Lower Bounds on Key Derivation for Square-Friendly Applications*

Maciej Skorski

IST, Klosterneuburg, Austria
mskorski@ist.ac.at

Abstract

Security of cryptographic applications is typically defined by security games. The adversary, within certain resources, cannot win with probability much better than 0 (for unpredictability applications, like one-way functions) or much better than $\frac{1}{2}$ (indistinguishability applications for instance encryption schemes). In so called *squared-friendly applications* the winning probability of the adversary, for different values of the application secret randomness, is not only close to 0 or $\frac{1}{2}$ on average, but also concentrated in the sense that its second central moment is small. The class of squared-friendly applications, which contains all unpredictability applications and many indistinguishability applications, is particularly important for key derivation. Barak et al. observed that for square-friendly applications one can beat the "RT-bound", extracting secure keys with significantly smaller entropy loss. In turn Dodis and Yu showed that in squared-friendly applications one can directly use a "weak" key, which has only high entropy, as a secure key.

In this paper we give sharp lower bounds on square security assuming security for "weak" keys. We show that *any* application which is either (a) secure with weak keys or (b) allows for entropy savings for keys derived by universal hashing, *must* be square-friendly. Quantitatively, our lower bounds match the positive results of Dodis and Yu and Barak et al. (TCC'13, CRYPTO'11) Hence, they can be understood as a general characterization of squared-friendly applications.

While the positive results on squared-friendly applications where derived by one clever application of the Cauchy-Schwarz Inequality, for tight lower bounds we need more machinery. In our approach we use convex optimization techniques and some theory of circular matrices.

1998 ACM Subject Classification F.2.3 Tradeoffs between Complexity Measures, G.1.6 [Optimization] Convex Programming

Keywords and phrases key derivation, square-friendly applications, lower bounds

Digital Object Identifier 10.4230/LIPIcs.STACS.2017.57

1 Introduction

When analyzing security of cryptographic primitives one typically assumes access to *perfect* randomness. In practice, we are often limited to *imperfect* sources of randomness. An important research problem is to understand when this "weak" randomness can be used to substitute or extract ideal randomness.

[*] Supported by the European Research Council consolidator grant (682815-TOCNeT). This is an extended abstract; the full version of the paper appears as ePrint report 2016/157 (https://eprint.iacr.org/2016/157).

34th Symposium on Theoretical Aspects of Computer Science (STACS 2017).
Editors: Heribert Vollmer and Brigitte Vallée; Article No. 57; pp. 57:1–57:12
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1.1 Key derivation

Ideal and real settings. For any cryptographic primitive (like encryption or signatures), which needs a "random" m-bit string R to sample the secure key[1], we compare two different settings:

(a) ideal setting: R is *perfectly* random: uniform and independent of any side information available to the attacker

(b) real settings: there is only an *imperfect* entropy source X and the secure key R needs to be derived from X. The attacker may have some *side information* about X, in particular the additional randomness used to derive R from X.

The security of the primitive is parametrized by ϵ, which is the success probability (for so called unpredictability applications) or the advantage (for so called indistinguishability applications) of an attacker with certain resources[2].

Generic approach and the entropy loss. The general way to derive a secure key is to "extract" the randomness from X by a seeded extractor. In particular, the Leftover Hash Lemma implies that if the min-entropy of X is at least $m + L$ then $H(X), H$, where H is randomly chosen function from a universal family, is δ-close to uniform with $\delta = \sqrt{2^{-L}}$. This means that if an application is ϵ-secure for uniform R, then the same application keyed with $R = H(X)$, and even published H, is ϵ'-secure with

$$\epsilon' \leqslant \epsilon + \sqrt{2^{-L}}. \tag{1}$$

where the entropy loss L is the difference between the entropy of X and m. Note that from Equation 1 it follows that we need $L = 2\log(1/\epsilon)$ to obtain (roughly) the same security $\epsilon' = 2\epsilon$. Unfortunately, if we want the security against *all statistical tests*, this loss is necessary for *any* extractor, as implied by the so called "RT-bound" [6].

Need for better techniques for cryptographic applications. The RT-bound does not exclude the possibility of deriving a secure key wasting *much less* than $2\log(1/\epsilon)$ bits of entropy for *particular* applications. Saving the entropy, apart from scientific curiosity, is a problem of real-word applications. Minimizing the entropy loss is of crucial importance for some applications where it affects efficiency (for example in the elliptic-curve Diffie-Hellman key exchange) and sometimes the entropy amount we have is bounded (e.g. biometric data) than the required length of a key; see also the discussion in [1]. Hence, better techniques than simple extracting are desired. Below we discuss what is known about possible improvements in key derivation for cryptographic applications.

Key derivation for unpredictability applications. It is known that unpredictability applications directly tolerate weak keys, provided that the entropy deficiency is not too big. More precisely, any unpredictability application which is ϵ-secure with the uniform m-bit key, is also $\epsilon' = 2^d\epsilon$-secure for any key of entropy $m - d$. If we have a source X that has "enough" entropy but its length is too big, we can condense it to a string of length m with almost full entropy. Essentially, since we achieve a very good condensing rate: any X of $m + \log\log(1/\epsilon)$ bits of entropy can be condensed to an m bit string with the entropy deficiency $d = 3$ which

[1] In applications like block-ciphers, R is the key itself. In other applications like RSA encryptions, R is used to sample public or secret keys. We will simply refer to R as the key.

[2] For example bounded running time, circuit size or the number of oracle queries.

is $\epsilon' = 2^3\epsilon$-close to uniform[3], we are able to derive a key (roughly) equally secure as the uniform key, with the entropy loss only $L = O(\log\log(1/\epsilon))$, i.e. actually without entropy waste [4].

Key derivation for indistinguishability applications. The situation for indistinguishability applications is completely different. For the one-time pad which needs an m-bit uniform key, a key of even $m-1$ bits of entropy might be insecure[1]. For some applications we can overcome this difficulty if the winning probability of the adversary, as a function of the key, is not only close to $1/2$ on average, but also *concentrated* around $1/2$. Recall that the advantage of an attacker, for a particular key, is defined as the difference[4] between the winning probability and $1/2$. One introduces the following two interesting properties:
(a) strong security: the *absolute* advantage is small on average (close to the advantage)
(b) square security: the *squared* advantage is small on average (close to the advantage)
Property (a) provides basically the same bounds as for the unpredictability applications. Namely, we can apply a weak key directly, losing a factor 2^d in the security where d is the entropy deficiency. Unfortunately, this holds only for a very limited class of applications. Property (b) offers slightly worse bounds but is satisfied for a wide class of indistinguishability applications, called "squared-friendly". One can use a "weak key" *directly* with a squared-friendly application achieving security of roughly $\sqrt{2^d\epsilon}$ where ϵ is the security with the uniform key and d is the entropy deficiency [5]. Alternatively, if we want to obtain security $O(\epsilon)$ instead of $O(\sqrt{\epsilon})$, one can use universal hashing to extract an ϵ-secure key with the entropy loss reduced by half[1] , i.e. up to $L = \log(1/\epsilon)$. The improvement in the security analysis over the "standard" Leftover Hash Lemma (LHL) comes from restrictions on the class of the test functions, imposed by the squared-friendly assumption.

1.2 Our results

In what follows we assume that P is an arbitrary indistinguishability application which needs an m-bit uniform key. We give *tight* lower bounds on the amount of square security (the expected square of the attacker's advantage) or strong-security (the expected absolute average of the attacker's advantage) that is necessary for an application to be secure with weak keys, that is keys with entropy deficiency. The notion of entropy here is either the min-entropy or the collision entropy. Collision entropy is less restrictive than min-entropy and is a natural choice to applications involving hash functions, like the LHL[5]. It is equally good for squared-friendly applications as min-entropy. Therefore, as remarked in [5], results for collision entropy are more desired. Nevertheless, we provide bounds for both entropy notions[6].

Summary of our contribution. We characterize squared-friendly applications by their "nice" features. Namely, we show that square-friendly applications are *precisely* those applications which are secure with weak keys or offers improvements in the entropy loss for a key derived

[3] Thus, for condensing we lose incomparably less in the amount than for extracting.

[4] In indistinguishability games, an adversary needs to guess a bit at the end of the game. Since he can flip his answer, any bias indicates that his guess is better than a random answer.

[5] For some applications we may prefer LHL over other extractors because of its simplicity, efficiency and nice algebraic features[1].

[6] Actually collision entropy is more challenging and our observations on strong security are known in folklore, but we study also the min-entropy case for the sake of completeness.

by the LHL. Hence the current state of art is optimal: we cannot do better key derivation than for squared-friendly applications unless we build a theory on stronger than collision entropy requirements for weak keys (which would be in some sense inconvenient because of a natural connection between collision entropy and hash functions).

Any application secure with weak keys has large square-security. The following results was proved by Dodis and Yu:

▶ **Theorem** ([5]). *Applications which are σ-square-secure with the uniform key, i.e. when the averaged squared advantage of any attacker is less than σ, are $\epsilon = \sqrt{2^d\sigma}$-secure with any key of collision entropy at least $m - d$.*

The following question is therefore natural

> **Q**: If P is secure for all keys of *high (collision or min-) entropy*, how much square-security does it have?

We give an answer in the following two theorems. The first is actually trivial and perhaps known in folklore.

▶ **Theorem.** *Let $d \geqslant 1$. Suppose that P is ϵ-secure with any key of min-entropy at least $m - d$. Then P is ϵ'-strongly secure with $\epsilon' = O(\epsilon)$.*

The second one is more interesting

▶ **Theorem** (Informal). *Let $d \geqslant 1$. If P is ϵ-secure with any key of collision entropy at least $m - d$, then P is σ-square-secure with $\sigma = O(\epsilon^2)$.*

The bounds in both cases are tight. Note that if the entropy deficiency d is bounded then our lower bound perfectly (up to a constant factor) matches the result of Dodis and Yu for *any P*.

Square Security is necessary to improve key derivation by condensing collision entropy. In the previous paragraph, we discussed the case when the entropy deficiency d is bounded away from 0. However, sometimes we intentionally extremely condense collision entropy so that this gap is close to 0, to achieve better than $O(\sqrt{\epsilon})$ security at the price of starting with more than m-bits of entropy. For ϵ-secure square friendly applications one can derive by universal hashing a (roughly) ϵ-secure key from any source having $m + \log(1/\epsilon)$ bits of min-entropy (or even collision entropy) [1]. Let us briefly discuss this result. The proof of the classical Leftover Hash lemma consists of two separate claims:

(a) *Universal hash functions can extremely condense collision entropy.*
(b) *Distributions of extremely high collision entropy are close to uniform.*

More precisely, in the first step one applies a random function from a universal family to "condense" the collision entropy of X from $m + L$ bits, where $L \gg 0$, to an m-bit string with $m - \log(1 + 2^{-L}) \approx m - 2^{-L}$ bits of collision entropy [7]. In the next step one shows that any m-bit random variable with collision entropy at least $m - \epsilon^2$ is ϵ-close to uniform. Thus, $L = 2\log(1/\epsilon)$ is enough to obtain ϵ-security. As observed by Barak et al. [1], for ϵ-secure applications which are *in addition* ϵ-square-secure, it suffices to start with $m - \epsilon$ bits of the collision entropy in step (b), which reduces by half, i.e. up to $L = \log(1/\epsilon)$ (comparing to the RT-bound), the entropy loss needed to achieve ϵ-closeness.

[7] Conditioned on the choice of the function, which can be thought as a *seed* for the condenser.

▶ **Theorem** ([1]). *Suppose that P with a uniform m-bit key is ϵ-secure and σ-square secure. Let R be any key of collision entropy at least $m-d$ (possibly given some side information). Then P keyed with R is ϵ'-secure with $\epsilon' = \epsilon + \sqrt{\sigma(2^d-1)}$, even if the used hash function is published. In particular, for $\sigma = O(\epsilon)$ and $d = O(\epsilon)$ we obtain $\epsilon' = O(\epsilon)$.*

Applying this to R being X condensed by universal hashing we get

▶ **Corollary** ([1]). *Suppose that P with a uniform m-bit key is ϵ-secure and σ-square secure (that is, average squared advantage of attackers is not bigger than σ). Suppose that X has min-entropy (or collision entropy) at least $m+L$ and let R be an m-bit key derived by universal hashing. Then P keyed with R is ϵ'-secure with $\epsilon' = \epsilon + \sqrt{2^{-L}\sigma}$, even if the used hash function is published. In particular, $\epsilon' = O(\epsilon)$ for $\sigma = O(\epsilon)$ and $L = \log(1/\epsilon)$.*

The first result motivates the following question about weak keys with the entropy deficiency close to 0.

> **Q**: Suppose that an application P is secure for all keys of *extremely condensed collision entropy*, possibly given side information. How much square-security does P have?

We give an answer in the following theorem

▶ **Theorem** (Informal). *Let $d \ll 1$ and suppose that P is ϵ-secure with all keys of collision entropy at least $m-d$ (possibly given side information, like the condenser's seed). Then P is σ-square secure with $\sigma = O(\max(d, \epsilon^2/d))$.*

Our theorem, applied for $d=\epsilon$, shows the full converse of the observation of Barak et al. A good illustrative example is the case of the Leftover Hash Lemma. As mentioned, universal hash functions condense $m+\log(1/\epsilon)$ bits of entropy into an m-bit string with $m-\epsilon$ bits of entropy. If we use universal hash functions *only as a condenser* (which is exactly how we use them in the LHL), then we have a "black-box" equivalence between distributions of collision entropy at least $m-\epsilon$ and hashes of distributions having at least $m+\log(1/\epsilon)$ bits of entropy[8]. If follows then that we want to reduce the entropy loss by half to $L = \log(1/\epsilon)$ and achieve ϵ-security, then our application must be ϵ-square secure. This lower bound matches the positive result of Barak et al. [1] and, since is holds for *any* application, can be viewed as a *general* characterization of squared-friendly applications.

Square security is necessary for reducing the entropy loss in the LHL. As remarked in the discussion in the previous paragraph, we can *heuristically* identify the set of randomly "hashed" high entropy distributions with the set of distributions of extremely high collision entropy (conditioned on the choice of a hash function as the uniform "seed"). This "equivalence", is reasonable for the "black-box" use of hash functions. However, it is natural to ask if we can prove it formally. That is, we ask if square security is necessary for improvements in the entropy loss for key derived not by a general "black-box" collision entropy condenser but precisely by *hashing*.

> **Q**: Suppose that an application P is secure for any key derived by applying a randomly chosen (almost) universal hash function to a min-entropy source, even if the hash function is published. Suppose that the entropy loss vs security trade-off is significantly better than (pessimistic) RT-bound. Is P square-secure?

[8] Because the only information that a general condenser provides is about the entropy in its output.

We give an affirmative answer and a lower bound that (almost) matches the results of [1] for *any* application.

▶ **Theorem** (The improved LHL [1] is tight for any application, informal). *Let $\epsilon = 2^{-(1-\beta)m}$ where β is some small positive number. Then there exists an ϵ-universal family $\mathcal{H}$ of hash functions from n to m bits, efficiently commutable and samplable with the use of n^2 uniformly random bits, with the following property: for any application P, if for every source X of min-entropy at least $k = m + \log(1/\epsilon)$ and H chosen randomly from $\mathcal{H}$ we have that P is secure with the key $H(X)$ and published H, then P must be σ-square-secure with $\sigma = \epsilon^{\frac{1-3\beta/2}{1-\beta}}$.*

This theorem for β close to 1 (exponential but meaningful security) shows that $\epsilon^{1-o(1)}$-square-security is *necessary* for saving $\log(1/\epsilon)$ bits of entropy in deriving an ϵ-secure key by universal hashing (which is almost tight since ϵ-square-security is enough.

Square-security bounds are generally optimal. The improved bound for applications which are square-secure is generally optimal, as observed in [4]. We provide an alternative proof of this result, using our techniques.

▶ **Theorem 1** (Square-friendly bounds are generally optimal [4]). *For any n, $k \leqslant n$, and $\delta \in (0,1)$ there exists an application which has δ-square security but for some key of Renyi entropy at least k it achieves only security $\epsilon = \Omega\left(\sqrt{2^{n-k}\delta}\right)$.*

The proof appears in the full version. The advantage of our proof is that it abstracts a general condition for this bound to be satisfied. In particular, similar bounds can obtained for all so called "strongly secure" applications, where attacker's advantage is nearly zero except a tiny subset of "weak" keys where the attacker wins with heavy advantage.

1.3 Our techniques

Our main technical contribution is an explicit characterization of a distribution which maximizes the expectation of a given function, subject to collision entropy constraints. We show that the worst-case distribution has the shape similar to the function, up to a transform which involves taking a threshold and scaling, as illustrated in Figure 1. We apply this characterization to settings where we want to find the distribution of keys which maximizes the attacker advantage. We stress that with our characterization one can compute *optimal* security with weak keys. Previous works [5, 1] obtained good bounds with the Cauchy-Schwarz inequality only, however these techniques cannot be extended to obtain optimal or lower bounds, as we do.

1.4 Organization of the paper

In Section 2 we provide the basic notations and definitions for security, square-security and entropy. In Section 3 we discuss the known positive result. Our key auxiliary result on optimization problems with collision entropy constraints is presented in Section 4. The lower bounds are given in Section 5.

2 Preliminaries

Basic notions. The min entropy of X is $\mathbf{H}_\infty(X) = \log(1/\max_x \Pr[X = x])$. The collision probability of X is $\mathrm{CP}(X) = \sum_x \Pr[X = x]^2$, that is $\mathrm{CP}(X) = \Pr[X = X']$ where X' is an

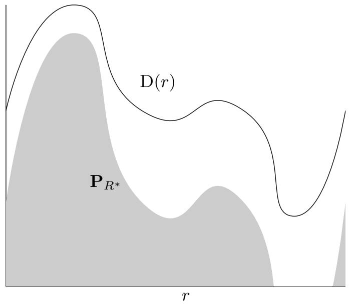

Figure 1 The shape of the best-advantage key distribution (under collision entropy constraints). In application the function $\mathrm{D}(r)$ equals the advantage of an attacker on the key r.

independent copy of X. The collision entropy of X is $\mathbf{H}_2 = \mathrm{CP}(X)$ and the conditional collision entropy $\mathbf{H}_2(X|Z)$ equals $-\log\left(\mathbf{E}_{z\leftarrow Z}\,\mathrm{CP}(X|_{Z=z})\right)$. The statistical distance of X and Y (taking values in the same space) is $\Delta(X;Y) = \sum_x |\Pr[X=x] - \Pr[Y=x]|$.

Security of indistinugishability and unpredictability applications. Consider any application whose security is defined by a *security game* between an attacker A and a challenger $\mathsf{C}(r)$, where r is an m-bit key derived from U_m in the "ideal" setting and from some distribution R in the "real" setting. For every key r we denote by $\mathsf{Win}_{\mathsf{A}}(r)$ the probability (over the randomness used by A and C) that the adversary A wins the game when challenged on the key r. The advantage of the adversary A on the key r is defined, depending on the type of the application (unpredictability, indistinugishability) as follows:

$$\mathsf{Adv}_{\mathsf{A}}(r) \overset{\text{def}}{=} \mathsf{Win}_{\mathsf{A}}(r), \qquad \text{(unpredictability)} \qquad (2)$$

$$\mathsf{Adv}_{\mathsf{A}}(r) \overset{\text{def}}{=} \mathsf{Win}_{\mathsf{A}}(r) - \frac{1}{2}, \qquad \text{(indistingusihability)} \qquad (3)$$

Now we define the security in the ideal and real models as follows:

▶ **Definition 2** (Security in the ideal and real model). An application $\mathcal{P}$ is (T,ϵ)-secure in the ideal model if

$$\left|\mathop{\mathbf{E}}_{r\leftarrow U_m} \mathsf{Adv}_{\mathsf{A}}(r)\right| \leqslant \epsilon \qquad (4)$$

for all attackers A with resources less than T. We say that $\mathcal{P}$ is (T,ϵ)-secure in the $(m-d)$-real$_2$ if for every distribution R of collision entropy at least $m-d$

$$\left|\mathop{\mathbf{E}}_{r\leftarrow R} \mathsf{Adv}_{\mathsf{A}}(r)\right| \leqslant \epsilon, \qquad (5)$$

for all attackers A with resources less than T.

▶ Remark (Strong security in the ideal model). If $\mathbf{E}_{r\leftarrow U_m} |\mathsf{Adv}_{\mathsf{A}}(r)| \leqslant \epsilon$ in the above setting then we say that $\mathcal{P}$ is (T,ϵ)-strongly secure (in the ideal model).

Square security. Finally, we define the notion of square-security (in the ideal model)

▶ **Definition 3** (Square security). An application $\mathcal{P}$ is (T,ϵ)-square-secure if

$$\mathop{\mathbf{E}}_{r\leftarrow U_m} \mathsf{Adv}_{\mathsf{A}}(r)^2 \leqslant \epsilon, \qquad (6)$$

for all attackers A with resources less than T.

Security in the presence of side information. Sometimes we need to consider stronger adversaries, which has additional information S. For example, this is always the case where the weak key has been derived from an entropy source using *public* randomness.

▶ **Definition 4** (Security in the presence of side information)**.** Given a side information $S \in \mathcal{S}$, an application $\mathcal{P}$ is (T,ϵ)-secure in the $(m-d)$-real$_2$ model if for every distribution R such that $\mathbf{H}_2(R|S) \geqslant m-d$ we have

$$\max_{s\in\mathcal{S}} |\mathop{\mathbf{E}}_{r\leftarrow R} \mathsf{Adv}_{\mathsf{A}}(r,s)| \leqslant \epsilon, \tag{7}$$

for all attackers A with resources less than T. In the ideal model $\mathcal{P}$ is (T,ϵ)-secure if $\max_{s\in\mathcal{S}} |\mathbf{E}_{r\leftarrow U_m} \mathsf{Adv}_{\mathsf{A}}(r,s)| \leqslant \epsilon$ and (T,ϵ)-square-secure if $\max_{s\in\mathcal{S}} |\mathbf{E}_{r\leftarrow U_m} \mathsf{Adv}_{\mathsf{A}}(r,s)^2| \leqslant \epsilon$ for all attackers A with resources less than T.

▶ Remark. Note that in the nonuniform setting, security and square security in the ideal model with and without side information coincide.

3 Square security – positive results

Improved key derivation for square-secure applications. Let D be an arbitrary real-valued function on $\{0,1\}^m$ and let Y be an arbitrary m-bit random variable with collision entropy $\mathbf{H}_2(X) \geqslant m-d$. By the Cauchy Schwarz Inequality one obtains [5, 1] the following inequalities

$$\mathbf{E}\,\mathrm{D}(Y) \leqslant \sqrt{\mathbf{E}\,\mathrm{D}(U_m)^2} \cdot \sqrt{2^d}, \tag{8}$$

$$\mathbf{E}\,\mathrm{D}(Y) - \mathbf{E}\,\mathrm{D}(U_m) \leqslant \sqrt{\mathrm{Var}\mathrm{D}(U_m)} \cdot \sqrt{2^d-1}. \tag{9}$$

When the side information S is present, and $\mathbf{H}_2(Y|S) \geqslant m-d$, we get

$$\mathbf{E}\,\mathrm{D}(Y,S) \leqslant \sqrt{\mathbf{E}\,\mathrm{D}(U_m,S)^2} \cdot \sqrt{2^d}, \tag{10}$$

$$\mathbf{E}\,\mathrm{D}(Y) - \mathbf{E}\,\mathrm{D}(U_{\mathcal{Y}}) \leqslant \sqrt{\mathop{\mathbf{E}}_{s\leftarrow S} \mathrm{Var}\mathrm{D}(U_m,s)} \cdot \sqrt{2^d-1}. \tag{11}$$

These inequalities, applied to $\mathrm{D} = \mathsf{Adv}_{\mathsf{A}}$ link the security in the real model with the entropy deficiency of a weak key and the security in the ideal model. In particular, one obtains the following results, already mentioned in Section 1.1.

▶ **Theorem 5** ([5])**.** *Suppose that $\mathcal{P}$ is (T,σ)-square secure in the ideal model. Then it is (T,ϵ) secure in the $(m-d)$-real$_2$ model with $\epsilon = \sqrt{2^d\sigma}$.*

▶ **Theorem 6** ([1, 5])**.** *Suppose that an application $\mathcal{P}$ in the ideal model is (T,ϵ)-secure and (T,σ)-square-secure. Then it is (T,δ) secure in the real $(m-d)$-model with $\delta = \epsilon + \sqrt{(2^d-1)\sigma}$.*

Theorem 5 states that a weak key can be used directly in a square-secure application provided that the entropy deficiency is not too big. The second theorem deals with the case where the deficiency is *extremely* small. It is essentially important when one notices that *universal hash functions* condense collision entropy at a very good rate. Theorem 6 yields the following important corollary

▶ **Corollary 7** (Improved LHL, [1])**.** *Suppose that $\mathcal{P}$ is as above. Let X be an n-bit random variable of collision entropy at least $m+L$, let $\mathcal{H}$ be a $\frac{1+\gamma}{2^m}$-universal family of functions from n to m bits and let H be a random member of $\mathcal{H}$. Then $\mathcal{P}$ keyed with $H(X)$ is ϵ'-secure with*

$\epsilon' \leqslant \epsilon + \sqrt{\sigma\left(\gamma + 2^{-L}\right)}$ against all adversaries with resources T and given H. In other words, for all A with resources T we have

$$\mathop{\mathbf{E}}_{(r,h)\leftarrow H(X),H} \mathsf{Adv}_{\mathsf{A}}(r,h) \leqslant \epsilon + \sqrt{\sigma\left(\gamma + 2^{-L}\right)}. \tag{12}$$

In particular, for $\gamma \leqslant \epsilon$ and $\sigma \leqslant 4\epsilon$ we achieve security $\epsilon' \leqslant 3\epsilon$ with only Ł $= \log(1/\epsilon)$ bits of the entropy loss.

Summing up, when we want to derive a secure key for an ϵ-square-secure application from a source X, we have two options

(a) We condense (if necessary) X by hashing into a string with small entropy deficiency. From a source which has $m - O(1)$ bits of entropy we derive a $O(\sqrt{\epsilon})$-secure key.

(b) If we want more security, we can condense X even stronger, with deficiency extremely close to 0, sacrificing some entropy amount. From a source which has $m + \log(1/\epsilon) - O(1)$ bits of entropy we derive a $O(\epsilon)$-secure key.

In every case we obtain the meaningful security, in particular even if entropy amount we start with is *smaller* than the length of the key we need. The application of a generic extractor in such a case gives *no* security guarantee! For more examples and applications we refer the reader to [5] and [1].

Security and square security – mathematical insight. It is worth of mentioning that the idea behind square security is, conceptually, simple and natural. All we need is the *concentration* of the adversary's winning probability, which is guaranteed by the small first central or second central moment.

What applications are square-secure? It is known that PRGs, PRFs and one-time pads cannot have good square security[2]. In turn, many applications such as such as stateless chosen plaintext attack (CPA) secure encryption and weak pseudo-random functions (weak PRFs), can be proven to be "square-friendly" that is they have square-security roughly the same as the standard security. The general method to prove that security implies square-security is the so called "double run trick" [1, 5].

4 Optimization: auxiliary results

Our main technical tool is a characterization of a distribution that maximizes the expectation of a given function under the collision entropy constraints. It has a nice geometrical interpretation, as the best shape is simply a combination of a threshold and scaling transformation, see Figure 1.

▶ **Lemma 8** (Maximizing the expectation subject to collision entropy constraints). *Let* $\mathrm{D} : S \to [0,1]$ *be a function on a finite set* S *and let* Y^* *be any optimal solution to the following problem*

$$\begin{aligned} &\underset{Y}{\text{maximize}} && \mathbf{E}\,\mathrm{D}(Y) \\ &\text{subject to} && \mathbf{H}_2(Y) \geqslant k \end{aligned} \tag{13}$$

where the maximum is taken over all random variables Y taking values in S. Then there exist numbers $\lambda \geqslant 0$ and $t \in \mathbb{R}$ such that Y^* satisfies the following condition

$$\max(\mathrm{D}(x) - t, 0) = \lambda \mathbf{P}_{Y^*}(x) \quad \textit{for all } x \in S. \tag{14}$$

In particular $\mathrm{Var}\mathrm{D}'(U) = \frac{\lambda^2}{|S|^2}$ *where* $\mathrm{D}'(x) = \max(\mathrm{D}(x) - t, 0)$. *Moreover, if values of* $\mathrm{D}(\cdot)$ *are all different, then we have* $\lambda > 0$ *and* λ, t *are unique.*

▶ **Remark.** If values of $\mathrm{D}(\cdot)$ are all different, then $\lambda > 0$ (see Appendix A).

▶ **Corollary 9.** *We have the following identities* $\mathbf{E}\,\mathrm{D}'(U_m) = \frac{\lambda}{|S|}$, $\mathbf{E}\,\mathrm{D}'(U_m)^2 = \frac{1+\theta}{|S|^2\lambda^2}$, *and* $\mathbf{E}\,\mathrm{D}'(Y^*) = \frac{(1+\theta)\lambda}{|S|}$ *(*D, θ, Y^*, λ, t *and* D' *are as in Theorem 8).*

5 Square security – lower bounds

5.1 Weak keys with the entropy deficiency bounded away from 0

We start with the following results, which states that every indistinguishability application which is secure with all keys of high min-entropy must be *strongly secure*. The proof is relatively easy and appears in the full version.

▶ **Theorem 10.** *Suppose that an indistinguishability application* P*, which needs an* m*-bit key, is* (T, ϵ)*-secure in the* $(m - d)$*-real*$_\infty$ *model, for some* $d \geqslant 1$*. Then* P *is* $(T, 2\epsilon)$*-strongly secure. The bound* 2ϵ *here is tight.*

More challenging and more interesting is the case of an application secure with all keys of high collision entropy.

▶ **Theorem 11.** *Suppose that an indistinguishability application* P*, which needs an* m*-bit key, is* (T, ϵ)*-secure in the* $(m - d)$*-real*$_2$ *model, for some* $d \geqslant 1$*. Then* P *is* (T, σ)*-square-secure with* $\sigma = 4\epsilon^2$

Note that for bounded d the level of square security perfectly matches the positive result of Dodis and Yu (Theorem 5). We also show (see the end remark in the proof) that this bound is tight up to a constant factor and thus we cannot get the bound $O\left(\epsilon^2/2^d\right)$, which would exactly match to the positive result in Theorem 5 for all d. The proof is heavily based on Theorem 8 and appears in the full version.

5.2 Weak keys with the entropy deficiency close to 0

Below we provide a lower bound when the entropy deficiency is close to 0.

▶ **Theorem 12.** *Suppose that* P*, which uses an* m*-bit key, is* (T, ϵ)*-secure in the* $(m - d)$*-real*$_2$ *model (possibly with side information). Then* P *is* σ*-square-secure with* $\sigma \leqslant \epsilon^2 + \max\left(2^d - 1, \frac{4\epsilon^2}{2^d - 1}\right)$*. In particular, if* $d \ll 1$ *then* $\sigma \leqslant 2\max(d, \frac{\epsilon^2}{d})$*.*

The proof is based on Theorem 8 and appears in the full version. From this we see that Theorem 6 for $d = \epsilon$ is tight.

5.3 Leftover Hash Lemma as a Key Derivation Function

Finally, we consider the case of a key derived by hashing.

▶ **Theorem 13.** *Let* $\alpha \in [1, 2]$ *and let* $\epsilon > 0$*. Suppose that an application* P*, which uses an* m*-bit secure key, has the following property: for every* n*-bit source* X *of min-entropy* $k \geqslant m + \alpha \log(1/\epsilon)$*, and every efficient* ϵ^α*-universal family* $\mathcal{H}$ *of hash functions from* n *to* m *bits, we have*

$$\mathbf{E}\,\mathsf{Adv}_{\mathsf{A}}(H(X), H) \leqslant C\epsilon,$$

for some constant C and all adversaries A with resources at most T. Then P is (T,σ)-square-secure with

$$\sigma \leqslant \frac{3}{2}\cdot \max\left(2^{-m/2}\epsilon^{\alpha/2}, 4(C+1)^2 2^{m/2}\epsilon^{2-\alpha/2}\right). \tag{15}$$

For $\epsilon > 2^{-m}$ we get $\sigma = O\left(2^{m/2}\epsilon^{2-\alpha/2}\right)$. In particular, if $\alpha = 1$ and $\epsilon = 2^{-(1-\beta)m}$ for some $\beta > 0$ then $\sigma = O\left(2^{-(1-3\beta/2)m}\right) = O\left(\epsilon^{\frac{1-3\beta/2}{1-\beta}}\right)$. Thus, *any* application P which allows deriving an ϵ'-secure key with $\epsilon' = O(\epsilon)$ and entropy loss $L = \log(1/\epsilon)$ must be $\sigma = O\left(\epsilon^{1-o(1)}\right)$-square-secure. On the positive side we know that σ-square-security with $\sigma = \epsilon$ is enough (Theorem 7).

▶ **Corollary 14** (The Improved LHL is tight for any application). *For* any *application P, the security guarantee in the improved Leftover Hash Lemma (Theorem 7) cannot be improved by more than a factor $\epsilon^{o(1)}$. Note that we require $\mathcal{H}$ to be efficiently computable and samplable, in order to exclude some (possible) "pathological" counterexamples.*

The proof of Theorem 13 relies on some advanced facts from matrices theory. We briefly sketch our approach, the full proof appears in the full version. The key technical fact we prove is that the hashes of high-min-entropy distributions are really mapped *onto* high collision entropy distributions (with quantitative parameters good enough for our purposes). Once we have a such a correspondence, we reduce the problem to Theorem 12. To this end, we consider the probability $\Pr[H(x) = y]$ that x is hashed into y as a *matrix* with rows y and columns x and observe use this matrix to obtain a linear map which realizes that correspondence. To obtain a map with a good behavior, we fill it using some special "pattern" which ensures nice algebraic properties and simplifies inverting.

6 Conclusion

We show that the technical condition called *square security* introduced in previous works of Dodis,Yu (TCC'13) and Barak et al. (CRYPTO'11), is not only sufficient but also necessary for better security with weak keys used directly.

References

1 B. Barak, Y. Dodis, H. Krawczyk, O. Pereira, K. Pietrzak, F. Standaert, and Yu Yu. Leftover hash lemma, revisited. In *Proc. 31th CRYPTO*, 2011.

2 B. Barak, R. Shaltiel, and A. Wigderson. Computational analogues of entropy. In *RANDOM-APPROX*, 2003.

3 Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

4 Y. Dodis, K. Pietrzak, and D. Wichs. Key derivation without entropy waste. In *EUROCRYPT*, pages 93–110. Springer Berlin Heidelberg, 2014. doi:10.1007/978-3-642-55220-5_6.

5 Y. Dodis and Yu Yu. Overcoming weak expectations. In *Theory of Cryptography*, volume 7785 of *Lecture Notes in Computer Science*. Springer, 2013. doi:10.1007/978-3-642-36594-2_1.

6 J. Radhakrishnan and A. Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM JOURNAL ON DISCRETE MATHEMATICS*, 13:2000, 2000.

A Proof of Theorem 8

Proof. Our problem is equivalent to the following constrained maximization problem over $\mathbb{R}^{|S|}$

$$\begin{aligned} \underset{(p(x))_x\in\mathbb{R}^{|S|}}{\text{maximize}} \quad & \sum_x \mathrm{D}(x)p(x) \\ \text{subject to} \quad & -p(x) \leqslant 0 \quad \text{for all } x\in S \\ & \sum_x p(x) = 1 \\ & \sum_x p(x)^2 \leqslant 2^{-k} \end{aligned} \tag{16}$$

The corresponding Lagrangian is given by

$$\begin{aligned} L((p(x))_x;(\lambda_1(x))_x,\lambda_2,\lambda_3) = & \sum_x \mathrm{D}(x)p(x) + \sum_x \lambda_1(x)p(x) - \lambda_2\left(\sum_x p(x) - 1\right) \\ & - \lambda_3\left(\sum_x p(x)^2 - 2^{-k}\right) \end{aligned} \tag{17}$$

Note that the equality constraint is linear, the inequality constraints are convex and, since $k < n$, there exists a vector $p = p(x)$ such that $p(x) \geqslant 0$ for all x, $\sum_x p(x) = 1$ and $\sum_x p(x)^2 < 2^k$. This means that Slater's Constraint Qualification is satisfied and the strong duality holds [3]. In this case the Karush-Kuhn-Tucker (KKT) conditions imply that for the optimal solution $p = p^*$ we have

$$\mathrm{D}(x) = -\lambda_1(x) + \lambda_2 + \lambda_3 p^*(x) \tag{18}$$

where $\lambda_1(x) \geqslant 0$ for all x, $\lambda_3 \geqslant 0$ and $\lambda_2 \in \mathbb{R}$ are the Lagrange Multipliers, satisfying the following so called "complementary slackness" condition

$$\begin{aligned} \forall x \quad \lambda_1(x) = 0 \qquad & \text{if } p^*(x) > 0, \\ \lambda_3 = 0 \qquad & \text{if } \textstyle\sum_x (p^*(x))^2 < 2^{-k}. \end{aligned} \tag{19}$$

The characterization in Equation 14 follows now by setting $\lambda = \lambda_3$ and $t = \lambda_2$. Indeed, by Equation 18, Equation 19 and $\lambda_1(x) \geqslant 0$ we get

$$\max(\mathrm{D}(x) - \lambda_2, 0) = \max(-\lambda_1(x) + \lambda_3 p^*(x), 0) = \lambda_3 p^*(x).$$

Finally, note that if all values of $\mathrm{D}(\cdot)$ are different then in Equation 18 we cannot have $\lambda_3 = 0$, because then Equation 19 implies that D is constant on the support of p^* (which has at least two points provided that $k > 0$). To proof the uniqueness part, observe that if there exists a different pair (t', λ') for the same optimal solution p^*, then for all x such that $p^*(x) > 0$ we have

$$\forall x \in \mathrm{supp}(p^*) \quad \mathrm{D}(x) = t + \lambda p^*(x) + t' = \lambda' p^*(x), \tag{20}$$

and, since the case $\lambda = \lambda'$ cannot happen because it implies $t = t'$, we get $p^*(x) = \frac{t-t'}{\lambda-\lambda'}$. ◀