, which may be either an injective key ek or a lossy key lk. It then simulates game $\mathrm{IND}^A_{\mathsf{DE1}}$, but K is used as the encryption key for LT. Then

$$\Pr[G_1^A(\cdot)] - \Pr[G_2^A(\cdot)] = \mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{DE1},B}(\cdot) \ .$$

Game G_3^A is identical to game G_2^A, except that instead of using $\mathsf{H.Ev}(1^\lambda, hk, \cdot, \cdot)$, we use a random oracle $\mathsf{RO}(\cdot, \cdot)$. Then

$$\Pr[G_2^A(\cdot)] - \Pr[G_3^A(\cdot)] \le \mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) \ .$$

Game G_4^A is identical to game G_3^A, except that RO now ignores consistency and always returns a fresh random answer for each query. The games G_3^A and G_4^A are identical-until-bad, and thus

$$\Pr[G_3^A(\cdot)] - \Pr[G_4^A(\cdot)] \le \Pr[G_4^A(\cdot) \text{ sets } \mathsf{bad}] \ .$$

Let v be a polynomial that bound $|\mathbf{m}_0|$. Since the strings $\mathbf{m}_b[1], \ldots, \mathbf{m}_b[|\mathbf{m}_0|]$ are distinct, game G_4^A sets bad only if (i) some coins r are repeated, which happens with probability at most $v^2/2^{1+\mathsf{LT.il}}$, or (ii) some coin r is identical to a message $m_b[i]$, which happens with probability at most $v^2/2^{\mathsf{LT.il}}$. Hence $\Pr[G_4^A(\cdot) \text{ sets } \mathsf{bad}] \le 3v^2/2^{1+\mathsf{LT.il}}$. Finally, $\Pr[G_4^A(\cdot)] = 1/2$ because whatever A_2 receives is independent of the challenge bit. Summing up,

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DE1},A}(\cdot) = 2\Pr[G_1^A(\cdot)] - 1 \le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) + \frac{3v^2}{2^{\mathsf{LT.il}}} \ .$$

What's left is to show that S is statistically unpredictable. Let P be a statistical predictor, and let q be a polynomial that bounds the size of P's output. Consider game $H_1^{A,P}$ and $H_2^{A,P}$ that are identical to game G_3^A and G_4^A respectively, except that at the end, we run $Q' \leftarrow\!\$\ P(1^\lambda, (b, lk, \mathbf{c}))$ and then return $(Q \cap Q' \neq \emptyset)$, where Q is the set of coins r and messages $\mathbf{m}_b[i]$. Game Pred_S^P is identical to game $H_1^{A,P}$, and $\Pr[H_1^{A,P}(\cdot)] - \Pr[H_2^{A,P}(\cdot)] \le 3v^2/2^{1+\mathsf{LT.il}}$. On the other hand, in game $H_2^{A,P}$, the predictor has no information of the messages $\mathbf{m}_b[i]$, and thus the chance that it can guess them is at most $qv\mathrm{Guess}_A(\cdot)$. Let τ be the lossiness of LT. The predictor is given the image of the strings r under LT, and thus the chance that it can guess those strings is at most $qv/2^\tau$. Hence $\Pr[H_2^{A,P}(\cdot)] \le qv\mathrm{Guess}_A(\cdot) + qv/2^\tau$, and thus

$$\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot) \le \frac{3v^2}{2^{1+\mathsf{LT.il}}} + qv\mathrm{Guess}_A(\cdot) + \frac{qv}{2^\tau} \ .$$

Hence S is statistically unpredictable.

C Proof of Lemma 4.1

For part (a), consider arbitrary strings x and y with $|x| \le m$ and $|y| = \overline{\mathsf{U}}.\mathsf{ol}(\lambda)$. If $|x| < \overline{\mathsf{U}}.\mathsf{ol}(\lambda)$ then for $uk \leftarrow\!\$\ \mathsf{U.Kg}(\lambda)$, we have $\Pr[y = \mathsf{U.Ev}(1^\lambda, uk, x)] = 2^{-\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$. If $|x| \ge \overline{\mathsf{U}}.\mathsf{ol}(\lambda)$ then let $x_1 = x[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ and $x_2 = x[\overline{\mathsf{U}}.\mathsf{ol}(\lambda) + 1, |x|]$. Then

$$\Pr[y = \mathsf{U.Ev}(1^\lambda, uk, x)] \quad \le \quad \Pr[y \oplus (x_1 \cdot rk) = \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, x)] \le \mathbf{Coll1}_{\overline{\mathsf{U}}}(\lambda, m);$$

the probability is taken over $uk \leftarrow\!\$\ \mathsf{U.Kg}(\lambda)$ for the first one, and over $\overline{uk} \leftarrow\!\$\ \overline{\mathsf{U}}.\mathsf{Kg}(\lambda)$ for the second one. Hence $\mathbf{Coll1}_{\mathsf{U}}(\lambda, m) \le \mathbf{Coll1}_{\overline{\mathsf{U}}}(\lambda, m) + 2^{-\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ as claimed.

```
$\overline{S}^{\textsc{Hash}}(1^\lambda)$
    $lk \leftarrow\$ \mathsf{LT.LKg}(1^\lambda)$ ; $u \leftarrow\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$
    $x \leftarrow \textsc{Hash}(u, 1^{\mathsf{LT.il}(\lambda)})$ ; $uk \leftarrow\$ \mathsf{U.Kg}(1^\lambda)$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$
    $w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x, 1^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
    $mask \leftarrow \textsc{Hash}(w, 1^{\ell(\lambda)})$
    Return $(lk, uk, mask, trap)$

$\overline{D}(1^\lambda, \overline{hk}, L)$
    $(lk, uk, mask, trap) \leftarrow L$ ; $hk \leftarrow (\overline{hk}, uk)$
    $(m_0, m_1, t) \leftarrow\$ A(1^\lambda, (lk, hk))$ ; $b \leftarrow\$ \{0,1\}$
    $c \leftarrow mask \oplus m_b$
    $b' \leftarrow\$ A(1^\lambda, t, (trap, c))$
    If $(b = b')$ then return 1 else return 0
```

```
$B(1^\lambda, K)$
    $u \leftarrow\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ ; $hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(\overline{hk}, uk) \leftarrow hk$
    $(m_0, m_1, t) \leftarrow\$ A(1^\lambda, (K, hk))$ ; $b \leftarrow\$ \{0,1\}$ ; $x \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, u, 1^{\mathsf{LT.il}(\lambda)})$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$ ; $w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x)$ ; $mask \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, w, 1^{\ell(\lambda)})$
    $c \leftarrow mask \oplus m_b$ ; $b' \leftarrow\$ A(1^\lambda, t, (trap, c))$
    If $(b = b')$ return 1 else return 0
```

Figure 15: **Top:** The source $\overline{S}$ and distinguisher $\overline{D}$ in the IND-CPA proof of Theorem 4.2. **Bottom:** Adversary B attacking LT in the IND-CPA proof of Theorem 4.2.

For part (b), consider distinct strings x, x' with $|x| \le m, |x'| \le m'$. By symmetry, there are only three cases.

Case 1: $|x|, |x'| < \overline{\mathsf{U}}.\mathsf{ol}(\lambda)$. Then

$$\Pr[\mathsf{U.Ev}(1^\lambda, uk, x) = \mathsf{U.Ev}(1^\lambda, uk, x')] = \Pr\big[x \,\|\, 10^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)-|x|} = x' \,\|\, 10^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)-|x'|}\big] = 0;$$

the probability is taken over $uk \leftarrow\$ \mathsf{U.Kg}(1^\lambda)$.

Case 2: $|x| \ge \overline{\mathsf{U}}.\mathsf{ol}(\lambda)$ and $|x'| < \overline{\mathsf{U}}.\mathsf{ol}(\lambda)$. Let $x_1 = x[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ and $x_2 = x[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |x|]$. Then

$$\begin{aligned} &\Pr[\mathsf{U.Ev}(1^\lambda, uk, x) = \mathsf{U.Ev}(1^\lambda, uk, x')] \\ \le\ &\Pr\big[\overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, x_2) \oplus (x_1 \times rk) = mk \oplus (x' \,\|\, 10^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)-|x'|})\big] \le 2^{-\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}; \end{aligned}$$

the probability is taken over $uk \leftarrow\$ \mathsf{U.Kg}(1^\lambda)$ for the first one, and over $mk \leftarrow\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ for the second one.

Case 3: $|x|, |x'| \ge \overline{\mathsf{U}}.\mathsf{ol}(\lambda)$. Let $x_1 = x[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)], x_2 = x[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |x|], x'_1 = x'[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ and $x'_2 = x'[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |x'|]$. If $x_1 \ne x'_1$ then

$$\begin{aligned} &\Pr[\mathsf{U.Ev}(1^\lambda, uk, x) = \mathsf{U.Ev}(1^\lambda, uk, x')] \\ \le\ &\Pr[\mathsf{U.Ev}(1^\lambda, uk, x) \oplus \mathsf{U.Ev}(1^\lambda, uk, x') = (x_1 \oplus x'_1) \times rk] \\ \le\ &\frac{1}{2^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)} - 1} \le \frac{2}{2^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}}; \end{aligned}$$

the probability is taken over $uk \leftarrow\$ \mathsf{U.Kg}(1^\lambda)$ for the first one, and over $rk \leftarrow\$ \mathrm{GF}(2^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}) \backslash \{0^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}\}$ for the second one. If $x_1 = x'_1$ then

$$\begin{aligned} &\Pr[\mathsf{U.Ev}(1^\lambda, uk, x) = \mathsf{U.Ev}(1^\lambda, uk, x')] \\ =\ &\Pr[\overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, x_2) = \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, x'_2)] \le \mathbf{Coll2}_{\overline{\mathsf{U}}}(\lambda, m, m'); \end{aligned}$$

the probability is taken over $uk \leftarrow\$ \mathsf{U.Kg}(\lambda)$ for the first one, and over $\overline{uk} \leftarrow\$ \overline{\mathsf{U}}.\mathsf{Kg}(\lambda)$ for the second one. Hence $\mathbf{Coll2}_{\mathsf{U}}(\lambda, m, m') \le \mathbf{Coll2}_{\overline{\mathsf{U}}}(\lambda, m, m') + 2/2^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$.

```
Game $G_1^A(\lambda)$, $\boxed{G_2^A(\lambda)}$
$(ek, dk) \leftarrow\$ \mathsf{LT.EKg}(1^\lambda)$ ; $K \leftarrow ek$
$\boxed{K \leftarrow lk \leftarrow\$ \mathsf{LT.LKg}(1^\lambda) \;;}$ $b \leftarrow\$ \{0,1\}$
$hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(\overline{hk}, uk) \leftarrow hk$ ; $r \leftarrow\$ \{0,1\}^{\mathsf{\overline{U}.ol}(\lambda)}$
$(\overline{uk}, rk, mk) \leftarrow uk$ ; $(m_0, m_1, t) \leftarrow\$ A(1^\lambda, (K, hk))$
$u \leftarrow \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, m_b) \oplus (rk \times r)$
$\boxed{u \leftarrow\$ \{0,1\}^{\mathsf{\overline{U}.ol}(\lambda)} \;;}$ $x \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, u, 1^{\mathsf{LT.il}(\lambda)})$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$ ; $w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x)$
$mask \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, w, 1^{\ell(\lambda)})$ ; $c \leftarrow mask \oplus m_b$
$b' \leftarrow\$ A(1^\lambda, t, (trap, c))$ ; Return $(b = b')$
```

```
Game $\boxed{G_3^A(\lambda)}$, $G_4^A(\lambda)$
$lk \leftarrow\$ \mathsf{LT.LKg}(1^\lambda)$ ; $hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(\overline{hk}, uk) \leftarrow hk$
$u \leftarrow\$ \{0,1\}^{\mathsf{\overline{U}.ol}(\lambda)}$ ; $b \leftarrow\$ \{0,1\}$ ; $x \leftarrow \mathsf{RO}(u, \mathsf{LT.il}(\lambda))$
$(m_0, m_1, t) \leftarrow\$ A(1^\lambda, (lk, hk))$ ; $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$
$w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x)$ ; $mask \leftarrow \mathsf{RO}(w, \ell(\lambda))$
$c \leftarrow mask \oplus m_b$ ; $b' \leftarrow\$ A(1^\lambda, t, (trap, c))$ ; Return $(b = b')$

$\mathsf{RO}(x, s)$
$y \leftarrow\$ \{0,1\}^s$
If $H[x, s] \neq \perp$ then bad $\leftarrow$ true ; $\boxed{y \leftarrow H[x, s]}$
$H[x, s] \leftarrow y$ ; Return $y$
```

Figure 16: **Games G_1–G_4 in the proof of Theorem 4.2.** Games G_2, G_3 contain the corresponding boxed statements, but games G_1, G_4 do not.

D Proof of Theorem 4.2

IND-CPA SECURITY. Let A be an IND-CPA adversary. Consider the source $\overline{S}$ and distinguisher $\overline{D}$ in Fig. 15 that attack the UCE security of $\overline{\mathsf{H}}$. Instead of sampling $r \leftarrow\$ \{0,1\}^{\mathsf{\overline{U}.ol}(\lambda)}$ and computing $u \leftarrow \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, m) \oplus (rk \times r)$, the source directly samples $u \leftarrow\$ \{0,1\}^{\mathsf{\overline{U}.ol}(\lambda)}$, and thus need not know the message m. Moreover, it samples a lossy key lk instead of an injective key ek. We argue that $\overline{S}$ is statistically unpredictable. Consider an arbitrary predictor $\overline{P}$. Consider the following games L_1 and L_2. Game $L_1^{\overline{S},\overline{P}}$ coincides with game $\mathrm{Pred}_{\overline{S}}^{\overline{P}}$. Game L_2 is identical to game L_1, except that the oracle HASH ignores consistency and always returns a fresh random answer. Recall that $\overline{S}$ makes only two queries u and w to HASH, and $\Pr[u = w] \le 2^{-\mathsf{\overline{U}.ol}}$. Hence

$$\Pr[L_1^{\overline{S},\overline{P}}(\cdot)] - \Pr[L_2^{\overline{S},\overline{P}}(\cdot)] \le 2^{-\mathsf{\overline{U}.ol}} \;.$$

We now bound the chance that $\overline{P}$ wins in game L_2. Let q be a polynomial that bounds the size of the output of $\overline{P}$. Assume that $q \le 2^{\mathsf{\overline{U}.ol}-2}$; otherwise the concrete claimed bound is trivial. No information of $u \leftarrow\$ \{0,1\}^{\mathsf{\overline{U}.ol}}$ is given to $\overline{P}$, and thus the chance that $\overline{P}$ can guess u is at most $q/2^{\mathsf{\overline{U}.ol}}$. To bound the chance that $\overline{P}$ can guess w, we shall use a result of Barak et al. [2]. We begin by introducing some definitions. For correlated random variables $X, Z \in \{0,1\}^*$, let

$$\widetilde{\mathbf{H}}_\infty(X \mid Z; q) = \mathbf{E}\Big(\max_{Q' \in (\{0,1\}^*)^q} \{\Pr[X \in Q' \mid Z]\}\Big)$$

be the expected value of the probability that the best (computationally unbounded) adversary can guess X after q attempts, if it's given Z. Lemma D.1 below shows a square-root degradation of $\widetilde{\mathbf{H}}_\infty$ if one applies a universal hash to random variable X.

Lemma D.1 [2] Fix $\lambda \in \mathbb{N}$. Let $X \in \{0,1\}^s$ and $Z \in \{0,1\}^*$ be correlated random variables. Let U be a universal hash function and let $W \leftarrow \mathsf{U.Ev}(1^\lambda, uk, X)$, where $uk \leftarrow\$ \mathsf{U.Kg}(1^\lambda)$. Then

$$\widetilde{\mathbf{H}}_\infty(W \mid (Z, uk); q) \le \frac{q}{2^{\mathsf{U.ol}(\lambda)}} + \sqrt{q \cdot \widetilde{\mathbf{H}}_\infty(X \mid Z; 1) + q \cdot \mathbf{Coll2}_{\mathsf{U}}(\lambda, s)}$$

for any $q \in \mathbb{N}$. ▮

Since LT is τ-lossy, $\widetilde{\mathbf{H}}_\infty(x \mid trap; 1) \le 2^{-\tau}$. Hence from Lemma D.1, the chance that $\overline{P}$ can guess w is at most

$$\sqrt{q \cdot 2^{-\tau} + q \cdot \mathbf{Coll2}_{\mathsf{U}}(\cdot, \mathsf{LT.il})} + \frac{q}{2^{\mathsf{\overline{U}.ol}}} \;.$$

```
$\underline{S^{\text{Hash}}(1^\lambda)}$
$b \leftarrow\!\!\$ \{0,1\}$ ; $lk \leftarrow\!\!\$ \mathsf{LT.LKg}(1^\lambda)$ ; $t \leftarrow\!\!\$ A_2^{\text{LR}}(1^\lambda)$ ; Return $(b, lk, t)$
$\underline{\text{LRSim}(d)}$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\!\!\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $x \leftarrow \text{Hash}(\mathbf{r}[i] \,\|\, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$ ; $c \leftarrow \text{Hash}(x, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$

$\underline{D(1^\lambda, hk, L)}$
$(b, lk, t) \leftarrow L$
$b' \leftarrow\!\!\$ A_2(t, (lk, hk))$
If $(b' = b)$ then return 1
Else return 0
```

```
$\underline{B(1^\lambda, K)}$
$hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow\!\!\$ \{0,1\}$
$t \leftarrow\!\!\$ A_2^{\text{LRSim}}(1^\lambda)$
$b' \leftarrow\!\!\$ A_2(t, (K, hk))$
If $(b = b')$ then return 1
Else return 0

$\underline{\text{LRSim}(d)}$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\!\!\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, \mathbf{r}[i] \,\|\, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$ ; $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$
    $c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$
```

Figure 17: **Top:** Source S and distinguisher D in the IND-CDA proof of Theorem 4.2. **Bottom:** Adversary B attacking LT in the IND-CDA proof of Theorem 4.2.

From Lemma 4.1, we have $\mathbf{Coll2}_{\mathsf{U}}(\cdot, \mathsf{LT.il}) \le \mathbf{Coll2}_{\overline{\mathsf{U}}}(\cdot, \mathsf{LT.il}) + 2/2^{\overline{\mathsf{U}}.\mathsf{ol}}$. Combining the results yields

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},P}(\cdot) \le \frac{\sqrt{q}}{2^{\tau/2}} + \sqrt{q \cdot \mathbf{Coll2}_{\overline{\mathsf{U}}}(\cdot, \mathsf{LT.il})} + \frac{2\sqrt{q}}{2^{\overline{\mathsf{U}}.\mathsf{ol}/2}} \ .$$

What remains is to bound $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE1},A}(\cdot)$ via $\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},\overline{S},\overline{D}}(\cdot)$. Consider adversary B in Fig. 15 that attacks LT. It is given a key K, which might be either an injective key ek or a lossy key lk. It then simulates game $\mathrm{CPA}^A_{\mathsf{HE1}}$, but uses key K instead of ek. Consider the games G_1–G_4 in Fig. 16. Game G_1^A corresponds to game $\mathrm{CPA}^A_{\mathsf{HE1}}$ and in game G_4^A, whatever the adversary receives is independent of the challenge bit b, and thus $\Pr[G_4^A(\cdot)] = 1/2$. Hence

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE1},A}(\cdot) = 2(\Pr[G_1^A(\cdot)] - \Pr[G_4^A(\cdot)]) \ .$$

We describe the game chain up to the terminal one. Game G_2 is identical to game G_1, except for the following changes. First, instead of sampling $r \leftarrow\!\!\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ and computing

$$u \leftarrow \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, m) \oplus (rk \times r),$$

the game directly samples $u \leftarrow\!\!\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$. Since $rk \ne 0^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$, this change makes no difference. Next, instead of sampling an injective key ek, game G_2 samples a lossy key lk. Then

$$\Pr[G_1^A(\cdot)] - \Pr[G_2^A(\cdot)] = \mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) \ .$$

In game G_3, instead of calling $\overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, \cdot, \cdot)$ to compute x and $mask$, we maintain a random oracle and make the corresponding queries to get x and $mask$. Then

$$\Pr[G_2^A(\cdot)] - \Pr[G_3^A(\cdot)] = \mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},\overline{S},\overline{D}}(\cdot) \ .$$

Game G_4 is identical to G_3, except that we always sample x and $seed$ at random. The two games are identical-until-bad, and thus

$$\Pr[G_3^A(\cdot)] - \Pr[G_4^A(\cdot)] \le \Pr[G_4^A(\cdot) \text{ sets } \mathsf{bad}] \le 2^{-\overline{\mathsf{U}}.\mathsf{ol}} \ .$$

Summing up,

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE1},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},\overline{S},\overline{D}}(\cdot) + 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2^{1-\overline{\mathsf{U}}.\mathsf{ol}} \ .$$

```
Game $H_1^A(\lambda)$, $\boxed{H_2^A(\lambda)}$
$hk \leftarrow_{\$} \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow_{\$} \{0,1\}$
$(K, dk) \leftarrow_{\$} \mathsf{LT.EKg}(1^\lambda)$ ; $\boxed{K \leftarrow_{\$} \mathsf{LT.LKg}(1^\lambda)}$
$t \leftarrow_{\$} A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow_{\$} A_2(t, (ek, hk))$
Return $(b = b')$

LR($d$)
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow_{\$} A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, \mathbf{r}[i] \,\|\, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$
    $c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\ell(\lambda)}) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$
```

```
Game $\boxed{H_3^A(\lambda)}$, $H_4^A(\lambda)$
$hk \leftarrow_{\$} \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow_{\$} \{0,1\}$ ; $lk \leftarrow_{\$} \mathsf{LT.LKg}(1^\lambda)$
$t \leftarrow_{\$} A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow_{\$} A_2(t, (ek, hk))$
Return $(b = b')$

LR($d$)
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow_{\$} A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $x \leftarrow \mathsf{RO}(\mathbf{m}_b[i] \,\|\, \mathbf{r}[i], \mathsf{LT.il}(\lambda))$ ; $c \leftarrow \mathsf{RO}(x, \ell(\lambda)) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$

RO($x, \ell$)
$y \leftarrow_{\$} \{0,1\}^\ell$
If $H[x, \ell] \neq \bot$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{y \leftarrow H[x, \ell]}$
$H[x, \ell] \leftarrow y$ ; Return $y$
```

Figure 18: **Games H_1–H_4 in the proof of Theorem 4.2**. Games H_2 and H_3 contain the corresponding boxed statements, but games H_1 and H_4 do not.

IND-CDA SECURITY. Let $A = (A_1, A_2)$ be an adversary attacking IND-CDA security of $\mathsf{HE1}$. First consider the source S and distinguisher D in Fig. 17 attacking H; we'll later translate them to $\overline{S}$ and $\overline{D}$ attacking $\overline{\mathsf{H}}$. They simulate game $\mathrm{CDA}_{\mathsf{HE1}}^A$ but use $\textsc{Hash}(\cdot,\cdot)$ instead of $\mathsf{H.Ev}(1^\lambda, hk, \cdot, \cdot)$ and use a lossy key lk instead of an injective key ek. Consider games H_1–H_4 in Fig. 18. Game H_1^A corresponds to game $\mathrm{CDA}_{\mathsf{HE1}}^A$. We explain the game chain up to the terminal one. Game H_2 is identical to H_2, except that we use a lossy key lk instead of the injective key ek. Consider the adversary B in Fig. 17 attacking LT. It's given a key K, which may be either an injective key ek or a lossy key lk. It then simulates game $\mathrm{CDA}_{\mathsf{HE1}}^A$, but K is used as the encryption key for LT. Then

$$\Pr[H_1^A(\cdot)] - \Pr[H_2^A(\cdot)] = \mathsf{Adv}_{\mathsf{LT},B}^{\mathsf{ltdf}}(\cdot) \ .$$

Game H_3 is identical to game H_2, but we maintain a random oracle RO and calls to $\mathsf{H.Ev}(1^\lambda, hk, \cdot, \cdot)$ are replaced by the corresponding queries to RO. Then

$$\Pr[H_2^A(\cdot)] - \Pr[H_3^A(\cdot)] = \mathsf{Adv}_{\mathsf{H},S,D}^{\mathsf{uce}}(\cdot) \ .$$

In game H_4, we ignore consistency among queries to RO, and always give fresh random answers. The two games H_3 and H_4 are identical-until-bad, and thus

$$\Pr[H_3^A(\cdot)] - \Pr[H_4^A(\cdot)] \le \Pr[H_4^A(\cdot) \text{ sets } \mathsf{bad}] \ .$$

We now bound the chance that game H_4^A sets bad. Let p be the total number of messages that A produces. There are $2p$ queries to RO. Each query is either (i) a uniformly random string of length $\mathsf{LT.il}$, or (ii) a string $\mathbf{r}[i] \,\|\, \mathbf{m}[i]$ produced by A. There are p queries in category (i), and the chance that they trigger bad is at most

$$\sum_{j=p}^{2p-1} \frac{j}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT.il}\}}} \le \frac{1.5p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT.il}\}}} \ .$$

For queries in category (ii), note that A_2 can only specify *distributions*, and within a distribution, it's guaranteed that the strings $(\mathbf{m}[i], \mathbf{r}[i])$ are distinct. Hence the chance these queries trigger bad is at most

$$\sum_{j=p}^{2p-1} j \cdot \mathrm{Guess}_A(\cdot) \le 1.5p^2 \cdot \mathrm{Guess}_A(\cdot) \ .$$

Summing up,

$$\Pr[H_4^A(\cdot) \text{ sets } \mathsf{bad}] \le 1.5p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{1.5p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}}} \ .$$

Finally, in game H_4, whatever A receives is independent of the challenge bit, and thus $\Pr[H_4^A(\cdot)] = 1/2$. Summing up,

$$\mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE1},A}(\cdot) = 2(\Pr[H_1^A(\cdot)] - \Pr[H_4^A(\cdot)]) \le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) + 3p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{3p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}}} \ .$$

Let σ be the total message length produced by LR queries of A. Then S makes $2p$ queries to HASH, and their total length is at most $s = p(\ell + \mathsf{LT}.\mathsf{il} + \overline{\mathsf{U}}.\mathsf{ol})$. We now translate S and D to $\overline{S}$ and $\overline{D}$ attacking $\overline{\mathsf{H}}$ by using a result of BHK2.

Lemma D.2 [9] Let $\overline{\mathsf{H}}$ be a function family of fixed input length, and U be an AU hash function family with $\overline{\mathsf{U}}.\mathsf{ol} = \overline{\mathsf{H}}.\mathsf{il}$ and $\mathsf{F}.\mathsf{IL} = \mathbb{N}$. Let $\mathsf{H} = \mathsf{AU\text{-}then\text{-}Hash}[\mathsf{U}, \overline{\mathsf{H}}]$.

Asymptotic result: If $\overline{\mathsf{H}}$ is $\mathsf{UCE}[\mathcal{S}^{\sup}]$-secure then so is H.

Concrete result: Let S be a source, D a distinguisher, and $\overline{P}$ a predictor. We can construct a source $\overline{S}$, a distinguisher $\overline{D}$, and a predictor P such that

$$\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) \le \mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\cdot, p, \sigma)$$

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le \sqrt{2q\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\cdot, p, \sigma)} + \sqrt{q\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot)}$$

where $p = \mathbf{Q}_S^{\text{HASH}}$, q is the maximum of the size of $\overline{P}$'s output in the execution of $\mathrm{Pred}_{\overline{S}}^{\overline{P}}$, and σ is the maximum of the total length of HASH queries that S makes in $\mathrm{UCE}_{\mathsf{H}}^{S,D}$. Furthermore, $\mathbf{Q}_{\overline{S}}^{\text{HASH}} = \mathbf{Q}_S^{\text{HASH}}$, $\mathbf{T}(\mathrm{UCE}_{\overline{\mathsf{H}}}^{\overline{S},\overline{D}}) \le \mathbf{T}(\mathrm{UCE}_{\mathsf{H}}^{S,D})$, and P outputs a set of size at most $\mathbf{Q}_S^{\text{HASH}}$. ▮

Let $\overline{P}$ be a predictor, and let q be a polynomial that bounds the size of the output of $\overline{P}$. From Lemma D.2, there are source $\overline{S}$, distinguisher $\overline{D}$, and predictor P such that

$$\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) \le \mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\cdot, 2p, s)$$

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le \sqrt{2q\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\cdot, 2p, s)} + \sqrt{q\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot)}$$

where the output of P contains at most $2p$ elements, $\mathbf{Q}_{\overline{S}}^{\text{HASH}} = \mathbf{Q}_S^{\text{HASH}}$, and $\mathbf{T}(\mathrm{UCE}_{\overline{\mathsf{H}}}^{\overline{S},\overline{D}}) \le \mathbf{T}(\mathrm{UCE}_{\mathsf{H}}^{S,D}) \le \mathbf{T}(\mathrm{CDA}_{\mathsf{HE1}}^{A}) + \mathbf{T}(\mathsf{LT.LKg})$. Moreover, from Lemma 4.1,

$$\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\cdot, 2p, s) \le \mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 2p, s) + \frac{8p^2}{2^{\overline{\mathsf{U}}.\mathsf{ol}}} \ .$$

Hence

$$\mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE1},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + 2\mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 2p, s) + 3p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{19p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}}} \ .$$

What's left is to bound $\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot)$. For $i \in \{3, 4\}$, let game J_i be identical to game H_i, except for the following change. Instead of returning $(b = b')$, we'll run $Q' \leftarrow\$ P(1^\lambda, (b, lk, t))$ and then return

```
$\underline{\overline{S}^{\text{HASH}}(1^\lambda)}$
$lk \leftarrow\!\!\$\ \mathsf{LT.LKg}(1^\lambda)$ ; $u \leftarrow\!\!\$\ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$
$x \leftarrow \text{HASH}(u, 1^{\mathsf{LT.il}(\lambda)})$ ; $uk \leftarrow\!\!\$\ \mathsf{U.Kg}(1^\lambda)$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$
$w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x, 1^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
$seed \leftarrow \text{HASH}(w, 1^{\mathsf{F.kl}(\lambda)+\overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
Return $(lk, uk, seed, trap)$
```

```
$\underline{\overline{D}(1^\lambda, \overline{hk}, L)}$
$(lk, uk, seed, trap) \leftarrow L$ ; $hk \leftarrow (\overline{hk}, uk)$
$y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |seed|]$
$(m_0, m_1, t) \leftarrow\!\!\$\ A(1^\lambda, (lk, hk))$ ; $b \leftarrow\!\!\$\ \{0,1\}$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow\!\!\$\ A(1^\lambda, t, (trap, c))$
If $(b = b')$ then return 1 else return 0
```

```
$\underline{B(1^\lambda, K)}$
$u \leftarrow\!\!\$\ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ ; $hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(\overline{hk}, uk) \leftarrow hk$
$(m_0, m_1, t) \leftarrow\!\!\$\ A(1^\lambda, (K, hk))$
$b \leftarrow\!\!\$\ \{0,1\}$ ; $x \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, u, 1^{\mathsf{LT.il}(\lambda)})$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$ ; $w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x)$
$seed \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, w, 1^{\mathsf{F.kl}(\lambda)+\overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
$y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow\!\!\$\ A(1^\lambda, t, (trap, c))$
If $(b = b')$ return 1 else return 0
```

```
$\underline{C^{\text{RR}}(1^\lambda)}$
$lk \leftarrow\!\!\$\ \mathsf{LT.LKg}(1^\lambda)$ ; $u \leftarrow\!\!\$\ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$
$hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(\overline{hk}, uk) \leftarrow hk$
$(m_0, m_1, t) \leftarrow\!\!\$\ A(1^\lambda, (lk, hk))$
$b \leftarrow\!\!\$\ \{0,1\}$ ; $x \leftarrow\!\!\$\ \{0,1\}^{\mathsf{LT.il}(\lambda)}$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$
$w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x)$
$y \leftarrow\!\!\$\ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ ; $mask \leftarrow \text{RR}(0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow\!\!\$\ A(1^\lambda, t, (trap, c))$
If $(b = b')$ return 1 else return 0
```

Figure 19: **Top:** Source $\overline{S}$ and distinguisher $\overline{D}$ in the IND-CPA proof of Theorem 4.3. **Bottom:** Adversaries B and C attacking LT and F respectively, in the IND-CPA proof of Theorem 4.3.

$(Q \cap Q' \neq \emptyset)$, where Q is the set of the strings $\mathbf{r}[i] \,\|\, \mathbf{m}_b[i], x$, and y specified in procedure LR. Game $J_3^{A,P}$ corresponds to game Pred_S^P, and

$$\Pr[J_3^{A,P}(\cdot)] - \Pr[J_4^{A,P}(\cdot)] \le \Pr[J_4^{A,P}(\cdot) \text{ sets } \mathsf{bad}] \le 1.5p^2 \cdot \text{Guess}_A(\cdot) + \frac{1.5p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT.il}\}}} \ .$$

In game J_4, the predictor is given only $\mathsf{LT.Ev}(1^\lambda, lk, x)$ and has to guess one of $2p$ strings $x, \mathbf{r}[i] \,\|\, \mathbf{m}_b[i]$ within $2p$ attempts. Then

$$\Pr[J_4^{A,P}(\cdot)] \le 2p^2 \cdot \text{Guess}_A(\cdot) + \frac{2p^2}{2^\tau} \ .$$

By summing up and observing that $\tau \le \mathsf{LT.il}$, we have

$$\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot) \le 3.5p^2 \cdot \text{Guess}_A(\cdot) + \frac{3.5p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \tau\}}} \ .$$

Combining the results yields

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le 2p\sqrt{q \cdot \text{Guess}_A} + \sqrt{2q\mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 2p, s)} + \frac{6p\sqrt{q}}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \tau\}/2}} \ .$$

E Proof of Theorem 4.3

IND-CPA SECURITY. Let A be an IND-CPA adversary. Consider the source $\overline{S}$ and distinguisher $\overline{D}$ in Fig. 19 that attack the UCE security of $\overline{\mathsf{H}}$. Instead of sampling $r \leftarrow\!\!\$\ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ and computing $u \leftarrow \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, m) \oplus (rk \times r)$, the source directly samples $u \leftarrow\!\!\$\ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$, and thus need not know the message m. Moreover, it samples a lossy key lk instead of an injective key ek. The computation of $\mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m|})$ and $\mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m|})$ is left to the distinguisher. We argue that $\overline{S}$ is statistically unpredictable. Consider an arbitrary predictor $\overline{P}$. Consider the following games L_1 and L_2. Game $L_1^{\overline{S},\overline{P}}$ coincides with game $\text{Pred}_{\overline{S}}^{\overline{P}}$. Game L_2 is identical to game L_1, except that the oracle HASH ignores

```
Game $G_1^A(\lambda)$, $\boxed{G_2^A(\lambda)}$
$(ek, dk) \leftarrow_\$ \mathsf{LT.EKg}(1^\lambda)$ ; $K \leftarrow ek$
$\boxed{K \leftarrow lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)\,;}$ $b \leftarrow_\$ \{0,1\}$
$hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(\overline{hk}, uk) \leftarrow hk$ ; $r \leftarrow_\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$
$(\overline{uk}, rk, mk) \leftarrow uk$ ; $(m_0, m_1, t) \leftarrow_\$ A(1^\lambda, (K, hk))$
$u \leftarrow \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, m_b) \oplus (rk \times r)$
$\boxed{u \leftarrow_\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}\,;}$ $x \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, u, 1^{\mathsf{LT.il}(\lambda)})$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$ ; $w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x)$
$seed \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, w, 1^{\mathsf{F.kl}(\lambda) + \overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
$y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda) + 1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow_\$ A(1^\lambda, t, (trap, c))$ ; Return $(b = b')$
```

```
Game $\boxed{G_3^A(\lambda)}$, $G_4^A(\lambda)$
$lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)$ ; $hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(\overline{hk}, uk) \leftarrow hk$
$u \leftarrow_\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ ; $b \leftarrow_\$ \{0,1\}$ ; $x \leftarrow \mathsf{RO}(u, \mathsf{LT.il}(\lambda))$
$(m_0, m_1, t) \leftarrow_\$ A(1^\lambda, (lk, hk))$ ; $w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x)$
$seed \leftarrow_\$ \mathsf{RO}(w, \mathsf{F.kl}(\lambda) + \overline{\mathsf{U}}.\mathsf{ol}(\lambda))$
$y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda) + 1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow_\$ A(1^\lambda, t, (trap, c))$ ; Return $(b = b')$

$\mathsf{RO}(x, \ell)$
$y \leftarrow_\$ \{0,1\}^\ell$
If $H[x, \ell] \neq \bot$ then bad $\leftarrow$ true ; $\boxed{y \leftarrow H[x, \ell]}$
$H[x, \ell] \leftarrow y$ ; Return $y$
```

```
Game $G_5^A(\lambda)$, $\boxed{G_6^A(\lambda)}$
$lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)$ ; $hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(\overline{hk}, uk) \leftarrow hk$ ; $u \leftarrow_\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ ; $b \leftarrow_\$ \{0,1\}$
$(m_0, m_1, t) \leftarrow_\$ A(1^\lambda, (lk, hk))$ ; $x \leftarrow_\$ \{0,1\}^{\mathsf{LT.il}(\lambda)}$ ; $seed \leftarrow_\$ \{0,1\}^{\mathsf{F.kl}(\lambda) + \overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$
$w \leftarrow \mathsf{U.Ev}(1^\lambda, uk, x)$ ; $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$
$y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda) + 1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$ ; $\boxed{mask \leftarrow_\$ \{0,1\}^{|m_b|}}$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$ ; $b' \leftarrow_\$ A(1^\lambda, t, (trap, c))$ ; Return $(b = b')$
```

Figure 20: **Games G_1–G_6 in the IND-CPA proof of Theorem 4.3.** Games G_2, G_3, G_6 contain the corresponding boxed statements, but games G_1, G_4, G_5 do not.

consistency and always return a fresh random answer. Recall that $\overline{S}$ makes only two queries u and w to Hash, and $\Pr[u = w] \leq 2^{-\overline{\mathsf{U}}.\mathsf{ol}}$. Hence

$$\Pr[L_1^{\overline{S},\overline{P}}(\cdot)] - \Pr[L_2^{\overline{S},\overline{P}}(\cdot)] \leq 2^{-\overline{\mathsf{U}}.\mathsf{ol}} \ .$$

We now bound the chance that $\overline{P}$ wins in game L_2. Let q be a polynomial that bounds the size of the output of $\overline{P}$. Assume that $q \leq 2^{\overline{\mathsf{U}}.\mathsf{ol}-2}$; otherwise the concrete claimed bound is trivial. No information of $u \leftarrow_\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}}$ is given to the predictor, and thus the chance that the predictor can guess u is at most $q/2^{\overline{\mathsf{U}}.\mathsf{ol}}$. Since LT is τ-lossy, from Lemma D.1, the chance that $\overline{P}$ can guess w is at most

$$\sqrt{q \cdot 2^{-\tau} + q \cdot \mathbf{Coll2}_{\mathsf{U}}(\cdot, \mathsf{LT.il})} + \frac{q}{2^{\overline{\mathsf{U}}.\mathsf{ol}}} \ .$$

By Lemma 4.1, we have $\mathbf{Coll2}_{\mathsf{U}}(\cdot, \mathsf{LT.il}) \leq \mathbf{Coll2}_{\overline{\mathsf{U}}}(\cdot, \mathsf{LT.il}) + 2/2^{\overline{\mathsf{U}}.\mathsf{ol}}$. Combining the results yields

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \leq \frac{\sqrt{q}}{2^{\tau/2}} + \sqrt{q \cdot \mathbf{Coll2}_{\overline{\mathsf{U}}}(\cdot, \mathsf{LT.il})} + \frac{2\sqrt{q}}{2^{\overline{\mathsf{U}}.\mathsf{ol}/2}} \ .$$

What remains is to bound $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE2},A}(\cdot)$ via $\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},\overline{S},\overline{D}}(\cdot)$. Consider the adversary B attacking LT and adversary C attacking F in Fig. 19. Adversary B is given a key K, which might be either an injective key ek or a lossy key lk. It then simulates game $\mathrm{CPA}^A_{\mathsf{HE2}}$, but uses key K instead of ek. Adversary C simulates game $\mathrm{UCE}^{\overline{S},\overline{D}}_{\mathsf{H}}$ but the strings x and y are always chosen at random and $\mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$ is replaced by $\mathrm{RR}(0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$. Consider the games G_1–G_6 in Fig. 20. Game G_1^A corresponds to game $\mathrm{CPA}^A_{\mathsf{HE2}}$ and in game G_6^A, whatever the adversary receives is independent of the challenge bit b, and thus $\Pr[G_6^A(\cdot)] = 1/2$. Hence

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE2},A}(\cdot) = 2(\Pr[G_1^A(\cdot)] - \Pr[G_6^A(\cdot)]) \ .$$

```
$\underline{S^{\textsc{Hash}}(1^\lambda)}$
$b \leftarrow\$ \{0,1\}$ ; $lk \leftarrow\$ \mathsf{LT.LKg}(1^\lambda)$ ; $t \leftarrow\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; Return $(b, lk, t)$
$\underline{\textsc{LRSim}(d)}$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
   $x \leftarrow \textsc{Hash}(\mathbf{r}[i] \,\|\, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$ ; $seed \leftarrow \textsc{Hash}(x, 1^{\mathsf{F.kl}(\lambda)+\overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
   $y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |seed|]$
   $c \leftarrow \textsc{Hash}(y, 1^{|\mathbf{m}_b[i]|}) \oplus \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
   $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$
```

```
$\underline{D(1^\lambda, hk, L)}$
$(b, lk, t) \leftarrow L$
$b' \leftarrow\$ A_2(t, (lk, hk))$
If $(b' = b)$ then return 1
Else return 0
```

```
$\underline{B(1^\lambda, K)}$
$hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow\$ \{0,1\}$
$t \leftarrow\$ A_2^{\textsc{LRSim}}(1^\lambda)$
$b' \leftarrow\$ A_2(t, (K, hk))$
If $(b = b')$ then return 1
Else return 0
```

```
$\underline{\textsc{LRSim}(d)}$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
   $x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, \mathbf{r}[i] \,\|\, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$
   $seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda)+\overline{\mathsf{U}}.\mathsf{ol}(\lambda)})$
   $y \leftarrow seed[1, \overline{\mathsf{U}}.\mathsf{ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}.\mathsf{ol}(\lambda)+1, |seed|]$
   $c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|\mathbf{m}_b[i]|}) \oplus \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
   $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$
```

Figure 21: **Top:** Source S and distinguisher D in the IND-CDA proof of Theorem 4.3. **Bottom:** Adversary B attacking LT in the IND-CDA proof of Theorem 4.3.

We describe the game chain up to the terminal one. Game G_2 is identical to game G_1, except for the following changes. First, instead of sampling $r \leftarrow\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$ and computing

$$u \leftarrow \overline{\mathsf{U}}.\mathsf{Ev}(1^\lambda, \overline{uk}, m) \oplus (rk \times r),$$

the game directly samples $u \leftarrow\$ \{0,1\}^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$. Since $rk \neq 0^{\overline{\mathsf{U}}.\mathsf{ol}(\lambda)}$, this change makes no difference. Next, instead of sampling an injective key ek, game G_2 samples a lossy key lk. Then

$$\Pr[G_1^A(\cdot)] - \Pr[G_2^A(\cdot)] = \mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) \ .$$

In game G_3, instead of calling $\overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, \overline{hk}, \cdot, \cdot)$ to compute x and $seed$, we maintain a random oracle and make the corresponding queries to get x and $seed$. Then

$$\Pr[G_2^A(\cdot)] - \Pr[G_3^A(\cdot)] = \mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) \ .$$

Game G_4 is identical to G_3, except that we always sample x and $seed$ at random. The two games are identical-until-bad, and thus

$$\Pr[G_3^A(\cdot)] - \Pr[G_4^A(\cdot)] \leq \Pr[G_4^A(\cdot) \text{ sets } \mathsf{bad}] \leq 2^{-\overline{\mathsf{U}}.\mathsf{ol}} \ .$$

Game G_5 is a simplified version of game G_4. In game G_6, instead of using $\mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$, we use a uniformly random string. Hence

$$\Pr[G_5^A(\cdot)] - \Pr[G_6^A(\cdot)] = \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},C}(\cdot) \ .$$

Summing up,

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE2},A}(\cdot) \leq 2\mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},C}(\cdot) + 2^{1-\overline{\mathsf{U}}.\mathsf{ol}} \ .$$

IND-CDA SECURITY. Let $A = (A_1, A_2)$ be an adversary attacking IND-CDA security of $\mathsf{HE2}$. First consider the source S and distinguisher D in Fig. 21 attacking H; we'll later translate them to $\overline{S}$ and $\overline{D}$ attacking $\overline{\mathsf{H}}$. They simulate game $\mathrm{CDA}^A_{\mathsf{HE2}}$ but use $\textsc{Hash}(\cdot, \cdot)$ instead of $\mathsf{H.Ev}(1^\lambda, hk, \cdot, \cdot)$ and use a lossy

```
Game $H_1^A(\lambda)$, $\boxed{H_2^A(\lambda)}$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow_\$ \{0,1\}$
$(K, dk) \leftarrow_\$ \mathsf{LT.EKg}(1^\lambda)$ ; $\boxed{K \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)}$
$t \leftarrow_\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow_\$ A_2(t, (ek, hk))$
Return $(b = b')$

LR$(d)$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow_\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, \mathbf{r}[i] \,\|\, \mathbf{m}_b[i], 1^{\mathsf{LT.il}(\lambda)})$
    $seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda) + \overline{\mathsf{U}}\mathsf{.ol}(\lambda)})$
    $\ell \leftarrow |\mathbf{m}_b[i]|$ ; $y \leftarrow seed[1, \overline{\mathsf{U}}\mathsf{.ol}(\lambda)]$
    $fk \leftarrow seed[\overline{\mathsf{U}}\mathsf{.ol}(\lambda) + 1, |seed|]$
    $mask \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^\ell)$
    $c \leftarrow mask \oplus \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^\ell) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$
```

```
Game $\boxed{H_3^A(\lambda)}$, $H_4^A(\lambda)$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow_\$ \{0,1\}$ ; $lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)$
$t \leftarrow_\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow_\$ A_2(t, (ek, hk))$
Return $(b = b')$

LR$(d)$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow_\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $x \leftarrow \mathsf{RO}(\mathbf{m}_b[i] \,\|\, \mathbf{r}[i], \mathsf{LT.il}(\lambda))$
    $seed \leftarrow \mathsf{RO}(x, \mathsf{F.kl}(\lambda) + \overline{\mathsf{U}}\mathsf{.ol}(\lambda))$
    $y \leftarrow seed[1, \overline{\mathsf{U}}\mathsf{.ol}(\lambda)]$ ; $fk \leftarrow seed[\overline{\mathsf{U}}\mathsf{.ol}(\lambda) + 1, |seed|]$
    $c \leftarrow \mathsf{RO}(y, |\mathbf{m}_b[i]|) \oplus \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$

$\mathsf{RO}(x, \ell)$
$y \leftarrow_\$ \{0,1\}^\ell$
If $H[x, \ell] \neq \bot$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{y \leftarrow H[x, \ell]}$
$H[x, \ell] \leftarrow y$ ; Return $y$
```

Figure 22: **Games H_1–H_4 in the proof of Theorem 4.3**. Games H_2 and H_3 contain the corresponding boxed statements, but games H_1 and H_4 do not.

key lk instead of an injective key ek. Consider games H_1–H_4 in Fig. 22. Game H_1^A corresponds to game $\mathrm{CDA}_{\mathsf{HE2}}^A$. We explain the game chain up to the terminal one. Game H_2 is identical to H_2, except that we use a lossy key lk instead of the injective key ek. Consider the adversary B in Fig. 21 attacking LT. It's given a key K, which may be either an injective key ek or a lossy key lk. It then simulates game $\mathrm{CDA}_{\mathsf{HE2}}^A$, but K is used as the encryption key for LT. Then

$$\Pr[H_1^A(\cdot)] - \Pr[H_2^A(\cdot)] = \mathsf{Adv}_{\mathsf{LT},B}^{\mathsf{ltdf}}(\cdot) \ .$$

Game H_3 is identical to game H_2, but we maintain a random oracle RO and calls to $\mathsf{H.Ev}(1^\lambda, hk, \cdot, \cdot)$ are replaced by the corresponding queries to RO. Then

$$\Pr[H_2^A(\cdot)] - \Pr[H_3^A(\cdot)] = \mathsf{Adv}_{\mathsf{H},S,D}^{\mathsf{uce}}(\cdot) \ .$$

In game H_4, we ignore consistency among queries to RO, and always give fresh random answers. The two games H_3 and H_4 are identical-until-bad, and thus

$$\Pr[H_3^A(\cdot)] - \Pr[H_4^A(\cdot)] \le \Pr[H_4^A(\cdot) \text{ sets } \mathsf{bad}] \ .$$

We now bound the chance that game H_4^A sets bad. Let p be the total number of messages that A produces. There are $3p$ queries to RO. Each query is either (i) a uniformly random string of length at least $\min\{\overline{\mathsf{U}}\mathsf{.ol}, \mathsf{LT.il}\}$, or (ii) a string $\mathbf{r}[i] \,\|\, \mathbf{m}[i]$ produced by A. There are $2p$ queries in category (i), and the chance that they trigger bad is at most

$$\sum_{j=p}^{3p-1} \frac{j}{2^{\min\{\overline{\mathsf{U}}\mathsf{.ol}, \mathsf{LT.il}\}}} \le \frac{4p^2}{2^{\min\{\overline{\mathsf{U}}\mathsf{.ol}, \mathsf{LT.il}\}}} \ .$$

For queries in category (ii), note that A_2 can only specify *distributions*, and within a distribution, it's guaranteed that the strings $(\mathbf{m}[i], \mathbf{r}[i])$ are distinct. Hence the chance these queries trigger bad is at most

$$\sum_{j=2p}^{3p-1} j \cdot \mathrm{Guess}_A(\cdot) \le 2.5p^2 \cdot \mathrm{Guess}_A(\cdot) \ .$$

Summing up,

$$\Pr[H_4^A(\cdot) \text{ sets } \mathsf{bad}] \le 2.5p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{4p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}}} \ .$$

Finally, in game H_4, whatever A receives is independent of the challenge bit, and thus $\Pr[H_4^A(\cdot)] = 1/2$. Summing up,

$$\mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE2},A}(\cdot) = 2(\Pr[H_1^A(\cdot)] - \Pr[H_4^A(\cdot)]) \le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) + 5p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{8p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}}} \ .$$

Let σ be the total message length produced by LR queries of A. Then S makes $3p$ queries to $\textsc{Hash}$, and their total length is at most $s = \sigma + 3p \max\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}$. We now translate S and D to $\overline{S}$ and $\overline{D}$ attacking $\overline{\mathsf{H}}$. Let $\overline{P}$ be a predictor and let q be a polynomial that bounds the size of the output of $\overline{P}$. From Lemma D.2, there are source $\overline{S}$, distinguisher $\overline{D}$, and predictor P such that

$$\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) \le \mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\cdot, 3p, s)$$

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le \sqrt{2q\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\cdot, 3p, s)} + \sqrt{q\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot)}$$

where the output of P contains at most $3p$ elements, $\mathbf{Q}^{\textsc{Hash}}_{\overline{S}} = \mathbf{Q}^{\textsc{Hash}}_{S}$, and $\mathbf{T}(\mathrm{UCE}^{\overline{S},\overline{D}}_{\overline{\mathsf{H}}}) \le \mathbf{T}(\mathrm{UCE}^{S,D}_{\mathsf{H}}) \le \mathbf{T}(\mathrm{CDA}^{A}_{\mathsf{HE2}}) + \mathbf{T}(\mathsf{LT}.\mathsf{LKg})$. Moreover, from Lemma 4.1,

$$\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{U}}(\cdot, 3p, s) \le \mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 3p, s) + \frac{18p^2}{2^{\overline{\mathsf{U}}.\mathsf{ol}}} \ .$$

Hence

$$\mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE2},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\cdot) + 2\mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 3p, s) + 5p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{44p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}}} \ .$$

What's left is to bound $\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot)$. For $i \in \{3, 4\}$, let game J_i be identical to game H_i, except for the following change. Instead of returning $(b = b')$, we'll run $Q' \leftarrow\!\!\$\ P(1^\lambda, (b, lk, t))$ and then return $(Q \cap Q' \neq \emptyset)$, where Q is the set of the strings $\mathbf{r}[i] \,\|\, \mathbf{m}_b[i], x$, and y specified in procedure LR. Game $J_3^{A,P}$ corresponds to game Pred^P_S, and

$$\Pr[J_3^{A,P}(\cdot)] - \Pr[J_4^{A,P}(\cdot)] \le \Pr[J_4^{A,P}(\cdot) \text{ sets } \mathsf{bad}] \le 2.5p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{4p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \mathsf{LT}.\mathsf{il}\}}} \ .$$

In game J_4, the predictor is given only $\mathsf{LT}.\mathsf{Ev}(1^\lambda, lk, x)$ and has to guess one of $3p$ strings $x, \mathbf{r}[i] \,\|\, \mathbf{m}_b[i], y$ within $3p$ attempts. Then

$$\Pr[J_4^{A,P}(\cdot)] \le 3p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{3p^2}{2^\tau} + \frac{3p^2}{2^{\overline{\mathsf{U}}.\mathsf{ol}}} \ .$$

By summing up and observing that $\tau \le \mathsf{LT}.\mathsf{il}$, we have

$$\mathsf{Adv}^{\mathsf{pred}}_{S,P}(\cdot) \le 5.5p^2 \cdot \mathrm{Guess}_A(\cdot) + \frac{10p^2}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \tau\}}} \ .$$

Combining the results yields

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\cdot) \le \sqrt{2q\mathsf{Adv}^{\mathsf{coll}}_{\overline{\mathsf{U}}}(\cdot, 3p, s)} + 2.5p\sqrt{q \cdot \mathrm{Guess}_A(\cdot)} + \frac{9.5p\sqrt{q}}{2^{\min\{\overline{\mathsf{U}}.\mathsf{ol}, \tau\}/2}} \ .$$

```
$\underline{S^{\text{Hash}}(1^\lambda)}$
$w \leftarrow_\$ \{0,1\}^{\rho(\lambda)}$ ; $lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)$
$x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, w, 1^{\mathsf{LT.il}(\lambda)})$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$
$seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
Return $(trap, seed)$
```

```
$\underline{D(1^\lambda, hk, L)}$
$(trap, seed) \leftarrow L$ ; $y \leftarrow seed[1,\lambda]$ ; $fk \leftarrow seed[\lambda+1, |seed|]$
$(m_0, m_1, t) \leftarrow_\$ A(1^\lambda, (lk, hk))$ ; $b \leftarrow_\$ \{0,1\}$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow_\$ A(1^\lambda, t, (trap, c))$
If $(b = b')$ then return 1 else return 0
```

```
$\underline{B(1^\lambda, K)}$
$w \leftarrow_\$ \{0,1\}^{\rho(\lambda)}$ ; $hk \leftarrow \mathsf{H.Kg}(1^\lambda)$
$(m_0, m_1, t) \leftarrow_\$ A(1^\lambda, (K, hk))$
$b \leftarrow_\$ \{0,1\}$ ; $x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, w, 1^{\mathsf{LT.il}(\lambda)})$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$
$seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, w, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
$y \leftarrow seed[1,\lambda]$ ; $fk \leftarrow seed[\lambda+1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow_\$ A(1^\lambda, t, (trap, c))$
If $(b = b')$ return 1 else return 0
```

```
$\underline{C^{\mathrm{RR}}(1^\lambda)}$
$lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)$ ; $w \leftarrow_\$ \{0,1\}^{\rho(\lambda)}$
$hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $(m_0, m_1, t) \leftarrow_\$ A(1^\lambda, (lk, hk))$
$b \leftarrow_\$ \{0,1\}$ ; $x \leftarrow_\$ \{0,1\}^{\mathsf{LT.il}(\lambda)}$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$ ; $y \leftarrow_\$ \{0,1\}^\lambda$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus \mathrm{RR}(0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|}) \oplus m_b$
$b' \leftarrow_\$ A(1^\lambda, t, (trap, c))$
If $(b = b')$ return 1 else return 0
```

Figure 23: **Top:** Source S and distinguisher D in the IND-CPA proof of Theorem 4.5. **Bottom:** Adversaries B and C attacking LT and F respectively, in the IND-CPA proof of Theorem 4.5.

F Proof of Lemma 4.4

Assume that $q \le 2^{\ell/3}$; otherwise the claim is trivial. For each string $v \in \{0,1\}^\ell$, we say that it's a *critical mass* if $\Pr[V = v] \ge 2^{-\ell/3}$. Let Z be the set of all critical masses and let $T = \{ z_1 \oplus z_2 : z_1, z_2 \in Z \}$. Then $|Z| \le 2^{\ell/3}$, and thus $|T| \le 2^{2\ell/3}$. Suppose that A makes q adaptive queries $(x_1, \ell_1), \ldots, (x_q, \ell_q)$ to RO. Let $Q = \{x_1, \ldots, x_q\}$. First, if none of x_i hits U then $\mathsf{RO}(U, \ell)$ is independent of Q, and thus the chance that $W \in Q$ is at most $q/2^\ell \le 0.5q \cdot 2^{-\ell/3}$. What's left is to bound the chance that both U and W are in Q. This only happens if there are some $i, j \le q$ such that $|x_i| = \ell, \ell_j = \ell$, and $(x_j, \mathsf{RO}(x_j, \ell) \oplus x_i) = (U, V)$.

First consider the pairs $(x_j, \mathsf{RO}(x_j, \ell) \oplus x_i)$, for all $i \le j \le q$ such that $|x_i| = \ell$ and $\ell_j = \ell$. Since x_i was created before the adversary queries (x_j, ℓ) to RO, the adaptivity of A doesn't help: the string $\mathsf{RO}(x_j, \ell) \oplus x_i$ is a uniformly random string, and thus the chance that these pairs hit (U, V) is at most $0.5q^2 \cdot 2^{-\ell} \le 0.5q \cdot 2^{-\ell/3}$. Next, consider the pairs $(x_j, \mathsf{RO}(x_j, \ell) \oplus x_i)$ for all $j < i \le q$ such that $|x_i| = \ell$ and $\ell_j = \ell$. This time A's adaptivity does help, because x_i is created *after* A sees $\mathsf{RO}(x_j, \ell)$. Let Bad be the event that there are more than q critical masses among those strings $\mathsf{RO}(x_j, \ell) \oplus x_i$. We claim that

$$\Pr[\mathsf{Bad}] \le 0.5q(q-1) \cdot 2^{-\ell/3} \ .$$

If Bad doesn't happen, the chance that the pairs above hit (U, V) is at most $q\epsilon + 0.5q(q-1) \cdot 2^{-\ell/3}$. To justify our claim, if Bad occurs then by Pigeonhole Principle, there must be $j < k < i \le q$ such that both $\mathsf{RO}(x_j, \ell) \oplus x_i$ and $\mathsf{RO}(x_k, \ell) \oplus x_i$ are critical masses, and thus $\mathsf{RO}(x_j, \ell) \oplus \mathsf{RO}(x_k, \ell)$ must be in T. On the other hand, the chance that there are $j < k \le q$ such that $\mathsf{RO}(x_j, \ell) \oplus \mathsf{RO}(x_k, \ell) \in T$ is at most

$$\frac{q(q-1)}{2} \cdot \frac{|T|}{2^\ell} \le 0.5q(q-1) \cdot 2^{-\ell/3} \ .$$

Summing up, the chance that both W and U are in Q is at most $q\epsilon + (q^2 - 0.5q) \cdot 2^{-\ell/3}$, establishing the result of the lemma.

```
Game $G_1^A(\lambda)$, $\boxed{G_2^A(\lambda)}$
$(ek, dk) \leftarrow_{\$} \mathsf{LT.EKg}(1^\lambda)$ ; $K \leftarrow ek$
$\boxed{K \leftarrow lk \leftarrow_{\$} \mathsf{LT.LKg}(1^\lambda) ;}$ $b \leftarrow_{\$} \{0,1\}$
$hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $r \leftarrow_{\$} \{0,1\}^{\rho(\lambda)}$
$(m_0, m_1, t) \leftarrow_{\$} A(1^\lambda, (K, hk))$
$w \leftarrow \mathsf{H.Ev}(1^\lambda, hk, m_b, 1^{|r|}) \oplus |r|$ ; $\boxed{w \leftarrow_{\$} \{0,1\}^{\rho(\lambda)}}$
$x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, w, 1^{\mathsf{LT.il}(\lambda)})$ ; $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$
$seed \leftarrow \overline{\mathsf{H}}\mathsf{.Ev}(1^\lambda, \overline{hk}, w, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
$y \leftarrow seed[1, \lambda]$ ; $fk \leftarrow seed[\lambda + 1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow_{\$} A(1^\lambda, t, (trap, c))$ ; Return $(b = b')$
```

```
Game $\boxed{G_3^A(\lambda)}$, $G_4^A(\lambda)$
$lk \leftarrow_{\$} \mathsf{LT.LKg}(1^\lambda)$ ; $hk \leftarrow \mathsf{H.Kg}(1^\lambda)$
$w \leftarrow_{\$} \{0,1\}^{\rho(\lambda)}$ ; $b \leftarrow_{\$} \{0,1\}$
$(m_0, m_1, t) \leftarrow_{\$} A(1^\lambda, (lk, hk))$
$x \leftarrow \mathsf{RO}(w, \mathsf{LT.il}(\lambda))$ ; $seed \leftarrow \mathsf{RO}(x, \mathsf{F.kl}(\lambda) + \lambda)$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$
$y \leftarrow seed[1, \lambda]$ ; $fk \leftarrow seed[\lambda + 1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$
$b' \leftarrow_{\$} A(1^\lambda, t, (trap, c))$ ; Return $(b = b')$

$\mathsf{RO}(x, \ell)$
$y \leftarrow_{\$} \{0,1\}^\ell$
If $H[x, \ell] \neq \bot$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{y \leftarrow H[x, \ell]}$
$H[x, \ell] \leftarrow y$ ; Return $y$
```

```
Game $G_5^A(\lambda)$, $\boxed{G_6^A(\lambda)}$
$lk \leftarrow_{\$} \mathsf{LT.LKg}(1^\lambda)$ ; $hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $w \leftarrow_{\$} \{0,1\}^{\rho\mathsf{U}(\lambda)}$ ; $b \leftarrow_{\$} \{0,1\}$
$(m_0, m_1, t) \leftarrow_{\$} A(1^\lambda, (lk, hk))$ ; $x \leftarrow_{\$} \{0,1\}^{\mathsf{LT.il}(\lambda)}$ ; $seed \leftarrow_{\$} \{0,1\}^{\mathsf{F.kl}(\lambda)+\lambda}$
$trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$ ; $y \leftarrow seed[1, \lambda]$ ; $fk \leftarrow seed[\lambda + 1, |seed|]$
$mask \leftarrow \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$ ; $\boxed{mask \leftarrow_{\$} \{0,1\}^{|m_b|}}$
$c \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m_b|}) \oplus mask \oplus m_b$ ; $b' \leftarrow_{\$} A(1^\lambda, t, (trap, c))$ ; Return $(b = b')$
```

Figure 24: **Games G_1–G_6 in the proof of Theorem 4.5.** Games G_2, G_3, G_6 contain the corresponding boxed statements, but games G_1, G_4, G_5 do not.

G Proof of Theorem 4.5

IND-CPA SECURITY. Consider the source S and distinguisher D in Fig. 23. Instead of sampling $r \leftarrow_{\$} \{0,1\}^{\rho(\lambda)}$ and computing $w \leftarrow r \oplus \mathsf{H.Ev}(1^\lambda, hk, m, 1^{|r|})$, the source directly samples $w \leftarrow_{\$} \{0,1\}^{\rho(\lambda)}$ and thus need not know the message m. Next, the source samples a lossy key lk instead of an injective key ek. The computation of $\mathsf{H.Ev}(1^\lambda, hk, y, 1^{|m|})$ and $\mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m|})$ is left to the distinguisher. We argue that the source S above is statistically unpredictable. Consider an arbitrary predictor P. Consider game $L_1^{S,P}$ that coincides with game Pred_S^P. Let game $L_2^{S,P}$ be identical to game $L_1^{S,P}$, except that the oracle HASH ignores consistency, and always return a fresh random answer for each query, even a repeated one. Recall that the source makes only two queries w and x to HASH, and $\Pr[x = w] \le 2^{-\rho}$. Hence

$$\Pr[L_1^{S,P}(\cdot)] - \Pr[L_2^{S,P}(\cdot)] \le 2^{-\rho} \ .$$

We now bound the chance that P wins in game L_2. Let q be a polynomial that bounds the size of the output of P and let τ be the lossiness of LT. The chance that the predictor can guess either x or w is at most $q/2^\rho + q/2^\tau$. Hence $\mathsf{Adv}_{S,P}^{\mathsf{pred}}(\cdot) \le 2q/2^\rho + q/2^\tau$. Next, consider the adversary B attacking the lossiness of LT and adversary C attacking the PRF security of F in Fig. 23. Adversary B is given a key K, which might be either an injective key ek or a lossy key lk. It then simulates game $\text{CPA}_{\mathsf{HE3}}^A$, but uses key K instead of ek. Adversary C simulates game $\text{UCE}_{\mathsf{H}}^{S,D}$ but the strings x and $seed$ are always chosen at random and $\mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$ is replaced by $\text{RR}(0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$. Consider the games G_1–G_6 in Fig. 24. Game G_1^A corresponds to game $\text{CPA}_{\mathsf{HE3}}^A$ and in game G_6^A, whatever the adversary receives is independent of the challenge bit b, and thus $\Pr[G_6^A(\cdot)] = 1/2$. Hence

$$\mathsf{Adv}_{\mathsf{HE3},A}^{\mathsf{ind\text{-}cpa}}(\cdot) = 2(\Pr[G_1^A(\cdot)] - \Pr[G_6^A(\cdot)]) \ .$$

```
$\underline{S^{\textsc{Hash}}(1^\lambda)}$
$b \leftarrow\$ \{0,1\}$ ; $lk \leftarrow\$ \mathsf{LT.LKg}(1^\lambda)$ ; $t \leftarrow\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; Return $(b, lk, t)$
$\underline{\textsc{LRSim}(d)}$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $w \leftarrow \textsc{Hash}(\mathbf{m}_b[i], 1^{\rho(\lambda)}) \oplus \mathbf{r}[i]$ ; $x \leftarrow \textsc{Hash}(w, 1^{\mathsf{LT.il}(\lambda)})$
    $seed \leftarrow \textsc{Hash}(x, 1^{\mathsf{F.kl}(\lambda)+\lambda})$ ; $y \leftarrow seed[1, \lambda]$ ; $fk \leftarrow seed[\lambda+1, |seed|]$
    $c \leftarrow \textsc{Hash}(y, 1^{|\mathbf{m}_b[i]|}) \oplus \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$
```

```
$\underline{D(1^\lambda, hk, L)}$
$(b, lk, t) \leftarrow L$
$b' \leftarrow\$ A_2(t, (lk, hk))$
If $(b' = b)$ then return 1
Else return 0
```

```
$\underline{B(1^\lambda, K)}$
$hk \leftarrow \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow\$ \{0,1\}$
$t \leftarrow\$ A_2^{\mathrm{LRSim}}(1^\lambda)$ ; $b' \leftarrow\$ A_2(t, (K, hk))$
If $(b = b')$ then return 1 else return 0
```

```
$\underline{\textsc{LRSim}(d)}$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $w \leftarrow \mathsf{H.Ev}(1^\lambda, hk, \mathbf{m}_b[i], 1^{\rho(\lambda)}) \oplus \mathbf{r}[i]$
    $x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, w, 1^{\mathsf{LT.il}(\lambda)})$ ; $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$
    $seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
    $y \leftarrow seed[1, \lambda]$ ; $fk \leftarrow seed[\lambda+1, |seed|]$
    $\mathbf{c}[i] \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|\mathbf{m}_b[i]|}) \oplus \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
Return $\mathbf{c}$
```

Figure 25: **Top:** Source S and distinguisher D in the IND-CDA proof of Theorem 4.5. **Bottom:** Adversary B attacking LT in the IND-CDA proof of Theorem 4.5.

We describe the game chain up to the terminal one. Game G_2 is identical to game G_1, except for the following changes. First, instead of sampling $r \leftarrow\$ \{0,1\}^{\rho(\lambda)}$ and computing $w \leftarrow \mathsf{H.Ev}(1^\lambda, hk, m, 1^{|r|}) \oplus r$, the game directly samples $w \leftarrow\$ \{0,1\}^{\rho(\lambda)}$. This change makes no difference. Next, instead of sampling an injective key ek, game G_2 samples a lossy key lk. Then

$$\Pr[G_1^A(\cdot)] - \Pr[G_2^A(\cdot)] = \mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) \ .$$

In game G_3, instead of calling $\mathsf{H.Ev}(1^\lambda, hk, \cdot, \cdot)$ to compute x and $seed$, we maintain a random oracle and make the corresponding queries to get x and $seed$. Then

$$\Pr[G_2^A(\cdot)] - \Pr[G_3^A(\cdot)] = \mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) \ .$$

Game G_4 is identical to G_3, except that we always sample x and $seed$ at random. The two games are identical-until-bad, and thus

$$\Pr[G_3^A(\cdot)] - \Pr[G_4^A(\cdot)] \le \Pr[G_4^A(\cdot) \text{ sets } \mathsf{bad}] \le 2^{-\rho} \ .$$

Game G_5 is a simplified version of game G_4. In game G_6, instead of using $\mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|m_b|})$, we use a uniformly random string. Hence

$$\Pr[G_5^A(\cdot)] - \Pr[G_6^A(\cdot)] = \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},C}(\cdot) \ .$$

Summing up,

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{HE3},A}(\cdot) \le 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) + 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) + 2\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},C}(\cdot) + 2^{1-\rho} \ .$$

IND-CDA SECURITY. Let $A = (A_1, A_2)$ be an adversary attacking IND-CDA security of $\mathsf{HE3}$. Consider the source S and distinguisher D in Fig. 25. They simulate game $\mathrm{CDA}^A_{\mathsf{HE3}}$ but use $\textsc{Hash}(\cdot,\cdot)$ instead of $\mathsf{H.Ev}(1^\lambda, hk, \cdot, \cdot)$ and use a lossy key lk instead of an injective key ek. Next, consider the adversary B attacking LT in Fig. 25. It's given a key K, which may be either an injective key ek or a lossy key lk. It then simulates game $\mathrm{CDA}^A_{\mathsf{HE3}}$, but K is used as the encryption key for LT. Consider games H_1–H_4 in Fig. 26. Game H_1^A corresponds to game $\mathrm{CDA}^A_{\mathsf{HE3}}$. We explain the game chain up to the terminal one.

```
GAME $H_1^A(\lambda)$, $\boxed{H_2^A(\lambda)}$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow_\$ \{0,1\}$
$(K, dk) \leftarrow_\$ \mathsf{LT.EKg}(1^\lambda)$ ; $\boxed{K \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)}$
$t \leftarrow_\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow_\$ A_2(t, (ek, hk))$
Return $(b = b')$

LR($d$)
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow_\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $w \leftarrow \mathsf{H.Ev}(1^\lambda, hk, \mathbf{m}_b[i], 1^{\mathsf{HE3.rl}(\lambda)}) \oplus \mathbf{r}[i]$
    $x \leftarrow \mathsf{H.Ev}(1^\lambda, hk, w, 1^{\mathsf{LT.il}(\lambda)})$
    $seed \leftarrow \mathsf{H.Ev}(1^\lambda, hk, x, 1^{\mathsf{F.kl}(\lambda)+\lambda})$
    $y \leftarrow seed[1, \lambda]$ ; $fk \leftarrow seed[\lambda+1, |seed|]$
    $mask \leftarrow \mathsf{H.Ev}(1^\lambda, hk, y, 1^{|\mathbf{m}_b[i]|})$
    $c \leftarrow mask \oplus \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, K, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$
```

```
GAME $\boxed{H_3^A(\lambda)}$, $H_4^A(\lambda)$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow_\$ \{0,1\}$ ; $lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)$
$t \leftarrow_\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow_\$ A_2(t, (ek, hk))$ ; Return $(b = b')$

LR($d$)
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{r}) \leftarrow_\$ A_1(1^\lambda, d)$
For $i = 1$ to $|\mathbf{r}|$ do
    $w \leftarrow \overline{\mathsf{RO}}(\mathbf{m}_b[i], \rho(\lambda)) \oplus \mathbf{r}[i]$ ; $x \leftarrow \mathsf{RO}(w, \mathsf{LT.il}(\lambda))$
    $seed \leftarrow \mathsf{RO}(x, \mathsf{F.kl}(\lambda) + \lambda)$ ; $y \leftarrow seed[1, \lambda]$
    $fk \leftarrow seed[\lambda+1, |seed|]$ ; $mask \leftarrow \mathsf{RO}(y, |\mathbf{m}_b[i]|)$
    $c \leftarrow mask \oplus \mathsf{F.Ev}(1^\lambda, fk, 0^{\mathsf{F.il}(\lambda)}, 1^{|\mathbf{m}_b[i]|}) \oplus \mathbf{m}_b[i]$
    $trap \leftarrow \mathsf{LT.Ev}(1^\lambda, lk, x)$ ; $\mathbf{c}[i] \leftarrow (trap, c)$
Return $\mathbf{c}$

$\mathsf{RO}(x, \ell)$
$y \leftarrow_\$ \{0,1\}^\ell$
If $H[x, \ell] \neq \perp$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{y \leftarrow H[x, \ell]}$
$H[x, \ell] \leftarrow y$ ; Return $y$

$\overline{\mathsf{RO}}(x, \ell)$
$y \leftarrow_\$ \{0,1\}^\ell$
If $H[x, \ell] \neq \perp$ then $y \leftarrow H[x, \ell]$
$H[x, \ell] \leftarrow y$ ; Return $y$
```

Figure 26: **Games H_1–H_4 in the proof of Theorem 4.5**. Games H_2 and H_3 contain the corresponding boxed statements, but games H_1 and H_4 do not.

Game H_2 is identical to H_2, except that we use a lossy key lk instead of the injective key ek. Then

$$\Pr[H_1^A(\cdot)] - \Pr[H_2^A(\cdot)] = \mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\cdot) \ .$$

Game H_3 is identical to game H_2, but we maintain a random oracle RO and calls to $\mathsf{H.Ev}(1^\lambda, hk, \cdot, \cdot)$ are replaced by the corresponding queries to RO. We also maintain another interface $\overline{\mathsf{RO}}$ of RO. Queries $\mathsf{RO}(\mathbf{m}_b[i], \rho(\lambda))$ are replaced by $\overline{\mathsf{RO}}(\mathbf{m}_b[i], \rho(\lambda))$. Then

$$\Pr[H_2^A(\cdot)] - \Pr[H_3^A(\cdot)] = \mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\cdot) \ .$$

In game H_4, we ignore consistency among queries to RO, giving independent answers, but the code for $\overline{\mathsf{RO}}$ doesn't change. The two games H_3 and H_4 are identical-until-bad, and thus

$$\Pr[H_3^A(\cdot)] - \Pr[H_4^A(\cdot)] \le \Pr[H_4^A(\cdot) \text{ sets } \mathsf{bad}] \ .$$

We now bound the chance that game H_4^A sets bad. Let p be the total number of messages that A produces. There are $3p$ queries to RO. Each query is either (i) a uniformly random string of length at least λ or (ii) a string w of the form $\overline{\mathsf{RO}}(m) \oplus r$ for some message m and coin r produced by A. There are $2p$ queries in category (i), and the chance that they trigger bad is at most

$$\sum_{j=p}^{3p-1} \frac{j}{2^\lambda} \le \frac{4p^2}{2^\lambda} \ .$$

For queries in category (ii), note that A_2 can only specify *distributions*, and within a distribution, it's guaranteed that the strings (m, r) are distinct. Let Bad be the event that the (lazily constructed) domains of RO and $\overline{\mathsf{RO}}$ overlap. Suppose that Bad doesn't occur. Then $\overline{\mathsf{RO}}$ is a random oracle, and its answers are not given to A_2.

- Consider two queries $w = \overline{\mathsf{RO}}(m) \oplus r$ and $w' = \overline{\mathsf{RO}}(m') \oplus r'$ within the same distribution. If $m = m'$ then $r \neq r'$, and thus $w \neq w'$. If $m \neq m'$ then the chance that $w = w'$ is at most $2^{-\rho}$.
- Consider two queries $w = \overline{\mathsf{RO}}(m) \oplus r$ and $w' = \overline{\mathsf{RO}}(m') \oplus r'$ in different distributions. If $m \neq m'$ then the chance that $w = w'$ is at most $2^{-\rho}$. Suppose that $m = m'$. Then $w = w'$ if and only if $r = r'$. The chance that $m = m'$ and $r = r'$ is at most $\mathrm{Guess}_A(\cdot)$.
- Consider a query in category (ii) and a prior query in category (i). The chance that they are equal is at most $2^{-\rho}$.

Summing up, if Bad doesn't occur then the chance that queries in category (ii) trigger bad is at most $2.5p^2/2^\rho + 0.5p^2 \cdot \mathrm{Guess}_A(\cdot)$. We now bound the chance that Bad occurs. Terminate the game immediately when Bad occurs; it doesn't affect the chance that Bad occurs. We then can still consider $\overline{\mathsf{RO}}$ a random oracle that is independent of RO. Note that A_2 only receives (1) the images of strings x under LT on a lossy key, and (2) random strings independent of the inputs of RO and the outputs of $\overline{\mathsf{RO}}$. The marginal distribution of each input to RO is uniformly random, independent of the prior inputs to $\overline{\mathsf{RO}}$. Then

$$\Pr[\mathsf{Bad}] \le \sum_{i=1}^{p} \frac{3i}{2^\rho} + \sum_{j=0}^{p-1} \Big(\frac{2j}{2^\rho} + \frac{j}{2^\tau} \Big) \le \frac{3p^2}{2^\rho} + \frac{0.5p^2}{2^\tau},$$

where τ is the lossiness of LT. Summing up,

$$\Pr[H_4^A(\lambda) \text{ sets } \mathsf{bad}] \le 0.5p^2 \cdot \mathrm{Guess}_A(\lambda) + \frac{4p^2}{2^\lambda} + \frac{5.5p^2}{2^{\rho(\lambda)}} + \frac{0.5p^2}{2^{\tau(\lambda)}}$$

for all $\lambda \in \mathbb{N}$. Finally, in game H_4, whatever A receives is independent of the challenge bit, and thus $\Pr[H_4^A(\cdot)] = 1/2$. Summing up,

$$\begin{aligned} \mathsf{Adv}^{\mathsf{cda}}_{\mathsf{HE},A}(\lambda) &= 2(\Pr[H_1^A(\lambda)] - \Pr[H_4^A(\lambda)]) \\ &\le 2\mathsf{Adv}^{\mathsf{ltdf}}_{\mathsf{LT},B}(\lambda) + 2\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\lambda) + p^2 \cdot \mathrm{Guess}_A(\lambda) + \frac{8p^2}{2^\lambda} + \frac{12p^2}{2^{\min\{\tau(\lambda),\rho(\lambda)\}}} \end{aligned}$$

for all $\lambda \in \mathbb{N}$. What remains is to prove that S is statistically reset-secure. Let R be a reset adversary. Wlog, assume that R never repeats a prior oracle query. Consider the following games J_1–J_4 in Fig. 27. Game $J_1^{A,R}$ is identical to game Reset_S^R with challenge bit $a = 1$. We explain the game chain up to the terminal one. Game J_2 is identical to game J_1, but we ignore consistency among RO queries, giving independent answers. The two games J_1 and J_2 are identical-until-bad, and thus

$$\Pr[J_1^{A,R}(\lambda)] - \Pr[J_2^{A,R}(\lambda)] \le \Pr[J_2^{A,R}(\lambda) \text{ sets } \mathsf{bad}] \le 0.5p^2 \cdot \mathrm{Guess}_A(\lambda) + \frac{4p^2}{2^\lambda} + \frac{5.5p^2}{2^{\rho(\lambda)}} + \frac{0.5p^2}{2^{\tau(\lambda)}},$$

for all $\lambda \in \mathbb{N}$. Game J_3 is is a simplified version of game J_2. Game J_4 is identical to game J_3, except for the following changes. In each game, we maintain the set X of queries to RO. In game J_4, if R queries a string $v \in X$, we'll return a fresh random answer, instead of giving the value consistent with RO. Games J_3 and J_4 are identical-until-bad, and thus

$$\Pr[J_3^{A,R}(\lambda)] - \Pr[J_4^{A,R}(\lambda)] \le \Pr[J_4^{A,R}(\lambda) \text{ sets } \mathsf{bad}]$$

for all $\lambda \in \mathbb{N}$. Let q be a polynomial that bounds the number of $\textsc{Hash}$ queries of R. To trigger bad, one of $\textsc{Hash}$ queries of R must hit one of the p strings $w = \overline{\mathsf{RO}}(m, \rho(\lambda)) \oplus r$ or $2p$ uniformly random strings x, y, given only $\mathsf{LT.Ev}(1^\lambda, lk, x)$. From Lemma 4.4,

$$\Pr[J_4^{A,R}(\lambda) \text{ sets } \mathsf{bad}] \le p\Big(q \cdot \mathrm{Guess}_A(\lambda) + \frac{q^2}{2^{\rho(\lambda)/3}}\Big) + \frac{p}{2^\lambda} + \frac{p}{2^{\tau(\lambda)}}$$

for every $\lambda \in \mathbb{N}$. Note that in game J_4, the answers of $\textsc{Hash}$ are uniformly random strings, independent of the leakage L that R receives. Hence

$$\Pr[J_4^{A,R}(\cdot)] = \Pr[\,\mathrm{Reset}_S^R(\cdot) \mid a = 0\,] \ .$$

```
GAME $\boxed{J_1^{A,R}(\lambda)}$, $J_2^{A,R}(\lambda)$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow_\$ \{0,1\}$ ; $lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)$
$t \leftarrow_\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow_\$ R^{\mathrm{HASH}}(1^\lambda, (b, lk, t))$
Return $(b' = 1)$

$\mathsf{RO}(x, \ell)$
$y \leftarrow_\$ \{0,1\}^\ell$ ; $X \leftarrow X \cup \{x\}$
If $H[x, \ell] \neq \perp$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{y \leftarrow H[x, \ell]}$
$H[x, \ell] \leftarrow y$ ; Return $y$

$\overline{\mathsf{RO}}(x, \ell)$
$y \leftarrow_\$ \{0,1\}^\ell$
If $H[x, \ell] \neq \perp$ then $y \leftarrow H[x, \ell]$
$H[x, \ell] \leftarrow y$ ; Return $y$

$\mathrm{HASH}(x, 1^\ell)$
$y \leftarrow_\$ \{0,1\}^\ell$
If $H[x, \ell] = \perp$ then return $y$ else return $H[x, \ell]$
```

```
GAME $J_3^{A,R}(\lambda)$, $\boxed{J_4^{A,R}(\lambda)}$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $b \leftarrow_\$ \{0,1\}$ ; $lk \leftarrow_\$ \mathsf{LT.LKg}(1^\lambda)$
$t \leftarrow_\$ A_2^{\mathrm{LR}}(1^\lambda)$ ; $b' \leftarrow_\$ R^{\mathrm{HASH}}(1^\lambda, (b, lk, t))$
Return $(b' = 1)$

$\mathsf{RO}(x, \ell)$
$H[x, \ell] \leftarrow_\$ \{0,1\}^\ell$ ; $X \leftarrow X \cup \{x\}$ ; Return $H[x, \ell]$

$\overline{\mathsf{RO}}(x, \ell)$
$y \leftarrow_\$ \{0,1\}^\ell$
If $H[x, \ell] \neq \perp$ then $y \leftarrow H[x, \ell]$
$H[x, \ell] \leftarrow y$ ; Return $y$

$\mathrm{HASH}(x, 1^\ell)$
$y \leftarrow_\$ \{0,1\}^\ell$
If $H[x, \ell] = \perp$ then return $y$
Elsif $x \in X$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{\text{return } y}$
Return $H[x, \ell]$
```

Figure 27: **Games J_1–J_4 in the proof of Theorem 4.5**. Games J_1 and J_4 contain the corresponding boxed statements, but games J_2 and J_3 do not. The games use the same procedure LR(d) as games H_3 and H_4 in Fig. 26 and thus the code of this procedure is omitted for simplicity.

Summing up,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{reset}}_{S,R}(\lambda) &= \Pr[J_1^{A,R}(\lambda)] - \Pr[J_4^{A,R}(\lambda)] \\ &\leq p(p+q) \cdot \mathrm{Guess}_A(\lambda) + \frac{5p^2}{2^\lambda} + \frac{6.5p^2}{2^{\min\{\rho(\lambda),\tau(\lambda)\}}} + \frac{pq^2}{2^{\rho(\lambda)/3}}\end{aligned}$$

for all $\lambda \in \mathbb{N}$.

H Practical output-length extension for UCE

So far we have assumed that there is a variable-output-length (VOL) UCE-secure family, which is needed for both D-PKE and H-PKE. In practice a cryptographic hash such as HMAC-SHA-256 only gives us up to 512 bits. BHK2 [9] construct a practical FIL, VOL hash and show that it's $\mathsf{UCE}[\mathcal{S}^{\mathrm{sup}}]$-secure in the random-oracle model. The idea is simple: given the input x and the wanted output length ℓ, we use, say HMAC-sha-256 to derive an AES key K and then use AES-CTR mode to expand the output; the payload of AES also consists of an encoding of ℓ in addition to the counter. Here sha-256 is the compression function of SHA-256. This construction is as fast as AES-CTR.

In a different direction, BHK1 [8] give a method to turn a FOL UCE-secure hash to a VOL one in the standard model: they apply the CTR mode on the hash itself, the payload is $1^\ell \,\|\, 0 \,\|\, 1^i \,\|\, 0 \,\|\, x$, where i is the counter. This construction is impractical: if one wants to hash m-bit input to get n-bit output, BHK1's method will need to process $O((m+n)m)$ bits. In this section, we'll give a practical variant of BHK1's method. Our method is fully parallelizable and processes $O(n+m)$ bits, which is optimal. The idea is to use BHK2's hash-then-CTR paradigm, but using CTR mode on the hash itself instead of AES; the payload of CTR mode will consist of an encoding of K and ℓ, in addition to the counter. To get a VIL, VOL hash, one can apply our extension on a VOL hash such as HMAC-SHA-256. Alternatively, one can apply our extension on HMAC-sha-256 to get a FIL, VOL hash, followed by the AU-then-Hash transform of BHK2 to make it VIL.

Formally, let $\overline{\mathsf{H}}$ be a function family with $3\lambda \in \overline{\mathsf{H}}.\mathsf{IL}(\lambda)$ and $\overline{\mathsf{H}}.\mathsf{ol}(\lambda) \geq \lambda$ for every $\lambda \in \mathbb{N}$. We'll construct

```
$\underline{\mathsf{H}(1^\lambda, hk, x, 1^\ell)}$
$m \leftarrow \lceil \ell/\overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$ ; $K \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, hk, \ell \,\|\, 0^\lambda \,\|\, x, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})[1,\lambda]$
For $i = 1$ to $m$ do $y_i \leftarrow \overline{\mathsf{H}}.\mathsf{Ev}(1^\lambda, hk, \ell \,\|\, i \,\|\, K, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1,\ell]$
```

Figure 28: **Construction of a VOL family $\mathsf{H} = \mathsf{Extend}[\overline{\mathsf{H}}]$ from a FOL family $\overline{\mathsf{H}}$.** The two families use the same key generation algorithm. The string ℓ is encoded as a λ-bit string. The number $i \in \{0, 1, \ldots, 2^\lambda - 1\}$ is also encoded as a λ-bit string, with the first bit as the most significant bit.

---

a VOL function family $\mathsf{H} = \mathsf{Extend}[\overline{\mathsf{H}}]$, with $\mathsf{H}.\mathsf{IL}(\lambda) = \{n - 2\lambda \mid n \in \overline{\mathsf{H}}.\mathsf{IL}(\lambda) \text{ and } n \geq 2\lambda\}$ and $\mathsf{H}.\mathsf{OL}(\lambda) = \{1, 2, \ldots, 2^\lambda - 1\}$ for every $\lambda \in \mathbb{N}$. The construction Extend is specified in Fig. 28. We note that the loop counter i in the code of Extend is never encoded as 0^λ, and thus there is effectively a domain separation in using $\overline{\mathsf{H}}$ for hashing $\ell \,\|\, 0^\lambda \,\|\, x$ and $\ell \,\|\, i \,\|\, K$. Theorem H.1 below shows the concrete security bounds in using Extend for extending the output length of UCE-secure function families.

Theorem H.1 Let $\overline{\mathsf{H}}$ be a function family with $3\lambda \in \overline{\mathsf{H}}.\mathsf{IL}(\lambda)$ and $\overline{\mathsf{H}}.\mathsf{ol}(\lambda) \geq \lambda$ for every $\lambda \in \mathbb{N}$. Let $\mathsf{H} = \mathsf{Extend}[\overline{\mathsf{H}}]$.

(a) If $\overline{\mathsf{H}}$ is $\mathsf{UCE}[\mathcal{S}^{\rm sup}]$-secure then so is H. Concretely, let S be a source, D a distinguisher, and $\overline{P}$ a predictor. We can construct source $\overline{S}$, distinguisher $\overline{D}$, and predictor P such that

$$\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\lambda) \leq \mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\lambda) + \frac{q^2}{2^\lambda}$$

$$\mathsf{Adv}^{\mathsf{pred}}_{\overline{S},\overline{P}}(\lambda) \leq \mathsf{Adv}^{\mathsf{pred}}_{S,P}(\lambda) + \frac{p(p+q)}{2^\lambda}$$

for all $\lambda \in \mathbb{N}$, where $q = \mathbf{Q}^{\rm HASH}_{S}$ and p is the maximum of the size of $\overline{P}$'s output in the execution of $\mathrm{Pred}^{\overline{P}}_{\overline{S}}$. Furthermore, $\mathbf{T}(\mathrm{UCE}^{\overline{S},\overline{D}}_{\overline{\mathsf{H}}}) \leq \mathbf{T}(\mathrm{UCE}^{S,D}_{\mathsf{H}})$; the size of the output of P in the execution of Pred^{P}_{S} is at most p; and $\mathbf{Q}^{\rm HASH}_{\overline{S}}$ is at most the maximum of the $\overline{\mathsf{H}}.\mathsf{ol}$-bit blocks in the output length of S's queries in $\mathrm{UCE}^{S,D}_{\mathsf{H}}$.

(b) If $\overline{\mathsf{H}}$ is $\mathsf{UCE}[\mathcal{S}^{\rm srs}]$-secure then so is H. Concretely, let S be a source, D a distinguisher, and $\overline{R}$ a reset adversary. We can construct source $\overline{S}$, distinguisher $\overline{D}$, and reset adversary R such that

$$\mathsf{Adv}^{\mathsf{uce}}_{\mathsf{H},S,D}(\lambda) \leq \mathsf{Adv}^{\mathsf{uce}}_{\overline{\mathsf{H}},\overline{S},\overline{D}}(\lambda) + \frac{q^2}{2^\lambda}$$

$$\mathsf{Adv}^{\mathsf{reset}}_{\overline{S},\overline{R}}(\lambda) \leq \mathsf{Adv}^{\mathsf{reset}}_{S,R}(\lambda) + \frac{2(p+q)^2}{2^\lambda}$$

for all $\lambda \in \mathbb{N}$, where $q = \mathbf{Q}^{\rm HASH}_{S}$ and $p = \mathbf{Q}^{\rm HASH}_{\overline{R}}$. Furthermore, $\mathbf{T}(\mathrm{UCE}^{\overline{S},\overline{D}}_{\overline{\mathsf{H}}}) \leq \mathbf{T}(\mathrm{UCE}^{S,D}_{\mathsf{H}})$; $\mathbf{Q}^{\rm HASH}_{R} \leq p$; and $\mathbf{Q}^{\rm HASH}_{\overline{S}}$ is at most the maximum of the $\overline{\mathsf{H}}.\mathsf{ol}$-bit blocks of the output length of S's queries in $\mathrm{UCE}^{S,D}_{\mathsf{H}}$. ▌

Proof: Let S be a source and D be a distinguisher. We now construct a source $\overline{S}$ as follows.

```
$\underline{\overline{S}^{\rm HASH}(1^\lambda)}$
$L \leftarrow_\$ S^{\rm HASHSIM}(1^\lambda)$
Return $L$

$\underline{\rm HASHSIM(x, 1^\ell)}$
$m \leftarrow \lceil \ell/\overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$ ; $K \leftarrow \mathrm{HASH}(\ell \,\|\, 0^\lambda \,\|\, x, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})[1,\lambda]$
For $i = 1$ to $m$ do $y_i \leftarrow \mathrm{HASH}(\ell \,\|\, i \,\|\, K, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1,\ell]$
```

```
Game $G_1^{S,D}(\lambda)$, $\boxed{G_2^{S,D}(\lambda)}$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $L \leftarrow_\$ S^{\textsc{HashSim}}(1^\lambda)$ ; $b' \leftarrow_\$ D(1^\lambda, hk, L)$
Return $(b' = 0)$

$\textsc{HashSim}(x, 1^\ell)$
$m \leftarrow \lceil \ell/\overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$ ; $K \leftarrow_\$ \{0,1\}^\lambda$
If $K \in \mathsf{Dom}$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{K \leftarrow_\$ \{0,1\}^\lambda \backslash \mathsf{Dom}}$
$\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{K\}$ ; $Q \leftarrow Q \cup \{\ell \,\|\, 0^\lambda \,\|\, x\}$
For $i = 1$ to $m$ do
    $y_i \leftarrow \mathsf{RO}(\ell \,\|\, i \,\|\, K)$ ; $Q \leftarrow Q \cup \{\ell \,\|\, i \,\|\, K\}$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1, \ell]$

$\mathsf{RO}(x)$
If $H[x] = \bot$ then $H[x] \leftarrow_\$ \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$
Return $H[x]$
```

```
Game $\boxed{G_3^{S,D}(\lambda),}$ $G_4^{S,D}(\lambda)$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $L \leftarrow_\$ S^{\textsc{HashSim}}(1^\lambda)$
$b' \leftarrow_\$ D(1^\lambda, hk, L)$ ; Return $(b' = 0)$

$\textsc{HashSim}(x, 1^\ell)$
$m \leftarrow \lceil \ell/\overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$ ; $K \leftarrow_\$ \{0,1\}^\lambda$
If $K \in \mathsf{Dom}$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{K \leftarrow_\$ \{0,1\}^\lambda \backslash \mathsf{Dom}}$
$\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{K\}$ ; $Q \leftarrow Q \cup \{\ell \,\|\, 0^\lambda \,\|\, x\}$
For $i = 1$ to $m$ do
    $y_i \leftarrow_\$ \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$ ; $Q \leftarrow Q \cup \{\ell \,\|\, i \,\|\, K\}$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1, \ell]$
```

Figure 29: **Games G_1–G_4 in the proof of Theorem H.1.** Games G_2 and G_3 include the corresponding boxed statements but games G_1 and G_4 don't. Sets are initialized to $\emptyset$. The games also lazily implement a random oracle $\mathsf{RO} : \{0,1\}^* \to \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$.

Then

$$\Pr[\,\mathrm{UCE}_{\overline{\mathsf{H}}}^{\overline{S},D}(\cdot) \mid \overline{a} = 1\,] = \Pr[\,\mathrm{UCE}_{\mathsf{H}}^{S,D}(\cdot) \mid a = 1\,] \ .$$

where a and $\overline{a}$ are the challenge bits of game $\mathrm{UCE}_{\mathsf{H}}^{S,D}$ and $\mathrm{UCE}_{\overline{\mathsf{H}}}^{\overline{S},D}$ respectively. Wlog, assume that S never repeats an oracle query. Then each query $\ell \,\|\, 0^\lambda \,\|\, x$ of $\overline{S}$ is never repeated, because the loop counter i is never encoded as 0^λ. Consider games G_1–G_4 in Fig. 29. Game $G_1^{S,D}$ coincides with game $\mathrm{UCE}_{\overline{\mathsf{H}}}^{\overline{S},D}$ for challenge bit $\overline{a} = 0$, and game $G_4^{S,D}$ coincides with game $\mathrm{UCE}_{\mathsf{H}}^{S,D}$ for challenge bit $a = 0$. We explain the game chain up to the terminal one. Game $G_2^{S,D}$ is identical to game $G_1^{S,D}$, except that now we make sure that K never repeats. Let q be a polynomial that bounds the number of Hash queries of S. Since games G_1 and G_2 are identical-until-bad, we have

$$\Pr[G_1^{S,D}(\lambda)] - \Pr[G_2^{S,D}(\lambda)] \le \Pr[G_1^{S,D}(\lambda) \text{ sets } \mathsf{bad}] \le \frac{q^2}{2^{\lambda+1}}$$

for all $\lambda \in \mathbb{N}$. Note that in game G_2, each query $\ell \,\|\, i \,\|\, K$ is never repeated. Hence effectively, we always sample each y_i at random. Game G_3 makes this explicit; it is still equivalent to game G_2. In game G_4, we now sample K at random. Games G_3 and G_4 are identical-until-bad, and thus

$$\Pr[G_3^{S,D}(\lambda)] - \Pr[G_4^{S,D}(\lambda)] \le \Pr[G_4^{S,D}(\lambda) \text{ sets } \mathsf{bad}] \le \frac{q^2}{2^{\lambda+1}}$$

for all $\lambda \in \mathbb{N}$. Hence

$$\Pr[\,\mathrm{UCE}_{\overline{\mathsf{H}}}^{\overline{S},D}(\lambda) \mid \overline{a} = 0\,] \le \Pr[\,\mathrm{UCE}_{\mathsf{H}}^{S,D}(\lambda) \mid a = 0\,] + \frac{q^2}{2^\lambda}$$

for all $\lambda \in \mathbb{N}$, and thus

$$\mathsf{Adv}_{\mathsf{H},S,D}^{\mathsf{uce}}(\lambda) \le \mathsf{Adv}_{\overline{\mathsf{H}},\overline{S},D}^{\mathsf{uce}}(\lambda) + \frac{q^2}{2^\lambda}$$

for all $\lambda \in \mathbb{N}$.

Unpredictability. Let $\overline{P}$ be an arbitrary predictor. Consider the following predictor P.

```
Game $H_5^{S,\overline{P}}(\lambda)$, $\boxed{H_6^{S,\overline{P}}(\lambda)}$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $L \leftarrow_\$ S^{\text{HashSim}}(1^\lambda)$ ; $U \leftarrow_\$ R(1^\lambda, L)$
For $u \in U$ do
  If $|u| = 3\lambda$ then $\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{u[2\lambda+1, 3\lambda]\}$
For $(\ell \,\|\, 0^\lambda \,\|\, x) \in Q$ do
  $m \leftarrow \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$ ; $K \leftarrow_\$ \{0,1\}^\lambda$
  If $K \in \mathsf{Dom}$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{K \leftarrow_\$ \{0,1\}^\lambda \backslash \mathsf{Dom}}$
  For $i = 1$ to $m$ do $Q \leftarrow Q \cup \{\ell \,\|\, i \,\|\, K\}$
Return $(U \cap Q \neq \emptyset)$

HashSim$(x, 1^\ell)$
$m \leftarrow \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$ ; $Q \leftarrow Q \cup \{\ell \,\|\, 0^\lambda \,\|\, x\}$
For $i = 1$ to $m$ do $y_i \leftarrow_\$ \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1, \ell]$
```

```
Game $H_7^{S,\overline{P}}(\lambda)$
$hk \leftarrow_\$ \mathsf{H.Kg}(1^\lambda)$ ; $L \leftarrow_\$ S^{\text{HashSim}}(1^\lambda)$ ; $U \leftarrow_\$ R(1^\lambda, L)$
Return $(U \cap Q \neq \emptyset)$

HashSim$(x, 1^\ell)$
$m \leftarrow \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$ ; $Q \leftarrow Q \cup \{\ell \,\|\, 0^\lambda \,\|\, x\}$
For $i = 1$ to $m$ do $y_i \leftarrow_\$ \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1, \ell]$
```

Figure 30: **Games H_5–H_6 in the proof of Theorem H.1.** Game H_6 includes the corresponding boxed statements but game H_5 doesn't. Sets are initialized to $\emptyset$.

```
$P(1^\lambda, L)$
$U \leftarrow_\$ \overline{P}(1^\lambda, L)$ ; $Q' \leftarrow \emptyset$
For $u \in U$ do
  If $u[\lambda+1, 2\lambda] = 0^\lambda$ then $x \leftarrow u[2\lambda+1, |u|]$ ; $Q' \leftarrow Q' \cup \{x\}$
Return $Q'$
```

For each $i \in \{1,2,3,4\}$, let game $H_i^{S,\overline{P}}$ be identical to game $G_i^{S,D}$, but instead of running $b' \leftarrow_\$ D(1^\lambda, hk, L)$ and returning $(b' = 0)$, we'll run $U \leftarrow_\$ \overline{P}(1^\lambda, L)$ and return $U \cap Q \neq \emptyset$. Game $H_1^{S,\overline{P}}$ coincides with game $\text{Pred}_S^{\overline{P}}$; games $H_2^{S,\overline{P}}$ and $H_3^{S,\overline{P}}$ are equivalent; and

$$\Pr[H_1^{S,\overline{P}}(\lambda)] - \Pr[H_2^{S,\overline{P}}(\lambda)] \;\le\; \Pr[H_1^{S,\overline{P}}(\lambda) \text{ sets } \mathsf{bad}] \le \frac{q^2}{2^{\lambda+1}}, \text{ and}$$

$$\Pr[H_3^{S,\overline{P}}(\lambda)] - \Pr[H_4^{S,\overline{P}}(\lambda)] \;\le\; \Pr[H_4^{S,\overline{P}}(\lambda) \text{ sets } \mathsf{bad}] \le \frac{q^2}{2^{\lambda+1}}$$

for all $\lambda \in \mathbb{N}$. Consider games H_5–H_7 in Fig. 30. Game $H_5^{S,\overline{P}}$ is equivalent to game $H_4^{S,\overline{P}}$: instead of eagerly generating keys K in procedure HashSim, we lazily generate them right after R outputs its set U. Game $H_6^{S,\overline{P}}$ is identical to game $H_5^{S,\overline{P}}$, but we sample K so that no element of U can be of the form $\ell \,\|\, i \,\|\, K$. Let p be a polynomial that bounds $|U|$. Games H_5 and H_6 are identical-until-bad, and thus

$$\Pr[H_5^{S,\overline{P}}(\lambda)] - \Pr[H_6^{S,\overline{P}}(\lambda)] \le \Pr[H_5^{S,\overline{P}}(\lambda) \text{ sets } \mathsf{bad}] \le \frac{pq}{2^\lambda}$$

for all $\lambda \in \mathbb{N}$. Game $H_7^{S,\overline{P}}$ is equivalent to game $H_6^{S,\overline{P}}$: since no element of U can hit $\ell \,\|\, i \,\|\, K$, there is no need to generate strings K and add $\ell \,\|\, i \,\|\, K$ to Q. On the other hand, since game $H_7^{S,\overline{P}}$ coincides with game Pred_S^P, it follows that $\mathsf{Adv}^{\mathsf{pred}}_{S,\overline{P}}(\lambda) \le \mathsf{Adv}^{\mathsf{pred}}_{S,P}(\lambda) + \frac{p(p+q)}{2^\lambda}$.

Reset security. Let $\overline{R}$ be an arbitrary reset adversary. Consider the reset adversary R in Fig. 31. Wlog, assume that adversary $\overline{R}$ never repeats an oracle query. Consider games J_1–J_6 in Fig. 32. Game $J_1^{S,\overline{R}}$ corresponds to game Reset_S^R with challenge bit $b = 0$, and game $J_6^{S,\overline{R}}$ corresponds to game $\text{Reset}_{\overline{S}}^{\overline{R}}$ with challenge bit $\overline{b} = 0$. We explain the game chain up to the terminal one. Let p be a polynomial that bounds the number of oracle queries of $\overline{R}$. Game $J_2^{S,\overline{R}}$ is identical to game $J_1^{S,\overline{R}}$, except that the

```
R^{Hash}(1^λ, L)
    b′ ←$ R̄^{HashSim}(1^λ, L)
    Return b′

HashSim(u, 1^{H̄.ol(λ)})
    r ←$ {0,1}^{H̄.ol(λ)}
    If (|u| ≥ 2λ) ∧ (u[λ+1, 2λ] = 0^λ) then
        ℓ ← u[1, λ] ; x ← u[2λ+1, |u|] ; K ←$ {0,1}^λ
        pad ←$ {0,1}^{H̄.ol(λ)−λ} ; H[K, ℓ] ← x ; Return K ∥ pad
    If |u| = 3λ then
        ℓ ← u[1, λ] ; i ← x[λ+1, 2λ] ; K ← u[2λ+1, 3λ]
        If (i > ⌈ℓ/H̄.ol(λ)⌉) ∨ (H[K, ℓ] = ⊥) then return r
        x ← H[K, ℓ] ; pad ←$ {0,1}^{H̄.ol(λ)−(ℓ mod H̄.ol(λ))} ; y ← Hash(x, 1^ℓ) ∥ pad
        Return y[(i−1)·H̄.ol(λ)+1, i·H̄.ol(λ)]
    Return r
```

Figure 31: **Constructed reset adversary R for the proof of Theorem H.1.**

```
Game J_1^{S,R̄}(λ), [J_2^{S,R̄}(λ)]
    L ←$ S^{Hash}(1^λ) ; b′ ←$ R̄^{HashSim}(1^λ, L) ; Return (b′ = 0)

Hash(x, 1^ℓ)
    Return RO_1(x, ℓ)

HashSim(u, 1^{H̄.ol(λ)})
    r ←$ {0,1}^{H̄.ol(λ)}
    If (|u| ≥ 2λ) ∧ (u[λ+1, 2λ] = 0^λ) then
        ℓ ← u[1, λ] ; x ← u[2λ+1, |u|] ; K ←$ {0,1}^λ
        If K ∈ Dom then bad ← true ; [K ←$ {0,1}^λ \ Dom]
        Dom ← Dom ∪ {K} ; H[K, ℓ] ← x ; pad ←$ {0,1}^{H̄.ol(λ)−λ} ; Return K ∥ pad
    If |u| = 3λ then
        ℓ ← u[1, λ] ; i ← x[λ+1, 2λ] ; K ← u[2λ+1, 3λ]
        If (i > ⌈ℓ/H̄.ol(λ)⌉) ∨ (H[K, ℓ] = ⊥) then return r
        x ← H[K, ℓ] ; pad ←$ {0,1}^{H̄.ol(λ)−(ℓ mod H̄.ol(λ))} ; y ← RO_2(x, ℓ) ∥ pad
        Return y[(i−1)·H̄.ol(λ)+1, i·H̄.ol(λ)]
    Return r
```

```
Game [J_3^{S,R̄}(λ)], J_4^{S,R̄}(λ)
    L ←$ S^{Hash}(1^λ) ; b′ ←$ R̄^{HashSim}(1^λ, L)
    Return (b′ = 0)

Hash(x, 1^ℓ)
    Return RO_1(x, ℓ)

HashSim(u, 1^{H̄.ol(λ)})
    r ←$ {0,1}^{H̄.ol(λ)}
    If (|u| ≥ 2λ) ∧ (u[λ+1, 2λ] = 0^λ) then
        K ←$ {0,1}^λ ; pad ←$ {0,1}^{2λ}
        If K ∈ Dom then bad ← true ; [K ←$ {0,1}^λ \ Dom]
        Dom ← Dom ∪ {K} ; Return K ∥ pad
    Return r
```

```
Game [J_5^{S,R̄}(λ)], J_6^{S,R̄}(λ)
    L ←$ S^{Hash}(1^λ) ; b′ ←$ R̄^{HashSim}(1^λ, L)
    Return (b′ = 0)

Hash(x, 1^ℓ)
    m ← ⌈ℓ/H̄.ol(λ)⌉ ; K ←$ {0,1}^λ
    If K ∈ Dom then bad ← true ; [K ←$ {0,1}^λ \ Dom]
    For i = 1 to m do y_i ← RO_3(ℓ ∥ i ∥ K, H̄.ol(λ))
    Dom ← Dom ∪ {K} ; y ← y_1 ∥ ··· ∥ y_m ; Return y[1, ℓ]

HashSim(u, 1^{H̄.ol(λ)})
    r ←$ {0,1}^{H̄.ol(λ)} ; Return r
```

Figure 32: **Games J_1–J_6 for the proof of Theorem H.1.** Games J_2, J_3, J_5 include the corresponding boxed statements but games J_1, J_4, J_6 don't, and $\mathsf{RO}_1, \mathsf{RO}_2, \mathsf{RO}_3$ are independent random oracles.

strings K are ensured to be distinct. The two games are identical-until-bad, and thus

$$\Pr[J_1^{S,\overline{R}}(\lambda)] - \Pr[J_2^{S,R}(\lambda)] \le \Pr[J_2^{S,\overline{R}}(\lambda) \text{ sets bad}] \le \frac{p^2}{2^{\lambda+1}}$$

```
GAME $L_1^{S,\overline{R}}(\lambda)$, $\boxed{L_2^{S,\overline{R}}(\lambda)}$
$L \leftarrow\!\!\$\; S^{\text{HASH}}(1^\lambda)$ ; $b' \leftarrow\!\!\$\; \overline{R}^{\text{HASHSIM}}(1^\lambda, L)$
Return $(b' = 0)$

HASH$(x, 1^\ell)$
$K \leftarrow$ RAND$(x,\ell)$ ; $U[K,\ell] \leftarrow x$ ; Return $\mathsf{RO}_1(x,\ell)$

RAND$(x,\ell)$
If $V[x,\ell] = \bot$ then $V[x,\ell] \leftarrow\!\!\$\; \{0,1\}^\lambda$
Return $V[x,\ell]$

HASHSIM$(u, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})$
$r \leftarrow\!\!\$\; \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$
If $(|u| \geq 2\lambda) \wedge (u[\lambda+1, 2\lambda] = 0^\lambda)$ then
    $\ell \leftarrow u[1,\lambda]$ ; $x \leftarrow u[2\lambda+1, |u|]$
    pad $\leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[\lambda+1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
    $K \leftarrow$ RAND$(x,\ell)$ ; $H[K,\ell] \leftarrow x$ ; Return $K \,\|\,$ pad
If $|u| = 3\lambda$ then
    $\ell \leftarrow u[1,\lambda]$ ; $i \leftarrow x[\lambda+1, 2\lambda]$ ; $K \leftarrow u[2\lambda+1, 3\lambda]$
    If $(i > \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil)$ then return $r$
    If $(H[K,\ell] \neq \bot)$ then
        Return BLOCK$(u, H[K,\ell], \ell, i)$
    Elsif $U[K,\ell] \neq \bot$ then
        $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{r \leftarrow \text{BLOCK}(u, U[K,\ell], \ell, i)}$
Return $r$

BLOCK$(u, x, \ell, i)$
pad $\leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[(\ell \bmod \overline{\mathsf{H}}.\mathsf{ol}(\lambda)) + 1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
$y \leftarrow \mathsf{RO}_1(x,\ell) \,\|\,$ pad
Return $y[(i-1)\cdot \overline{\mathsf{H}}.\mathsf{ol}(\lambda) + 1, i \cdot \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
```

```
GAME $L_3^{S,\overline{R}}(\lambda)$, $\boxed{L_4^{S,\overline{R}}(\lambda)}$
$L \leftarrow\!\!\$\; S^{\text{HASH}}(1^\lambda)$ ; $b' \leftarrow\!\!\$\; \overline{R}^{\text{HASHSIM}}(1^\lambda, L)$
Return $(b' = 0)$

HASH$(x, 1^\ell)$
Return $\mathsf{RO}_1(x,\ell)$

$\mathsf{RO}_1(x,\ell)$
$K \leftarrow$ RAND$(x,\ell)$
If $Z[x,\ell] = \bot$ then $Z[x,\ell] \leftarrow\!\!\$\; \{0,1\}^\ell$
Return $Z[x,\ell]$

RAND$(x,\ell)$
$K \leftarrow\!\!\$\; \{0,1\}^\lambda$
If $V[x,\ell] = \bot$ then
    If $K \in \mathsf{Dom}$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{K \leftarrow\!\!\$\; \{0,1\}^\lambda \backslash \mathsf{Dom}}$
    $\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{K\}$ ; $V[x,\ell] \leftarrow K$
Return $V[x,\ell]$

HASHSIM$(u, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})$
$r \leftarrow\!\!\$\; \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$
If $(|u| \geq 2\lambda) \wedge (u[\lambda+1, 2\lambda] = 0^\lambda)$ then
    pad $\leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[\lambda+1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$ ;
    $\ell \leftarrow u[1,\lambda]$ ; $x \leftarrow u[2\lambda+1, |u|]$ ; $K \leftarrow$ RAND$(x,\ell)$
    Return $K \,\|\,$ pad
If $|u| = 3\lambda$ then
    $\ell \leftarrow u[1,\lambda]$ ; $i \leftarrow x[\lambda+1, 2\lambda]$ ; $K \leftarrow u[2\lambda+1, 3\lambda]$
    $\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{K\}$
    If $(i > \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil)$ then return $r$
    If $(H[K,\ell] \neq \bot)$ then
        Return BLOCK$(u, H[K,\ell], \ell, i)$
Return $r$

BLOCK$(u, x, \ell, i)$
pad $\leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[(\ell \bmod \overline{\mathsf{H}}.\mathsf{ol}(\lambda)) + 1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
$y \leftarrow \mathsf{RO}_1(x,\ell) \,\|\,$ pad
Return $y[(i-1)\cdot \overline{\mathsf{H}}.\mathsf{ol}(\lambda) + 1, i \cdot \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
```

Figure 33: **Games L_1–L_4 for the proof of Theorem H.1.** Games L_2 and L_4 include the corresponding boxed statements but games L_1 and L_3 don't, and $\mathsf{RO}_1, \mathsf{RO}_2$ are independent random oracles.

Note that in game J_2, the answer to each HASHSIM query, generated by the second if-statement, is a fresh random string. To justify this, note that in the second if-statement, the return value is a $\overline{\mathsf{H}}.\mathsf{ol}$-bit block of $\mathsf{RO}_2(x,\ell)$ (with random padding if necessary). Since the strings K don't repeat, each K corresponds to at most one pair (ℓ, x). Hence the blocks we give to $\overline{R}$ never overlap, and thus are independent random strings. Game $J_3^{S,\overline{R}}$ makes this explicit; it's equivalent to game $J_2^{S,\overline{R}}$. Game $J_4^{S,\overline{R}}$ is identical to game $J_3^{S,\overline{R}}$, except that it forgoes the condition that the strings K are distinct, and thus HASHSIM always returns a fresh random string. Then

$$\Pr[J_3^{S,\overline{R}}(\lambda)] - \Pr[J_4^{S,R}(\lambda)] \leq \Pr[J_4^{S,\overline{R}}(\lambda) \text{ sets } \mathsf{bad}] \leq \frac{p^2}{2^{\lambda+1}}$$

for all $\lambda \in \mathbb{N}$. Game $J_5^{S,\overline{R}}$ simplifies procedure HASHSIM, and also unrolls the implementation of the variable-output-length random oracle $\mathsf{RO}_1(\cdot,\cdot)$ in HASH procedure, by querying a fixed-output-length

```
$\underline{\text{Game } L_5^{S,\overline{R}}(\lambda)}$
$L \leftarrow_\$ S^{\text{Hash}}(1^\lambda)$ ; $b' \leftarrow_\$ \overline{R}^{\text{HashSim}}(1^\lambda, L)$
Return $(b' = 0)$
$\underline{\text{Hash}(x, 1^\ell)}$
Return $\mathsf{RO}_1(x, \ell)$
$\underline{\mathsf{RO}_1(x, \ell)}$
$K \leftarrow \text{Rand}(x, \ell)$ ; $m \leftarrow \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$
For $i = 1$ to $m$ do $y_i \leftarrow \mathsf{RO}_2(\ell \,\|\, i \,\|\, K, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1, \ell]$
$\underline{\text{Rand}(x, \ell)}$
If $V[x, \ell] = \bot$ then
  $V[x, \ell] \leftarrow_\$ \{0,1\}^\lambda \backslash \mathsf{Dom}$ ; $\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{V[x, \ell]\}$
Return $V[x, \ell]$
$\underline{\text{HashSim}(u, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})}$
$r \leftarrow_\$ \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$
If $(|u| \geq 2\lambda) \wedge (u[\lambda + 1, 2\lambda] = 0^\lambda)$ then
  pad $\leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[\lambda + 1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
  $\ell \leftarrow u[1, \lambda]$ ; $x \leftarrow u[2\lambda + 1, |u|]$ ; $K \leftarrow \text{Rand}(x, \ell)$
  Return $K \,\|\,$ pad
If $|u| = 3\lambda$ then
  $\ell \leftarrow u[1, \lambda]$ ; $i \leftarrow x[\lambda + 1, 2\lambda]$ ; $K \leftarrow u[2\lambda + 1, 3\lambda]$
  $\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{K\}$
  If $(i > \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil)$ then return $r$
  If $(H[K, \ell] \neq \bot)$ then
    Return $\text{Block}(u, H[K, \ell], \ell, i)$
Return $r$
$\underline{\text{Block}(u, x, \ell, i)}$
pad $\leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[(\ell \bmod \overline{\mathsf{H}}.\mathsf{ol}(\lambda)) + 1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
$y \leftarrow \mathsf{RO}_1(x, \ell) \,\|\,$ pad
Return $y[(i - 1) \cdot \overline{\mathsf{H}}.\mathsf{ol}(\lambda) + 1, i \cdot \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
```

```
$\underline{\text{Game } L_6^{S,\overline{R}}(\lambda), \boxed{L_7^{S,\overline{R}}(\lambda)}}$
$L \leftarrow_\$ S^{\text{Hash}}(1^\lambda)$ ; $b' \leftarrow_\$ \overline{R}^{\text{HashSim}}(1^\lambda, L)$
Return $(b' = 0)$
$\underline{\text{Hash}(x, 1^\ell)}$
$K \leftarrow \text{Rand}(x, \ell)$ ; $m \leftarrow \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$
For $i = 1$ to $m$ do $y_i \leftarrow \mathsf{RO}_2(\ell \,\|\, i \,\|\, K, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1, \ell]$
$\underline{\text{Rand}(x, \ell)}$
$K \leftarrow_\$ \{0,1\}^\lambda$
If $V[x, \ell] = \bot$ then
  If $K \in \mathsf{Dom}$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{K \leftarrow_\$ \{0,1\}^\lambda \backslash \mathsf{Dom}}$
  $\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{K\}$ ; $V[x, \ell] \leftarrow K$
Return $V[x, \ell]$
$\underline{\text{HashSim}(u, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})}$
$r \leftarrow_\$ \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$
If $(|u| \geq 2\lambda) \wedge (u[\lambda + 1, 2\lambda] = 0^\lambda)$ then
  pad $\leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[\lambda + 1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$
  $\ell \leftarrow u[1, \lambda]$ ; $x \leftarrow u[2\lambda + 1, |u|]$ ; $K \leftarrow \text{Rand}(x, \ell)$
  Return $K \,\|\,$ pad
If $|u| = 3\lambda$ then
  $\ell \leftarrow u[1, \lambda]$ ; $i \leftarrow x[\lambda + 1, 2\lambda]$ ; $K \leftarrow u[2\lambda + 1, 3\lambda]$
  $\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{K\}$
  If $(i > \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil)$ then return $r$
  If $(H[K, \ell] \neq \bot)$ then return $\mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))$
Return $r$
```

Figure 34: **Games L_5–L_7 for the proof of Theorem H.1.** Games L_7 includes the corresponding boxed statement but game L_6 doesn't.

random oracle $\mathsf{RO}_3(\cdot, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))$. The queries to RO_3 are distinct, and thus the answers of Hash are still independent random strings. Hence $J_5^{S,\overline{R}}$ is equivalent to $J_4^{S,\overline{R}}$. Game $J_6^{S,\overline{R}}$ is identical to game $J_5^{S,\overline{R}}$, except that it modifies procedure Hash, choosing the strings K independently instead of making them distinct. The two games are identical-until-bad, and thus

$$\Pr[J_5^{S,\overline{R}}(\lambda)] - \Pr[J_6^{S,R}(\lambda)] \leq \Pr[J_5^{S,\overline{R}}(\lambda) \text{ sets } \mathsf{bad}] \leq \frac{q^2}{2^{\lambda+1}}$$

for all $\lambda \in \mathbb{N}$. Summing up,

$$\Pr[\,\text{Reset}_S^R(\lambda) \,|\, b = 0\,] \leq \Pr[\,\text{Reset}_{\overline{S}}^{\overline{R}}(\lambda) \,|\, \overline{b} = 0\,] + \frac{q^2 + p^2}{2^\lambda}$$

for all $\lambda \in \mathbb{N}$.

Consider games L_1–L_8 in Figures 33–35. We claim that game $L_1^{S,\overline{R}}$ is equivalent to game Reset_S^R with challenge bit $b = 1$. To justify this, note that $\overline{R}$ never repeats queries to HashSim, and thus we never repeat queries to RO_2. Hence calling pad $\leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[(\ell \bmod \overline{\mathsf{H}}.\mathsf{ol}(\lambda)) + 1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$ is effectively

```
Game $L_8^{S,\overline{R}}(\lambda)$
$L \leftarrow_{\$} S^{\text{Hash}}(1^\lambda)$ ; $b' \leftarrow_{\$} \overline{R}^{\text{HashSim}}(1^\lambda, L)$
Return $(b' = 0)$

Hash$(x, 1^\ell)$
$K \leftarrow \text{Rand}(x, \ell)$ ; $m \leftarrow \lceil \ell / \overline{\mathsf{H}}.\mathsf{ol}(\lambda) \rceil$
For $i = 1$ to $m$ do $y_i \leftarrow \mathsf{RO}_2(\ell \,\|\, i \,\|\, K, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))$
$y \leftarrow y_1 \,\|\, \cdots \,\|\, y_m$ ; Return $y[1, \ell]$

Rand$(x, \ell)$
Return $\mathsf{RO}_2(\ell \,\|\, 0^\lambda \,\|\, x, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[1, \lambda]$

HashSim$(u, 1^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)})$
$r \leftarrow_{\$} \{0,1\}^{\overline{\mathsf{H}}.\mathsf{ol}(\lambda)}$
If $(|u| \geq 2\lambda) \wedge (u[\lambda+1, 2\lambda] = 0^\lambda)$ then
    $\ell \leftarrow u[1, \lambda]$ ; $x \leftarrow u[2\lambda+1, |u|]$ ; $K \leftarrow \text{Rand}(x, \ell)$
    $v \leftarrow \mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))[\lambda+1, \overline{\mathsf{H}}.\mathsf{ol}(\lambda)]$ ; Return $K \,\|\, v$
If $|u| = 3\lambda$ then
    $\ell \leftarrow u[1, \lambda]$ ; $i \leftarrow x[\lambda+1, 2\lambda]$ ; $K \leftarrow u[2\lambda+1, 3\lambda]$
    $\mathsf{Dom} \leftarrow \mathsf{Dom} \cup \{K\}$
    If $(i > \ell) \vee (i \leq 0)$ then return $r$
    If $(H[K, \ell] \neq \bot)$ then return $\mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))$
```

Figure 35: **Game L_8 for the proof of Theorem H.1.**

sampling pad at random. Moreover, in HashSim, we never repeat queries to Rand, effectively sampling K at random. Finally, RO_2 calls from HashSim and Block are distinct: each call from Block has a nonzero second λ-bit block, whereas each call from HashSim has a zero second block. Game $L_2^{S,\overline{R}}$ is identical to game $L_1^{S,\overline{R}}$, except that if $\overline{R}$ happens to query $u = \ell \,\|\, i \,\|\, K$ such that (K, ℓ) was previously created in Hash(x, ℓ), then we'll return the i-th $\overline{\mathsf{H}}.\mathsf{ol}(\lambda)$-bit block of $\mathsf{RO}_1(x, \ell)$ (with random padding if necessary), instead of returning a random string. The two games are identical-until-bad, and thus for all $\lambda \in \mathbb{N}$, we have

$$\Pr[L_1^{S,\overline{R}}(\lambda)] - \Pr[L_2^{S,R}(\lambda)] \leq \Pr[L_1^{S,\overline{R}}(\lambda) \text{ sets } \mathsf{bad}] \leq \frac{pq}{2^\lambda} \ .$$

Game $L_3^{S,\overline{R}}$ is identical to game $L_2^{S,\overline{R}}$, except for the following changes. First, we add the code for RO_1. Next, both Hash and HashSim use the same array H, instead of keeping separate arrays H and U. The two games are equivalent, because in game $L_2^{S,\overline{R}}$, writes to H always happen after writes to U, and in its procedure HashSim, if both $H[K, \ell]$ and $U[K, \ell]$ are written then we'll process according to the value in $H[K, \ell]$. Game $L_4^{S,\overline{R}}$ is identical to game $L_3^{S,\overline{R}}$, except that the return values of Rand are now always distinct. Then

$$\Pr[L_3^{S,\overline{R}}(\lambda)] - \Pr[L_4^{S,R}(\lambda)] \leq \Pr[L_3^{S,\overline{R}}(\lambda) \text{ sets } \mathsf{bad}] \leq \frac{(p+q)^2}{2^{\lambda+1}}$$

for all $\lambda \in \mathbb{N}$. In game $L_5^{S,\overline{R}}$, we change the implementation of RO_1: we now implement it from CTR-mode on RO_2. Since Rand's outputs are distinct, RO_1 calls on different inputs never have a common RO_2 call. Moreover, while RO_1 and Block may make the same RO_2 call, they return non-overlapping parts of RO_2's outputs. Hence game $L_5^{S,\overline{R}}$ is equivalent to game $L_4^{S,\overline{R}}$. Note that due to the new implementation of RO_1, each Block in HashSim effectively calls $\mathsf{RO}_2(u, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))$. Game $L_6^{S,\overline{R}}$ makes this explicit; it is still equivalent to game $L_5^{S,\overline{R}}$. Game $L_7^{S,\overline{R}}$ modifies Rand, dropping the requirement that the outputs be distinct. Games L_6 and L_7 are identical-until-bad, and thus

$$\Pr[L_6^{S,\overline{R}}(\lambda)] - \Pr[L_7^{S,R}(\lambda)] \leq \Pr[L_7^{S,\overline{R}}(\lambda) \text{ sets } \mathsf{bad}] \leq \frac{(p+q)^2}{2^{\lambda+1}}$$

for all $\lambda \in \mathbb{N}$. In game $L_8^{S,\overline{R}}$, we change the implementation of Rand, returning the first λ bits of $\mathsf{RO}_2(\ell \,\|\, 0^\lambda \,\|\, x, \overline{\mathsf{H}}.\mathsf{ol}(\lambda))$. Note that the first if-branch of HashSim and Rand may make the same query to RO_2, but they use non-overlapping parts of the outputs. Then game $L_8^{S,\overline{R}}$ is equivalent to game $L_7^{S,\overline{R}}$. Summing up,

$$\Pr[\,\text{Reset}_{\overline{S}}^{\overline{R}}(\lambda) \,|\, \overline{b} = 1\,] \leq \Pr[\,\text{Reset}_S^R(\lambda) \,|\, b = 1\,] + \frac{(p+q)^2 + pq}{2^\lambda}$$

for all $\lambda \in \mathbb{N}$. Hence

$$\mathsf{Adv}^{\mathsf{reset}}_{\overline{R},\overline{S}}(\lambda) \leq \mathsf{Adv}^{\mathsf{reset}}_{S,R}(\lambda) + \frac{2(p+q)^2}{2^\lambda}$$

for all $\lambda \in \mathbb{N}$. ▌