The proceedings version of this paper appears in CRYPTO 2017. This is the full version.

A New Distribution-Sensitive Secure Sketch and Popularity-Proportional Hashing

Joanne Woodage[1], Rahul Chatterjee[2], Yevgeniy Dodis[3], Ari Juels[2], Thomas Ristenpart[2]

[1] Royal Holloway, University of London

[2] Cornell Tech

[3] New York University

Abstract. Motivated by typo correction in password authentication, we investigate cryptographic error-correction of secrets in settings where the distribution of secrets is a priori (approximately) known. We refer to this as the distribution-sensitive setting.

We design a new secure sketch called the layer-hiding hash (LHH) that offers the best security to date. Roughly speaking, we show that LHH saves an additional $\log H_0(W)$ bits of entropy compared to the recent layered sketch construction due to Fuller, Reyzin, and Smith (FRS). Here $H_0(W)$ is the size of the support of the distribution W. When supports are large, as with passwords, our new construction offers a substantial security improvement.

We provide two new constructions of typo-tolerant password-based authentication schemes. The first combines a LHH or FRS sketch with a standard slow-to-compute hash function, and the second avoids secure sketches entirely, correcting typos instead by checking all nearby passwords. Unlike the previous such brute-force-checking construction, due to Chatterjee et al., our new construction uses a hash function whose runtime is proportional to the popularity of the password (forcing a longer hashing time on more popular, lower entropy passwords). We refer to this as popularity-proportional hashing (PPH). We then introduce a framework for comparing different typo-tolerant authentication approaches. We show that PPH always offers a better time / security trade-off than the LHH and FRS constructions, and for certain distributions outperforms the Chatterjee et al. construction. Elsewhere, this latter construction offers the best trade-off. In aggregate our results suggest that the best known secure sketches are still inferior to simpler brute-force based approaches.

1 Introduction

In many settings, secrets needed for cryptography are measured in a noisy fashion. Biometrics such as fingerprints [31, 35], keystroke dynamics [23, 24], voice [22], and iris scans [31] are examples — each physical measurement produces slight variants of one another. A long line of work has built ad hoc solutions for various cryptographic settings [17, 22–24], while another line of work starting with Dodis, Ostrovsky, Reyzin and Smith [13] explored a general primitive, called a fuzzy extractor, that can reproducibly derive secret keys given noisy measurements. The canonical fuzzy extractor construction combines a traditional key-derivation function (KDF) with a secure sketch, the latter serving as an error-correction code that additionally leaks a bounded amount of information about the secret.

In this work, we explore error correction for noisy secrets in the distribution-sensitive setting, in which one knows the distribution of secrets while designing cryptographic mechanisms. We ground our investigations in an important running case study: typo-tolerant password checking [12, 20], and ultimately offer a number of improvements, both theoretical and practical, on cryptographic error-tolerance in general and the design of typo-tolerant password hardening systems in particular.

Typo-tolerant password checking. Recent work by Chatterjee et al. [12] revealed that users suffer from a high rate of typographical errors (typos), with even a handful of simple-to-correct typos (such as caps lock or other capitalization errors) occurring in 10% of all login attempts at

Dropbox. They offered a usability improvement called "brute-force checking": enumerate probable corrections of the submitted (incorrect) password, and check each of them using a previously stored slow-to-compute cryptographic hash of the correct password (e.g., scrypt [26], argon2 [6], or the PKCS#5 hashes [18,27]). They also show empirically that this relaxed checking approach does not noticeably degrade security, assuming careful selection of the typos to correct.

To maintain performance, however, one must limit the runtime of password checking. One can at most handle approximately $b = \mathsf{RT}/\mathsf{c}$ errors given a runtime budget RT and cryptographic hash function that takes time c to compute.[1] Given that c should be slow — in order to prevent brute-force attacks — the size b of the ball, or set of potential corrections around an incorrect password, must be fairly small. Extending to larger numbers of errors — for example we would like to handle the most frequent 64 typos, which would account for approximately 50% of all typos seen in measurement studies — would appear to force c to be too low to ensure security in the face of attackers that obtain the password hash and mount dictionary attacks.

An existing alternative approach to brute-force ball search is to store, along with the password hash, a small bit of information to help in correcting errors. Because we want to maintain security in the case of compromise of a password database, we must ensure that this helper information does not unduly speed up brute-force cracking attacks. We therefore turn to secure sketches [13].

Secure sketches. Introduced by Dodis, Ostrovsky, Reyzin and Smith [13], sketches allow correction of errors together with bounds on the information leaked about the original secret to an attacker. Traditionally, sketch security is measured by the conditional min-entropy $\tilde{\mathrm{H}}_\infty(W|s)$ of the secret W given the sketch s against unbounded adversaries. Fuller, Reyzin, and Smith (FRS) [15] show that the best one can hope for when achieving correction error at most δ is $\tilde{\mathrm{H}}_\infty(W|s) \geq \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log(1-\delta)$, where $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)$ is called the fuzzy min-entropy of the distribution and captures the worst-case cumulative weight of all points in a ball.

FRS give a clever construction, called layered hashing, that almost achieves the optimal result. They prove that

$$\tilde{\mathrm{H}}_\infty(W|s) \geq \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log(1/\delta) - \log \mathrm{H}_0(W) - 1 \ .$$

Here $\mathrm{H}_0(W)$ is the Hartley entropy, defined to be the logarithm of the size of the distribution's support. The FRS construction provides better bounds than any prior secure sketch construction (and, by the usual construction, the best known fuzzy extractor [13]). The construction works by splitting possible secrets into different layers according to their probability in the distribution W, and then applying a universal hash of a specific length based on a message's layer. Both the layer identifier and the resulting hash value are output. Intuitively, the idea is to tune hash lengths to balance error correction with entropy loss: more probable points are grouped into layers that have much shorter hash values, with less probable points grouped into layers with longer hashes.

The layered sketch works only in (what we call) the distribution-sensitive setting, meaning that the distribution of messages must be known at the time one designs the sketching algorithm. As another potential limitation, correcting an error using the sketch takes time linear in the size of the ball around the point, meaning the construction is only computationally efficient should balls be efficiently enumerable. That said, both conditions are satisfied in some settings, including typo-tolerant password checking: password leaks allow accurate characterization of password distributions [7, 19, 21, 33] when constructing sketches, and as mentioned above, the ball of errors required to cover most observed typos is small and fast to enumerate.

Our contributions. In this work, we explore the open question above: How can we securely correct more errors than Chatterjee et al. in [12]? We offer two new approaches. The first uses secure sketching, and we give a new scheme, called the layer-hiding hash (LHH), and prove that it leaks less information than prior constructions. Perhaps counter-intuitively, LHH does so by actually lengthening, rather than shortening, the output of the sketch as compared to the FRS construction. Our second approach is a new distribution-sensitive brute-force based technique called popularity-proportional hashing (PPH), in which the time required to hash a password is tuned based on its popularity: The more probable the password is, the longer the hashing should take.

[1] This ignores parallelism, but the point remains should one consider it.

Finally, we offer a framework for comparing various approaches, and show that PPH offers a better time / security trade-off than LHH and FRS. For certain error settings, PPH allows us to correct more errors securely than Chatterjee et al.'s brute-force checking. Elsewhere their brute-force checking offers a better trade-off still. In fact, we conjecture that for many distributions no sketch will beat brute-force based approaches.

The layer-hiding hash sketch. Our first contribution is to provide a new sketch that we call the layer-hiding hash (LHH) sketch. We prove that LHH enjoys an upper bound of $\tilde{\mathrm{H}}_\infty(W|s) \geq \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log(1/\delta) - 1$, yielding a substantial saving of $\log \mathrm{H}_0(W)$ bits of entropy over FRS. The LHH starts with the same general approach as FRS, that of putting passwords into layers based on their probability. The key insight is that one can, as the name implies, hide the layer of the password underlying a sketch. To do so, the construction takes the output of applying a layer-specific strongly universal hash to the password and pads it to a carefully chosen maximum length with random bits. During recovery, one looks for a matching prefix of the sketch value when applying (differing length) strongly universal hashes. Hiding the level intuitively avoids leaking additional information to the adversary, but, counterintuitively, the proof of security does not require any properties of the hash functions used. Rather, the proof only uses that the length of hash outputs is bounded plus the fact that (unlike in the FRS construction) sketches from different layers can collide. The proof of correctness relies on the strong universality of the underlying hashes.

LHH's bound improves over FRS (and, consequently, all other known constructions) because it removes the $\log \mathrm{H}_0(W)$ term. The improvement in the bound can be significant. Assuming W places non-zero probability on all passwords from the RockYou password leak [29] already makes $\log \mathrm{H}_0(W) \geq 3$. The min-entropy (let alone fuzzy min-entropy) of common password distributions is commonly measured to be only about 7 bits, making a loss of 3 bits significant. Of course, as pointed out by FRS, the loss due to $\log(1/\delta)$ — which LHH also suffers — is likely to be even more problematic since we'd like δ to be small. An important question left open by our work is whether one can build a sketch that replaces $\log(1/\delta)$ with the optimal $\log(1-\delta)$.

Sketch-based typo-tolerant checking. A seemingly attractive way of building a typo-tolerant password-based authentication scheme is to store a sketch of the password along with a slow-to-compute hash of the password. To later authenticate a submitted string, one first checks it with the slow hash and, if that fails, uses the sketch to error correct, and checks the result with the slow hash. In terms of security, we are primarily concerned about attackers that obtain (e.g., by system compromise) the sketch and slow hash value and mount offline brute-force dictionary attacks. The sketch will leak some information useful to the attacker.

The first challenge that arises in security analysis is that the traditional sketch security measure, conditional min-entropy $\tilde{\mathrm{H}}_\infty(W|s)$, does *not* provide good bounds when adversaries can make many guesses. The reason is that it measures the worst-case probability of guessing the message given the sketch in a single attempt, and for non-flat distributions the success probability of subsequent guesses after the first will be much lower. We therefore introduce a more general conditional q-min-entropy notion, denoted $\tilde{\mathrm{H}}^q_\infty(W|s)$. It is the worst-case aggregate probability of a message being any of q values, conditioned on the sketch. We revisit secure sketches in this new regime and analyze the q-min-entropy for the FRS and LHH constructions. These results are actually strictly more general since they cover the $q = 1$ bounds as well, and so in the body we start with the more general treatment and show the $q = 1$ results mentioned above as corollaries.

Popularity-proportional hashing. We also offer a new distribution-sensitive variant of brute-force checking called popularity-proportional hashing. Recall that brute-force checking uses the same slow hash function for all passwords. In popularity-proportional hashing, we use knowledge of the distribution of passwords to tune the time required to hash each password. The more popular a password, equivalently the more probable, the longer the hash computation.

In typo-tolerant hashing this has a nice effect for certain distributions: the ball of possible passwords around a submitted string will consist of a mix of lower- and higher-probability points, making the aggregate time required to check all of them lower than in brute-force checking. Timing side-channels can be avoided by fixing an upper bound on this aggregate time, and setting the hashing costs of the scheme such that every password can be checked within this time. The checking algorithm is then implemented to process each password for this maximum time, and accordingly its run time reveals nothing about the password being checked. This serves to "smooth" the distribution from the point of view of a brute-force attacker, who must choose between checking a popular password versus lower-cost checks of less probable passwords. We shall ultimately see that PPH offers a better time / security trade-off than sketch-based checking using both FRS and LHH. We note that the benefits of population-proportional hashing appear to be specific to the typo-tolerant setting; in exact checking schemes one would want to hash passwords with the maximum cost allowed by the runtime of the scheme, regardless of their weight.

Comparing the approaches. We use the following methodology to compare the time / security trade-offs of the various approaches to error-tolerant authentication. First, one fixes an error setting, such as choosing a set of 64 frequently made typos, as well as a runtime budget RT for authentication. Then, one compares the brute-force attack security of various constructions that operate in time at most RT and correct the specified errors. So for brute-force checking, for example, one must pick a slow hash function that takes $\mathsf{RT}/64$ time to compute, and for secure sketches one can use a slow hash of time $\mathsf{RT}/2$ (where for simplicity we ignore the sketch cost, which is in practice negligible relative to RT). For popularity-proportional hashing one picks hash speeds so that the ball whose passwords have the highest aggregate probability can be checked in time RT.

With this framework in place, we prove that PPH provides a better time / security trade-off than both FRS-assisted and LHH-assisted checking. The proofs require lower-bounding the security of the FRS and LHH constructions in the face of a computationally efficient attacker whose runtime constraint affords him q slow hash queries (equivalently q guesses at the underlying password). The attack is simple: enumerate probable passwords, check which match the sketch, and output the heaviest q that match. It may not be obvious that this is efficient; we will argue so in the body.

To analyze the attacker's success, we use a proof strategy which at a high level proceeds as follows. We first model the hash underlying the sketch as a random oracle. This is conservative as it can only make the adversary's task harder. We then transform the analysis of the attacker's success probability to a type of balls-in-bins analysis that differs slightly based on the construction. For the FRS case, which is simpler, balls of differing sizes represent passwords of differing weights, and bins represent individual sketch values within a layer. The random oracle 'throws' the balls into the bins; the compromise of a sketch and subsequent guessing attack is captured by sampling a bin and allowing the attacker to choose q balls from it. As such computing a lower bound on the q-conditional min-entropy is reduced to computing the expected (over the random oracle coins) aggregate weight of the q heaviest balls across all bins.

Instead of tackling analysis of this expectation directly, we instead form direct comparison with PPH by showing that with overwhelming probability the set of points queried by an optimal brute-force adversary running in the same time against PPH will be included in the set of points that the adversary against FRS chooses. As such a brute-force attacker against FRS-assisted checking will always either match or (more often) beat attacks against PPH. We derive a similar result for LHH-assisted checking via a modified balls-in-bins experiment.

With the improvement of PPH over sketch-assisted checking established, we next compare PPH and brute-force checking. We quantify precisely the conditions which determine whether PPH or brute-force checking represents the better trade-off for a given error setting, and show that for certain error settings PPH allows us to correct many more errors securely than brute-force checking.

While PPH can be shown to improve on sketch-assisted checking for any distribution, the same is not true for brute-force checking — indeed there exist settings in which brute-force checking will lead to a dramatic reduction in security — and in general comparing the brute-force and sketch-assisted approaches directly appears technically challenging. However by combining the above results, we show that for certain error settings (including passwords) the seemingly simplest brute-force checking approach provides the best trade-off of all — first by invoking the result showing PPH outperforms sketch-assisted checking, and then by showing that brute-force checking offers an even better trade-off than PPH. As such for any given error setting, our results can be used to determine how many errors can be tolerated, and whether PPH or brute-force checking offers the better approach to typo-tolerance.

Extensions and open problems. We frame our results in the context of typo-tolerant password hashing and (reflecting the majority of in-use password hashing functions) primarily measure hashing cost in terms of time. We will in Section 7 briefly discuss how our results may be extended to incorporate memory-hard functions [1–3, 6, 26] and indicate other cryptographic applications, such as authenticated encryption and fuzzy extraction, in which they are applicable. Finally we will discuss the key open problem — can *any* distribution-sensitive secure sketch offer a better time / security trade-off than brute-force based approaches? We conjecture that for a large class of error settings no sketch can perform better. We offer some intuition to this end, and highlight it as an interesting direction for future research.

2 Definitions and Preliminaries

Notation. The set of binary strings of length n is denoted by $\{0,1\}^n$. We use $\perp$ to represent the null symbol. We write $x||y$ to denote the concatenation of two binary strings x and y, and $[y]_1^j$ to denote the binary string y truncated to the lowest order j bits. We let $[j]$ denote the set of integers from 1 to j inclusive, and $[j_1, j_2]$ the set of integers between j_1 and j_2 inclusive. The notation $x \xleftarrow{\$} \mathcal{X}$ denotes sampling an element uniformly at random from the set $\mathcal{X}$, and we let $x \xleftarrow{W} \mathcal{X}$ denote sampling an element from the set $\mathcal{X}$ according to the distribution W. All logs are to base 2, and e denotes Euler's constant. For a given distribution W where $\mathcal{M} = \text{supp}(W)$, we let $w_1, \ldots, w_{|\mathcal{M}|}$ denote the points in the support of W in order of descending probability, with associated probabilities $p_1, \ldots, p_{|\mathcal{M}|}$.

Hash Functions. Here we recall the definitions of universal and strongly universal hash function families.

Definition 1. *A family of hash functions* $\mathsf{F} : \mathcal{S} \times \{0,1\}^\ell \to \{0,1\}^d$ *is said to be universal if for all* $w \neq w' \in \mathcal{S}$*, it holds that*

$$\Pr\left[\mathsf{F}(w;\mathsf{sa}) = \mathsf{F}(w';\mathsf{sa}) \;:\; \mathsf{sa} \xleftarrow{\$} \{0,1\}^\ell\right] = 2^{-d}\,.$$

Definition 2. *A family of hash functions* $\mathsf{F} : \mathcal{S} \times \{0,1\}^\ell \to \{0,1\}^d$ *is said to be strongly universal if for all* $w \neq w' \in \mathcal{S}$*, and* $y, y' \in \{0,1\}^d$*, it holds that*

$$\Pr\left[\mathsf{F}(w;\mathsf{sa}) = y \;:\; \mathsf{sa} \xleftarrow{\$} \{0,1\}^\ell\right] = 2^{-d}\,, \text{ and}$$

$$\Pr\left[\mathsf{F}(w;\mathsf{sa}) = y \wedge \mathsf{F}(w';\mathsf{sa}) = y' \;:\; \mathsf{sa} \xleftarrow{\$} \{0,1\}^\ell\right] \leq 2^{-2d}\,.$$

Error settings and typos. Let $\mathcal{S}$ be a set with associated distance function $\mathsf{dist} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^{\geq 0}$. If dist is a metric over $\mathcal{S}$ — that is to say that dist is non-negative, symmetric, and for all $x, y, z \in \mathcal{S}$, it holds that $\mathsf{dist}(x, z) \leq \mathsf{dist}(x, y) + \mathsf{dist}(y, z)$ — then we say that the pair $(\mathcal{S}, \mathsf{dist})$ is a metric space. We can assign to $\mathcal{S}$ a distribution W, and let $\mathcal{M}$ denote the set of possible messages, $\mathcal{M} = \text{supp}(W)$. We set an error threshold t, denoting the maximum distance between points $w, \tilde{w}$ for which will consider $\tilde{w}$ an error of w. Together these components, $(\mathcal{S}, W, \mathsf{dist}, t)$ define an *error setting.*

For an error setting $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$, the (closed) ball of size t around $\tilde{w} \in \mathcal{S}$ is the set of all points $w' \in \text{supp}(W)$ such that $\mathsf{dist}(w', \tilde{w}) \leq t$, that is $B_t(\tilde{w}) = \{w' \in \text{supp}(W) \mid \mathsf{dist}(w', \tilde{w}) \leq t\}$. We let $\beta_{\max}$ denote the size of the largest ball in the error setting; that is to say $\beta_{\max} = \max_{\tilde{w}} |B_t(\tilde{w})|$. In this work, we shall be especially interested in error settings for which balls are *efficiently enumerable*, a property which we formalize below.

Definition 3. *Let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error setting with maximum ball size* $\beta_{\max}$*. We say* E *has efficiently enumerable balls, if there exists an algorithm* Enum *which takes as input a point* $\tilde{w} \in \mathcal{S}$*, and outputs a set of points* $\mathcal{L}$ *such that for all* $\tilde{w} \in \mathcal{S}$ *it holds that*

$$\Pr\left[\mathcal{L} = B_t(\tilde{w}) \;:\; \mathcal{L} \xleftarrow{\$} \mathsf{Enum}(\tilde{w})\right] = 1\,,$$

and Enum *runs in time polynomial in* $\beta_{\max}$.

Entropy. We now discuss several notions of entropy which capture the maximum success probability of an attacker who attempts to guess a point sampled from a given distribution. Traditionally these notions only consider the case in which the adversary gets one guess. However in subsequent work, when we wish to capture the success rate of an adversary attempting to perform a brute-force attack, it will be useful to generalize these entropy notions to capture the maximum success probability of an adversary who may output a vector of q guesses. We define these notions below generalized to the multi-guess setting; one can easily extract the familiar definitions by setting $q = 1$.

Definition 4. *Let W and Z be distributions. We define the q-min-entropy of W, denoted $\mathrm{H}^q_\infty(W)$ to be,*

$$\mathrm{H}^q_\infty(W) = -\log\left(\max_{w_1,\dots,w_q} \sum_{i=1}^{q} \Pr[\,W = w_i\,]\right) ,$$

where $w_1, \dots, w_q$ are distinct elements of $\mathcal{S}$. The conditional q-min-entropy of W conditioned on Z, denoted $\tilde{\mathrm{H}}^q_\infty(W|Z)$, is defined to be,

$$\tilde{\mathrm{H}}^q_\infty(W|Z) = -\log\left(\sum_z \max_{w_1,\dots,w_q} \sum_{i=1}^{q} \Pr[\,W = w_i \mid Z = z\,] \cdot \Pr[\,Z = z\,]\right) ;$$

and the q-min-entropy of W joint with Z, denoted $\mathrm{H}^q_\infty(W, Z)$, is defined,

$$\mathrm{H}^q_\infty(W, Z) = -\log\left(\max_{\substack{w_1,\dots,w_q\\ z_1,\dots,z_q}} \sum_{i=1}^{q} \Pr[\,W = w_i \wedge Z = z_i\,]\right) ,$$

where the $w_1, \dots, w_q$ and $z_1, \dots, z_q$ are distinct elements of the supports of W and Z respectively. The Hartley entropy of W, denoted $\mathrm{H}_0(W)$, is defined to be,

$$\mathrm{H}_0(W) = \log|\mathrm{supp}(W)| .$$

For an example which surfaces the usefulness of extending min-entropy definitions beyond one guess, consider a pair of distributions W_1 and W_2, such that W_1 is flat with $2^{-\mathrm{H}_\infty(W)} = 2^{-\mu}$ and W_2 consists of one point of probability $2^{-\mu}$ and $2^{2\mu} - 2^{\mu}$ points of probability $2^{-2\mu}$. While $\mathrm{H}^1_\infty(W_1) = \mathrm{H}^1_\infty(W_2) = \mu$, the two distributions are clearly very different, and in particular an attacker given some $q > 1$ guesses to predict a value sampled from each of the distributions is going to have a much easier time with W_1. This difference is highlighted when considering the q-min-entropy, with $\mathrm{H}^q_\infty(W_1) = q \cdot 2^{-\mu}$, whereas $\mathrm{H}^q_\infty(W_2) = 2^{-\mu} + (q-1) \cdot 2^{-2\mu}$.

In the $q = 1$ case, the conditional min-entropy and Hartley entropy are linked via the chain rule for conditional min-entropy [13]. It is straightforward to see that this result extends to the multi-guess setting; for completeness we include a proof in Appendix B.

Lemma 1. *Let W, Z be distributions. Then*

$$\tilde{\mathrm{H}}^q_\infty(W|Z) \geq \mathrm{H}^q_\infty(W, Z) - \mathrm{H}_0(Z) .$$

Secure sketches. Let $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ be an error setting. Secure sketches, introduced by Dodis et al. in [13], allow reconstruction of a message which may be input with noise, while preserving as much of the min-entropy of the original message as possible.

In this work we focus on sketches in the distribution-sensitive setting, in which the distribution of secrets is precisely known at the time of designing the sketch. While distribution-sensitivity may not always be feasible, in the case of passwords there is a long line of work on accurately modeling the distribution of human-chosen passwords. Primarily motivated by password cracking, modeling techniques such as hidden Markov models (HMM) [11], probabilistic context free grammars (PCFG) [32,33], or neural networks [21] use the plethora of real password leaks (e.g., [9]) to learn good estimates of W. See [19] for a detailed discussion of these approaches. Of course, estimates may be wrong. A discussion on the effect of transferring our results to a setting in which the distribution is only approximately known is given in Appendix A. We recall the formal definition of secure sketches below.

Definition 5. *Let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error setting. A secure sketch for* E *is a pair of algorithms* $\mathsf{S} = (\mathsf{SS}, \mathsf{Rec})$ *defined as follows:*

– SS *is a randomized algorithm which takes as input* $w \in \mathcal{S}$, *and outputs a bit string* $s \in \{0,1\}^*$.
– Rec *is an algorithm, possibly randomized, which takes as input* $\tilde{w} \in \mathcal{S}$ *and* $s \in \{0,1\}^*$, *and outputs* $w' \in B_t(\tilde{w}) \cup \{\perp\}$.

We note that we slightly modify the definition of [15] so that Rec on input $\tilde{w}$ always outputs $w' \in B_t(\tilde{w}) \cup \{\perp\}$, as opposed to $w' \in \mathcal{S} \cup \{\perp\}$. As we shall see in the following definition, we only require Rec to return the correct point if that point lies in $B_t(\tilde{w})$. As such this is mainly a syntactic change, and all pre-existing sketch constructions discussed in this work already adhere to the condition. In the following definition, we generalize the security requirement to the multi-guess setting in the natural way; the usual definition (e.g. [13, 15]) is obtained by setting $q = 1$.

Definition 6. *A sketch* $\mathsf{S} = (\mathsf{SS}, \mathsf{Rec})$ *is an* $((\mathcal{S}, W, \mathsf{dist}, t), \bar{\mu}_q, \delta)$-secure sketch *if:*

1. *(*Correctness*) For all* $w, \tilde{w} \in \mathcal{S}$ *for which* $\mathsf{dist}(w, \tilde{w}) \leq t$, *it holds that*
$$\Pr\left[\, w = w' \ : \ w' \leftarrow \mathsf{Rec}(\tilde{w}, \mathsf{SS}(w)) \,\right] \geq 1 - \delta \ ,$$
where the probability is taken over the coins used by SS *and* Rec.

2. *(*Security*) The q-min-entropy of* W *conditioned on* $\mathsf{SS}(W)$ *is such that,*
$$\tilde{\mathrm{H}}^{q}_{\infty}(W|\mathsf{SS}(W)) \geq \bar{\mu}_q \ .$$

Since the help string s is public any party — including the adversary — can query $\mathsf{Rec}(\cdot, s)$ on $\tilde{w} \in \mathcal{S}$. In an ideal secure sketch, knowledge of s would offer no greater advantage than that gained via oracle access to $\mathsf{Rec}(\cdot, s)$. In this case, an adversary challenged to guess the original value $w \in \mathcal{S}$ is forced to guess some $\tilde{w}$ such that $\mathsf{dist}(w, \tilde{w}) \leq t$. To capture this notion of ideal secure sketch security, Fuller et al. [15] introduce the notion of fuzzy min-entropy, which we generalize to the multi-guess setting in the natural way.

Definition 7. *Let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error setting. The q-fuzzy min-entropy of* W *is defined to be,*
$$\mathrm{H}^{q,\mathrm{fuzz}}_{t,\infty}(W) = -\log\left(\max_{\tilde{w}_1, \ldots, \tilde{w}_q} \sum_{w' \in \cup_{i=1}^{q} B_t(\tilde{w}_i)} \Pr\left[\, W = w' \,\right] \right) ,$$
where $\tilde{w}_1, \ldots, \tilde{w}_q$ *are distinct elements of* $\mathcal{S}$.

3 New Bounds for FRS Sketches

In this section we describe and analyze two constructions of secure sketches due to Fuller, Reyzin, and Smith [15]. The FRS sketches have a number of attractive properties. The first is that these are the only secure sketches (to our knowledge) that can be utilized with *any* choice of distance function dist. We would like this flexibility so that ultimately we can tailor the distance function used to the context of correcting password typos for which, being non-symmetric, traditional metrics such as edit distance are not best suited [12].

Even if edit distance were appropriate, we know of no constructions which provide sufficient security when used with parameters typical to password distributions. Constructions in [13, 14] either embed the edit metric into the Hamming or set distance metrics using a low distortion embedding of Ostrovsky and Rabani [25], or use a novel c-shingling technique.

As pointed out in [12], when applied to typical password distributions which have a large alphabet of 96 ASCII characters, then even if we only attempt to correct edit distance one errors, these constructions incur entropy loss ≈ 91 bits and ≈ 31 bits respectively. Given that password distributions typically have at most 8 bits of min-entropy [8], it is clear these constructions are unsuitable for our purposes.

Most importantly, the FRS constructions achieve almost optimal security in the $q = 1$ case. It was shown in [15] that high fuzzy min-entropy is a necessary condition for the existence of a good

```
FRS1-SS(w) :                    FRS1-Rec(w̃, s) :

sa ←$ {0,1}^ℓ                    (y, sa) ← s
y ← F(w; sa)                    for w′ ∈ B_t(w̃)
s ← (y, sa)                         if F(w′; sa) = y
Ret s                                   Ret w′
                                Ret ⊥
```

Fig. 1: Construction of a secure sketch $\mathsf{FRS1} = (\mathsf{FRS1\text{-}SS}, \mathsf{FRS1\text{-}Rec})$ for an error setting $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ from a universal hash function family $\mathsf{F} : \mathcal{S} \times \{0,1\}^\ell \to \{0,1\}^{\log(\beta_{\max})+\log(1/\delta)}$. Here $\beta_{\max}$ denotes the size of the largest ball in the error setting.

secure sketch or fuzzy extractor for a given error setting, surfacing a lower bound on the security of such schemes. We recall the result in the lemma below, which we extend to the multi-guess setting. The proof is given in Appendix C.

Lemma 2. *Let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error setting, and let* $\mathsf{S} = (\mathsf{SS}, \mathsf{Rec})$ *be an* $((\mathcal{S}, W, \mathsf{dist}, t), \bar{\mu}_q, \delta), \bar{\mu}_q, \delta)$*-secure-sketch. Then* $\bar{\mu}_q \le \mathrm{H}^{q,\mathrm{fuzz}}_{t,\infty}(W) - \log(1-\delta)$.

FRS showed that in the distribution-sensitive setting, in which the precise distribution is known at the time of building the sketch, high fuzzy min-entropy also implies the existence of a good secure sketch for that distribution. We recall their constructions, and prove new results about them.

3.1 Secure Sketches for Flat Distributions

FRS describe a secure sketch which is nearly optimal for error settings $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ such that W is flat, which we recall in Figure 1. We refer to this construction as $\mathsf{FRS1} = (\mathsf{FRS1\text{-}SS}, \mathsf{FRS1\text{-}Rec})$.

The construction is built from a universal hash function family with output length $\log(\beta_{\max}) + \log(1/\delta)$ bits, where $\beta_{\max}$ denotes the size of the largest ball in the error setting. $\mathsf{FRS1\text{-}SS}$ chooses a salt $\mathsf{sa} \xleftarrow{\$} \{0,1\}^\ell$, computes $y = \mathsf{F}(w; \mathsf{sa})$, and outputs $s = (y, \mathsf{sa})$. On input $\tilde{w} \in \mathcal{S}$ and s, Rec searches in $B_t(\tilde{w})$ for a point w' such that $\mathsf{F}(w'; \mathsf{sa}) = y$, returning the first match which it finds. The authors note that the construction is not novel, with universal hash functions representing a commonly used tool for information reconciliation (e.g., [5], [28], [30]). Correctness follows from a straightforward application of Markov's Inequality. In the following lemma we extend analysis to cover the q-conditional min-entropy. The proof is given in Appendix C.

Lemma 3. *Let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error setting for which* W *is flat, and let* $\beta_{\max}$ *denote the size of the largest ball. Let* $\mathsf{FRS1} = (\mathsf{FRS1\text{-}SS}, \mathsf{FRS1\text{-}Rec})$ *be as described in Figure 1, and let* $\mathsf{F} : \mathcal{S} \times \{0,1\}^\ell \to \{0,1\}^{\log(\beta_{\max})+\log(1/\delta)}$ *be a family of universal hash functions where* $0 < \delta < 1$. *Then* $\mathsf{FRS1}$ *is a* $((\mathcal{S}, W, \mathsf{dist}, t), \bar{\mu}_q, \delta), \bar{\mu}_q, \delta)$*-secure sketch, where*

$$\bar{\mu}_q \ge \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log(q) - \log(1/\delta) \; .$$

3.2 Layered Hashing for Non-flat Distributions

The above construction may be significantly less secure in settings where the distribution in question is non-flat. In this case, having high fuzzy min-entropy does not exclude the possibility that the distribution contains a dense ball consisting of many low probability points. Disambiguating between points in this dense ball forces a hash function with a large range to be used, which leaks more information to adversaries.

The key idea is to split the support of the distribution into nearly flat layers; the layer in which a point lies is determined by its probability, and the layers are defined such that the probabilities of points in any given layer differ by at most a factor of two. We include the index of the layer in which a point lies as part of its sketch, and then apply the secure sketch for flat distributions of Lemma 3 tuned to the parameters of the appropriate layer. Revealing the layer in which a point lies degrades security; in an attempt to limit the damage, the number of layers is restricted so the extra loss amounts to $\log \mathrm{H}_0(W) + 1$ bits; for full details of the proof see [15].

```
FRS2-SS(w) :
$j \leftarrow \mathsf{L}(w)$
if $j = \lambda$
    $s \leftarrow (w, \perp, \lambda)$
else
    $\mathsf{sa} \xleftarrow{\$} \{0,1\}^{\ell_j}$
    $y \leftarrow \mathsf{F}_j(w; \mathsf{sa})$
    $s \leftarrow (y, \mathsf{sa}, j)$
Ret $s$

FRS2-Rec($\tilde{w}$, s) :
$(y, \mathsf{sa}, j) \leftarrow s$
If $j = \lambda$
    Ret $y$
for $w' \in B_t(\tilde{w}) \cap L_j$
    if $\mathsf{F}_j(w'; \mathsf{sa}) = y$
        Ret $w'$
Ret $\perp$

FRS2-Layer(W) :
$\lambda \leftarrow \mathrm{H}_\infty(W) + \lfloor \mathrm{H}_0(W) - 1 \rfloor$
for $j = \mu, \ldots, \lambda - 1$
    $L_j \leftarrow (2^{-(j+1)}, 2^{-j}]$
$L_\lambda \leftarrow (0, 2^{-\lambda}]$
Ret $\{L_j \ : \ j \in [\mu, \lambda]\}$
```

Fig. 2: Secure sketch $\mathsf{FRS2} = (\mathsf{FRS2\text{-}SS}, \mathsf{FRS2\text{-}Rec})$ for an error setting $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ from a set of universal hash function families $\mathsf{F}_j : \mathcal{S} \times \{0,1\}^{\ell_j} \to \{0,1\}^{j - \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) + \log(1/\delta) + 1}$ for $j \in [\mu, \lambda]$, utilizing layering FRS2-Layer.

For simplicity of exposition, we assume the existence of an efficient algorithm L which takes as input a point $w \in \mathcal{S}$ and outputs the index $j \in \mathcal{J}$ of the layer in which it lies. We note that the parameters required to compute the cut-off points between layers are readily obtained from the password model, so computing the partitions is straightforward in practice; provided we can efficiently look up the weights of points in the password model, the algorithm L will be efficient also. The full construction is given in Figure 2.

Theorem 1. [15] *Let $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ be an error setting. Let $\mathsf{F}_j : \mathcal{S} \times \{0,1\}^{\ell_j} \to \{0,1\}^{j - \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) + \log(1/\delta) + 1}$ be a family of universal hash functions, where $0 < \delta \leq \frac{1}{2}$. Consider* FRS2 = (FRS2-SS, FRS2-Rec) *with layering* FRS2-Layer *as defined in Figure 2. Then* FRS2 *is a $((\mathcal{S}, W, \mathsf{dist}, t), \bar{\mu}_1, \delta)$-secure sketch where*

$$\bar{\mu}_1 = \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log \mathrm{H}_0(W) - \log(1/\delta) - 1.$$

In the following theorem, we provide an analysis for FRS2 in the q-min-entropy setting. Our analysis also provides a tighter bound in the case that $q = 1$. The full proof is given in Appendix C.

Theorem 2. *Let $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ be an error setting, where $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = \tilde{\mu}$. Let $\mathsf{F}_j : \mathcal{S} \times \{0,1\}^{\ell_j} \to \{0,1\}^{j - \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) + \log(1/\delta) + 1}$ be a family of universal hash functions, where $0 < \delta \leq \frac{1}{2}$. Consider* FRS2 = (FRS2-SS, FRS2-Rec) *with layering* FRS2-Layer *as defined in Figure 2. Then* FRS2 *is a $((\mathcal{S}, W, \mathsf{dist}, t), \bar{\mu}_q, \delta)$-secure sketch for,*

$$2^{-\bar{\mu}_q} \leq \Pr[W \in L_\lambda] + \sum_{j=\mu}^{\lambda-1} \Pr[W \in L_j(q \cdot |R_j|)] .$$

Here $L_j(q')$ denotes the set of the $\min\{q', |L_j|\}$ heaviest points in layer L_j. We let $R_j = range(\mathsf{F}_j)$ and let $L_\lambda = \{w \in W : \Pr[W = w] < 2^{-\lambda}\}$ where $\lambda = \mathrm{H}_\infty(W) + \lfloor \mathrm{H}_0(W) - 1 \rfloor$.

We note that the additional tightness in the bound is especially beneficial when considering distributions with many sparsely populated or empty layers. To give a concrete example of a distribution for which the tightness in the bound makes a significant difference, consider an error setting for which W contains 2^{99} points of weight 2^{-100}, and 2^{199} points of weight 2^{-200}, and the case that $q = 1$. Since $\mathrm{H}_0(W) \approx 199$, the bound of Theorem 1 implies that $\tilde{\mathrm{H}}_\infty(W | \mathsf{FRS2\text{-}SS}(W)) \geq \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log(1/\delta) - 8.64$. In contrast applying the new bound of Theorem 2 implies that $\tilde{\mathrm{H}}_\infty(W | \mathsf{FRS2\text{-}SS}(W)) \geq \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log(1/\delta) - 2$, (since $\Pr[W \in L_j(|R_j|)] \leq 2^{-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) + \log(1/\delta) + 1}$ for $j = 100, 200$, and 0 otherwise). This results in a saving of over 6.6 bits of entropy.

4 A New Construction: Layer Hiding Hash

In this section we present a new construction which yields a substantial increase in security over FRS2, while enjoying the same degree of correctness. The construction, which we call layer hiding hash and denote LHH = (LHH-SS, LHH-Rec) is similar to FRS2, but crucially does not explicitly reveal the layer in which a point lies as part of the sketch.

First we split the distribution into layers as shown in Figure 3. Note that this layering is slightly different to that used in FRS2. We now require a family of *strongly* universal hash functions, which we use to hash points $w \in \mathcal{M}$ to a fixed length which is a parameter of the scheme, and then truncate this hash to various lengths depending on the layer in which the point lies (in turn creating a family of strongly universal hash functions for each layer). The strong universality of the hash is required for the proof of correctness in which we bound the probability that the hash of a point w collides with a given string; this represents a recovery error and the lengths of the truncated hashes are chosen such that the probability this event occurs is at most δ.

The twist that enables the security savings is that rather than outputting this truncated hash as is and revealing the layer in which a point lies, we now view this hash as a *prefix.* The sketch is then computed by choosing a string at random from the set of all strings of a given length (a parameter of the scheme) which share that prefix. This is done efficiently by padding the hash with the appropriate number of random bits. The effect of this is to nearly flatten the joint distribution of W and $\mathsf{SS}(W)$ such that for all $w \in \mathcal{M}$ and $s \in \text{supp}(\mathsf{SS}(W))$, it holds that $\Pr[W = w \wedge \mathsf{SS}(W) = s] \le 2^{-(\gamma+\ell)}$ (where γ indexes the layer of least probable points, and ℓ denotes the length of the salt) *regardless* of the layer in which the point lies. During recovery, the sketch searches in the ball of the input for a point whose truncated hash matches the prefix of the sketch value, and outputs the first match it finds. The full construction is shown in Figure 3.

```
LHH-SS($w$) :
$\mathsf{sa} \xleftarrow{\$} \{0,1\}^\ell$
$j \leftarrow \mathsf{L}(w)$
$y_1 \leftarrow [\mathsf{F}(w;\mathsf{sa})]_1^{j-\tilde{\mu}+\log(\frac{1}{\delta})+1}$
$y_2 \xleftarrow{\$} \{0,1\}^{\gamma-j}$
$y \leftarrow y_1 || y_2$
$s \leftarrow (y, \mathsf{sa})$
Ret $s$
```

```
LHH-Rec($\tilde{w}, s$) :
$(y, \mathsf{sa}) \leftarrow s$
for $w' \in B_t(\tilde{w})$
    $j' \leftarrow \mathsf{L}(w')$
    $y' \leftarrow [\mathsf{F}(w';\mathsf{sa})]_1^{j'-\tilde{\mu}+\log(\frac{1}{\delta})+1}$
    if $y' = [y]_1^{j'-\tilde{\mu}+\log(\frac{1}{\delta})+1}$
        Ret $w'$
Ret $\perp$
```

```
LHH-Layer($W$) :
$\gamma \leftarrow \left\lfloor -\log\left(\min_{w \in W} \Pr[W = w]\right)\right\rfloor$
for $j = \mu, \ldots, \gamma$
    $L_j \leftarrow (2^{-(j+1)}, 2^{-j}]$
Ret $\{L_j \; : \; j \in [\mu, \gamma]\}$
```

Fig. 3: Construction of secure sketch $\mathsf{LHH} = (\mathsf{LHH\text{-}SS}, \mathsf{LHH\text{-}Rec})$ for an error setting $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ with $\tilde{\mu} = \mathrm{H}_{t,\infty}^{\text{fuzz}}(W)$, from a family of strongly universal hash functions $\mathsf{F} : \mathcal{S} \times \{0,1\}^\ell \to \{0,1\}^{\gamma-\tilde{\mu}+\log(\frac{1}{\delta})+1}$, utilizing layering $\mathsf{LHH\text{-}Layer}$.

In the following theorem we analyze the correctness and security of LHH, and emphasize the substantial entropy saving in the $q = 1$ case of $\log \mathrm{H}_0(W)$ bits in comparison to $\mathsf{FRS2}$. The proof is given in Appendix C.

Theorem 3. *Let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error setting. Let* $\mathsf{F} : \mathcal{S} \times \{0,1\}^\ell \to \{0,1\}^{\gamma - \mathrm{H}_{t,\infty}^{\text{fuzz}}(W) + \log(1/\delta)+1}$ *be a family of strongly universal hash functions where* $0 < \delta < 1$*. Let* $\mathsf{LHH} = (\mathsf{LHH\text{-}SS}, \mathsf{LHH\text{-}Rec})$ *be as shown in Figure 3 with layering* $\mathsf{LHH\text{-}Layer}$*. Then* LHH *is a* $((\mathcal{S}, W, \mathsf{dist}, t), \bar{\mu}_q, \delta), \bar{\mu}_q, \delta)$*-secure sketch, where,*

$$\bar{\mu}_q = \mathrm{H}_\infty^{q\cdot\eta}(W') \; .$$

Here $\eta = 2^{\gamma - \mathrm{H}_{t,\infty}^{\text{fuzz}}(W) + \log(1/\delta)+1}$*, and* W' *is the distribution constructed by taking each point* $w \in \mathcal{M}$ *and replacing it with* $2^{(\gamma-j)}$ *points, each of weight* $\Pr[W = w] \cdot 2^{-(\gamma-j)}$*, where* $w \in L_j$*. In particular in the case where* $q = 1$*, this gives,*

$$\bar{\mu}_1 \ge \mathrm{H}_{t,\infty}^{\text{fuzz}}(W) - \log(1/\delta) - 1 \; .$$

5 Typo-Tolerant Password-Based Key Derivation

In this section we consider the application of secure sketches to typo-tolerant password-based key-derivation functions (PBKDF). PBKDFs are used in a number of settings, for example in password-based authentication during login and password-based encryption. PBKDFs are designed to slow attackers that mount brute-force attacks, by incorporating a computationally slow and / or memory-consuming task.

We begin by treating password-based authentication schemes (PBAS). We discuss how to extend to other PBKDF settings in Section 7. Roughly speaking, our results will apply in any

```
$\underline{\mathrm{MR}^{\mathcal{A}}_{\mathsf{PBAS,E}}}$
$w \xleftarrow{W} \mathcal{S}$
$h \xleftarrow{\$} \mathrm{Reg}(w)$
$w' \leftarrow\!\$\; \mathcal{A}^{\mathbf{H}}(h)$
$\mathrm{Ret}\ (w' = w)$
```

Fig. 4: Security game for password recovery in an offline brute-force cracking attack for a PBAS $\mathsf{PBAS} = (\mathrm{Reg}, \mathrm{Chk})$ and error setting $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$.

situation in which the PBKDF-derived key is used in a cryptographically strong authentication setting, including notably password-based authenticated encryption. We will use an oracle model to capture computational slowness, analogous to prior treatments of PBKDFs in the random oracle model (ROM) [4, 34]. We will denote by $\mathbf{H}$ the oracle, and assume it behaves as a random oracle mapping arbitrary length strings to randomly chosen strings of a fixed length ℓ_H. We let $\mathbf{H}$ take an additional input c representing the unit cost of querying $\mathbf{H}$. We formalize such schemes below, following [12].

Definition 8. *A PBAS is a pair of algorithms* $\mathsf{PBAS} = (\mathrm{Reg}, \mathrm{Chk})$ *defined as follows:*

– $\mathrm{Reg}^{\mathbf{H}}$ *is a randomized algorithm which takes as input a password* $w \in \mathcal{M}$ *and returns a string* h*.*
– $\mathrm{Chk}^{\mathbf{H}}$ *is a (possibly randomized) algorithm which takes as input* $\tilde{w} \in \mathcal{S}$ *and string* h*, and returns either true or false.*

Both algorithms have access to oracle $\mathbf{H}(\cdot;\cdot,\mathsf{c}) : \{0,1\}^* \times \{0,1\}^{\ell_{\mathsf{sa}}} \rightarrow \{0,1\}^{\ell_H}$ *where* c *denotes the unit cost of calling* $\mathbf{H}$*.*

The canonical scheme $\mathsf{PBAS} = (\mathrm{Reg}, \mathrm{Chk})$, used widely in practice, has $\mathrm{Reg}^{\mathbf{H}}$ choose a random salt sa and output $(\mathsf{sa}, \mathbf{H}(w; \mathsf{sa}, \mathsf{c}))$. Then, $\mathrm{Chk}^{\mathbf{H}}(\tilde{w}, (\mathsf{sa}, h))$ computes $h' = \mathbf{H}(\tilde{w}; \mathsf{sa}, \mathsf{c})$ and outputs $h' \overset{?}{=} h$. The runtime is c, the cost of one query to $\mathbf{H}$. Typically PBKDF $\mathbf{H}$ will be the c-fold iteration $\mathbf{H}(\cdot;\cdot,\mathsf{c}) = H^{\mathsf{c}}(\cdot;\cdot)$ of some cryptographic hash function $H : \{0,1\}^* \times \{0,1\}^{\ell_{\mathsf{sa}}} \rightarrow \{0,1\}^{\ell_H}$ which we model as a random oracle. We will, in general, ignore the cost of other operations (e.g., the comparison $h = h'$) as they will be dominated by c. For example if $\mathbf{H}$ consists of 10,000 iterations of a hash function such as SHA-256 then c would be the cost of 10,000 computations of SHA-256.

We do not yet consider memory-hardness, and leave a proper treatment of it to future work (see Section 7).

Security against cracking attacks. We will focus primarily on security against offline cracking attacks. Should an adversary obtain access to the output of Reg, we want that it should be computationally difficult — in terms of the number of oracle calls to $\mathbf{H}$ — to recover the password w. We formalize this in game MR shown in Figure 4, a close relative of existing security notions capturing brute-force attacks against passwords (e.g. [4,16]). For an error setting $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$, we define the advantage of an adversary $\mathcal{A}$ against a scheme PBAS by

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS,E}}(\mathcal{A}) = \Pr\left[\mathrm{MR}^{\mathcal{A}}_{\mathsf{PBAS,E}} \Rightarrow \mathsf{true}\right] .$$

The probability is over the coins used in the game and those of the adversary. We assume that the adversary $\mathcal{A}$ has exact knowledge of the error setting E. The number of queries $\mathcal{A}$ may make to oracle $\mathbf{H}$ is determined by its run time T and the cost c of querying $\mathbf{H}$, and for simplicity all other computations are assumed to be free. For example if $\mathbf{H}$ has cost c, then an adversary $\mathcal{A}$ running in time T may make $q = T/\mathsf{c}$ queries to $\mathbf{H}$.

5.1 Brute-force checkers

To improve the usability of a given $\mathsf{PBAS} = (\mathrm{Reg}, \mathrm{Chk})$ for some error setting $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$, Chatterjee et al. [12] advocate retaining the original Reg algorithm but modify the Chk algorithm to a 'relaxed checker' that loosens the requirement that a password be entered exactly. They define the (what we will call) brute-force error correction scheme $\mathsf{PBAS\text{-}BF} = (\mathrm{Reg}, \mathrm{Chk\text{-}BF})$ as follows.

Definition 9. *Let* $\mathsf{PBAS} = (\text{Reg}, \text{Chk})$*, and let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error-setting. Let* $\mathbf{H}(\cdot;\cdot,\mathsf{c}_{\text{bf}})$ *be a random oracle. Then the brute-force error-correction scheme* $\mathsf{PBAS\text{-}BF} = (\text{Reg}, \text{Chk-BF})$ *is defined as follows,*

– $\text{Reg}(w)$ *chooses a salt* sa *at random, and outputs* $(\mathsf{sa}, \mathbf{H}(w; \mathsf{sa}, \mathsf{c}_{\text{bf}}))$*.*
– $\text{Chk-BF}(\tilde{w}, (\mathsf{sa}, h))$ *checks whether* $h = \mathbf{H}(\tilde{w}; \mathsf{sa}, \mathsf{c}_{\text{bf}})$ *or* $h = \mathbf{H}(w'; \mathsf{sa}, \mathsf{c}_{\text{bf}})$ *for each* $w' \in B_t(\tilde{w})$*. If it finds a match, it returns true, and otherwise returns false.*

Since c_{bf} denotes the unit cost of running $\mathbf{H}$, it follows that the runtime RT of this algorithm is the unit cost of $\mathbf{H}$ times the worst case ball size, i.e., $\mathsf{RT} = \mathsf{c}_{\text{bf}} \cdot \beta_{\max}$, where $\beta_{\max} = \max_{\tilde{w}} |B_t(\tilde{w})|$. To avoid potential side-channels, one may want to always compute $\mathbf{H}$ the same number of times, making the run time always RT.

An adversary $\mathcal{A}$ running in time at most T in game MR can make at most $q_{\text{bf}} = T/\mathsf{c}_{\text{bf}}$ queries to $\mathbf{H}$. It is straightforward to see that $\mathcal{A}$'s optimal strategy is to query the q_{bf} most probable points in W to $\mathbf{H}$, and so $\mathbf{Adv}^{\text{mr}}_{\mathsf{PBAS\text{-}BF},\mathsf{E}}(\mathcal{A}) \leq 2^{-\mathrm{H}^{q_{\text{bf}}}_{\infty}(W)}$. This value is precisely the q-success rate of Boztas [10], and is a standard measure of the predictability of a password distribution.

Empirical analysis in [12] finds that when we only attempt to correct a very small number of errors per password (e.g. balls of size at most four) then the brute-force checker yields a noticeable increase in usability for a small reduction in security. However the above security bound highlights a potential limitation of the brute-force checker; if we wish to correct balls with larger numbers of points, we either need to accept an impractically long run time, or reduce c_{bf} to a level which for some error settings may result in significant security loss. This is an important consideration in the context of password typos where the large alphabet (of up to 96 ASCII characters depending on the password creation policy) means that the set of points within edit distance one of a six character string $\tilde{w} \in \mathcal{S}$ contains well over 1000 points. This raises the question of whether secure sketches can be employed to achieve a better time / security trade-off.

5.2 Typo-Tolerant PBAS using Secure Sketches

The error-correcting properties of secure sketches (see Section 2) make them a natural candidate to build typo-tolerant PBAS schemes. We now describe how to compose a secure sketch with any existing PBAS scheme to create a typo-tolerant PBAS. The construction is so simple it is essentially folklore. See also a discussion by Dodis et al. [13]. Our contribution here is merely to formalize it so that we can provide a full security analysis in our computational setting.

Definition 10. *Let* $\mathsf{S} = (\mathsf{SS}, \mathsf{Rec})$ *be an secure-sketch for error setting* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$*. Let* $\mathbf{H}(\cdot;\cdot,\mathsf{c}_{\text{ss}})$ *be a random oracle. Then we define the scheme* $\mathsf{PBAS\text{-}SS[S]} = (\text{Reg-SS}, \text{Chk-SS})$ *as follows:*

- $\text{Reg-SS}(w)$ *runs* $\mathsf{SS}(w)$ *to obtain a sketch* s*. It additionally chooses a salt* sa *at random, and outputs* $(s, \mathsf{sa}, \mathbf{H}(w; \mathsf{sa}, \mathsf{c}_{\text{ss}}))$*.*
- $\text{Chk-SS}(\tilde{w}, (s, \mathsf{sa}, h))$ *first runs* $w' \leftarrow\!\!\$\ \mathsf{Rec}(s, \tilde{w})$*. It then checks whether* $h = \mathbf{H}(\tilde{w}; \mathsf{sa}, \mathsf{c}_{\text{ss}})$ *or* $h = \mathbf{H}(w'; \mathsf{sa}, \mathsf{c}_{\text{ss}})$*. If either matches, it returns true, and otherwise returns false.*

As written the run time of checking is always two calls[2] to $\mathbf{H}$ with unit cost c_{ss}; it follows that $\mathsf{RT} = 2 \cdot \mathsf{c}_{\text{ss}}$. One could short-circuit checking by first checking $\tilde{w}$ and only computing the secure sketch if authentication fails, however side-channels would now reveal when a user makes a typo. We would not want to short-circuit the calculations of $\mathbf{H}$ on the sketch outputs, as this could reveal even more information about w to a side-channel adversary.

An adversary $\mathcal{B}$ running in time at most T in game MR can make at most $q_{\text{ss}} = T/\mathsf{c}_{\text{ss}}$ queries to $\mathbf{H}$. It is clear that $\mathcal{B}$'s optimal strategy on input $(s, \mathsf{sa}, \mathbf{H}(w; \mathsf{sa}, \mathsf{c}_{\text{ss}}))$ is to query the q_{ss} heaviest points when ordered in terms of $\Pr[\, W = w \mid \mathsf{SS}(W) = s\,]$ to $\mathbf{H}(\cdot; \mathsf{sa}, \mathsf{c}_{\text{ss}})$. As such for a given $\mathsf{S} = (\mathsf{SS}, \mathsf{Rec})$ and error setting E, the definition of q-conditional min-entropy implies that $\mathbf{Adv}^{\text{mr}}_{\mathsf{PBAS\text{-}SS[S]},\mathsf{E}}(\mathcal{B}) \leq 2^{-\tilde{\mathrm{H}}^{q_{\text{ss}}}_{\infty}(W|\mathsf{SS}(W))}$.

[2] If S is perfectly correct, it would be sufficient to simply run $w' \leftarrow \mathsf{Rec}(s, \tilde{w})$ and check if $h = \mathbf{H}(w'; \mathsf{sa}, \mathsf{c}_{\text{ss}})$, reducing the number of calls to $\mathbf{H}$ to one.

5.3 Popularity-Proportional Hashing

We now describe a new distribution-sensitive variant of brute-force checking — popularity-proportional hashing (PPH). We shall see in Section 6 that for certain error settings and cracking attack run times, PPH allows us to correct more password errors securely than brute-force checking. For all other error settings, it serves as a useful stepping stone to show that brute-force checking provides a superior time / security trade-off than sketch-based typo-tolerant PBAS based on FRS and LHH.

The key idea is to partition the points in the error setting into layers based upon their probability (as done in LHH), then have the hashing cost vary across the layers. This is accomplished by having the PBKDF $\mathbf{H}$ take as input a different iteration count for each layer. Formally, for a distribution W with $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = \tilde{\mu}$, if a password w is such that $\Pr[W = w] \in (2^{-(j+1)}, 2^{-j}]$, then hashing w incurs a cost of $\mathsf{c}^{j}_{\mathrm{PPH}} = \mathsf{c}_{\mathrm{PPH}} \cdot 2^{\tilde{\mu}-(j+1)}$, where $\mathsf{c}_{\mathrm{PPH}}$ is a parameter of the scheme. By making it more computationally intensive for an attacker to hash popular passwords, the boost to an attacker's success probability resulting from querying a likely password is offset by the greater cost incurred to compute the relevant PBKDF output. We provide full details of the scheme in Figure 5. In the following lemma, we show how to set the parameter $\mathsf{c}_{\mathrm{PPH}}$ to achieve a desired checking run time RT.

Lemma 4. *Let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error setting. Let* $\mathsf{PBAS\text{-}PPH}$ *be as shown in Figure 5 using random oracle* $\mathbf{H}$*. Then setting* $\mathsf{c}_{\mathrm{PPH}} = \mathsf{RT}$ *implies that*

$$\mathsf{RT}(\text{Chk-PPH}, \mathsf{c}_{\mathrm{PPH}}) \leq \mathsf{RT} ,$$

where $\mathsf{RT}(\text{Chk-PPH}, \mathsf{c}_{\mathrm{PPH}})$ *denotes the maximum run time of* Chk-PPH *with cost parameter* $\mathsf{c}_{\mathrm{PPH}}$ *on any input* $\tilde{w} \in \mathcal{S}$*.*

Proof: Fix any point $\tilde{w} \in \mathcal{S}$. Then if W is such that $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = \tilde{\mu}$, and recalling that $w \in L_j$ implies that $\Pr[W = w] > 2^{-(j+1)}$ it follows that

$$2^{-\tilde{\mu}} \geq \Pr[W \in B_t(\tilde{w})] > \sum_{j=\mu}^{\gamma} |B_t(\tilde{w}) \cap L_j| 2^{-(j+1)} .$$

Multiplying both sides by $\mathsf{c}_{\mathrm{PPH}} \cdot 2^{\tilde{\mu}}$ and recalling that $\mathsf{c}_{\mathrm{PPH}} = \mathsf{RT}$ and $\mathsf{c}^{j}_{\mathrm{PPH}} = \mathsf{c}_{\mathrm{PPH}} \cdot 2^{\tilde{\mu}-(j+1)}$ gives

$$\mathsf{RT} > \sum_{j=\mu}^{\gamma} |B_t(\tilde{w}) \cap L_j| \mathsf{c}_{\mathrm{PPH}} \cdot 2^{\tilde{\mu}-(j+1)} = \sum_{j=\mu}^{\gamma} |B_t(\tilde{w}) \cap L_j| \mathsf{c}^{j}_{\mathrm{PPH}} ,$$

where the right hand side is precisely the run time of Chk-PPH on input $\tilde{w}$. Since the choice of $\tilde{w}$ was arbitrary, it follows that the run time of Chk-PPH on any input $\tilde{w} \in \mathcal{S}$ is at most RT, proving the claim. ▌

```
Reg-PPH(w) :
sa ←$ {0,1}^{ℓ_sa}
j ← L(w)
h ← H(w; sa, c^j_PPH)
Ret (sa, h)
```

```
Chk-PPH(w̃, (sa, h)) :
for w' ∈ B_t(w̃)
    j' ← L(w')
    if H(w'; sa, c^{j'}_PPH) = y
        Ret true
Ret false
```

```
PPH-Layer(W) :
γ ← ⌊ −log ( min_{w∈M} Pr[W = w] ) ⌋
for j = μ, ..., γ
    L_j ← (2^{−(j+1)}, 2^{−j}]
Ret {L_j : j ∈ [μ, γ]}
```

Fig. 5: The popularity-proportional hashing PBAS scheme $\mathsf{PBAS\text{-}PPH} = (\text{Reg-PPH}, \text{Chk-PPH})$, from a PBKDF $\mathbf{H}$ such that $\mathbf{H}(\cdot;\cdot,\mathsf{c}^{j}_{\mathrm{PPH}})$ costs $\mathsf{c}^{j}_{\mathrm{PPH}} = \mathsf{c}_{\mathrm{PPH}} \cdot 2^{\tilde{\mu}-(j+1)}$ to compute where $\mathsf{c}_{\mathrm{PPH}}$ is a parameter of the scheme, and $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = \tilde{\mu}$. The scheme uses layering $\mathsf{PPH\text{-}Layer}$.

6 Comparing the PBAS Approaches

In the last section we saw three different ways to provide typo-tolerant password-based authentication. Now we dig deeper into the trade-offs incurred by the different schemes, in an attempt to determine which provides the best time / security trade-off. We are most interested in the following question:

> *When balls are efficiently enumerable, can* PBAS-SS *ever provide a better time / security trade-off compared to* PBAS-BF/PBAS-PPH*?*

We will answer this question, in the negative, for the cases of using FRS or LHH. To do so, we will fix an error setting E with computationally enumerable balls (Definition 3), fix the time allotted to authentication, and show that for *any* error setting the popularity-proportional hashing PBAS PBAS-PPH provably provides better security than both PBAS-SS[FRS2] or PBAS-SS[LHH]. We will then discuss the conditions on error settings and attacker run time under which PBAS-BF offers a better trade-off still.

An incorrect interpretation of our results would be that sketches are useless. This would be the wrong takeaway for several reasons. First, our analysis will only be for specific sketches, not all sketches in general, and so answering our question in full generality remains an interesting open question (see Section 7). Second, even if the answer to our main question is negative, it only considers computationally enumerable balls, and many of the error correction settings motivating secure sketches have balls too large to be efficiently enumerated. Potential examples include high-entropy biometrics such as iris scans and fingerprints. Another reason is that we only consider settings where one can check that a correction is in fact correct, which allows the brute-force ball search. Finally, and most broadly, we do not consider information theoretic security — the original setting of most sketch constructions.

With these caveats in place, we turn to setting up a framework by which we can make apples-to-apples comparisons between the different PBAS schemes.

Qualities of typo-tolerant PBAS. There are three key axes upon which we compare efficacy of typo-tolerant PBAS schemes; correctness, security and run time. Correctness is readily assessed — it is straightforward to see that for any error setting E and δ-correct sketch $\mathsf{S} = (\mathsf{SS}, \mathsf{Rec})$, PBAS-SS[S] inherits the δ-correctness of underlying sketch. On the other hand, the two brute-force correction schemes, PBAS-BF and PBAS-PPH are perfectly correct. This highlights a bonus of the brute-force approach — if the correct password lies in the ball around a typo, these schemes will *always* recover the correct point.

The comparison between the time / security trade-offs incurred by the different approaches is less immediate. For a given PBAS, this trade-off is primarily dictated by the computational cost c we assign to $\mathbf{H}$ (corresponding, in practice, to picking larger security parameters for the slow hashing scheme). In order to compare different approaches, we fix a runtime budget RT for checking passwords, set each of the schemes' parameters to achieve maximal security subject to the run time constraint RT, and compare the security as measured by the message recovery game of Figure 4.

6.1 PBAS-BF versus FRS1 for Flat Distributions

As a warm up, we discuss the trade-off between PBAS-BF and PBAS[FRS1] where FRS1 = (FRS1-SS, FRS1-Rec) (Lemma 3).

Let $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ be an error setting, such that W is flat with $\mathrm{H}_\infty(W) = \mu$ and maximum ball size $\beta_{\max}$. For a given run time budget RT, setting $\mathsf{c}_{\mathrm{ss}} = \mathsf{RT}/2$ and $\mathsf{c}_{\mathrm{bf}} = \mathsf{c}_{\mathrm{ss}} \cdot \frac{2}{\beta_{\max}}$ ensures both schemes have equal run times. Let $\mathcal{A}$ be an adversary in game MR running in time at most T. Letting $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$, it follows that,

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF},\mathsf{E}}(\mathcal{A}) = \left(q_{\mathrm{ss}} \cdot \frac{\beta_{\max}}{2}\right) 2^{-\mu} \;\; ; \text{ and}$$

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS1]},\mathsf{E}}(\mathcal{A}) \leq \left(q_{\mathrm{ss}} \cdot \frac{\beta_{\max}}{\delta}\right) 2^{-\mu} \; .$$

The first statement arises since $\mathcal{A}$ can query at most $T/\mathsf{c}_{\mathrm{bf}} = q_{\mathrm{ss}} \cdot \frac{\beta_{\max}}{2}$ points in time T, each of which contributes weight $2^{-\mu}$ to its success probability. The latter follows since $\mathcal{B}$ can query at most $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$ points in time T; substituting this into the bound on q-conditional min-entropy given in Lemma 3 yields the claim. Since $0 < \delta < 1$ (and since δ represents the error probability of the sketch, in practice we would like δ to be small), this clearly illustrates that in terms of existing upper bounds $\mathsf{PBAS\text{-}BF}$ offers a significantly better time / security trade-off than $\mathsf{PBAS\text{-}SS[FRS1]}$. However, this does not rule out tighter upper bounds being found. To conclusively show that $\mathsf{PBAS\text{-}BF}$ offers the best performance, we would like to reverse the inequality sign in the above statement, and prove a *lower* bound on security for $\mathsf{PBAS\text{-}SS[FRS1]}$ that is larger than the upper bound on security for $\mathsf{PBAS\text{-}BF}$.

Let's unpick what this means. Let $\mathcal{B}$ be the optimal attacker in game MR against $\mathsf{PBAS\text{-}SS[FRS1]}$. We model the universal hash function family $\mathsf{F} : \mathcal{S} \times \{0,1\}^\ell \to \{0,1\}^{\log(\beta_{\max})+\log(1/\delta)}$ utilized by $\mathsf{FRS1}$ as a family of random oracles $\mathcal{H} = \{\mathsf{h}\}$; rather than including $\mathsf{sa} \xleftarrow{\$} \{0,1\}^\ell$ as part of the sketch, we now give access to the chosen random oracle $\mathsf{h} \xleftarrow{\$} \{\mathcal{H}\}$. We note that this modeling is conservative, since it can only make the attacker's job harder. With this in place, we may lower bound $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS1],E}}(\mathcal{A})$ via a balls-in-bins experiment. We represent each point $w \in \mathcal{M} = \mathrm{supp}(W)$ by a ball of weight $2^{-\mu}$, and associate each of the $2^{\log(\beta_{\max})+\log(1/\delta)} = \frac{\beta_{\max}}{\delta}$ points $y \in \mathrm{range}(\mathcal{H})$ with a bin. The choice of oracle $\mathsf{h} \xleftarrow{\$} \mathcal{H}$ fixes a 'throwing' of the balls into the bins, and the adversary's success probability is equal to the expected weight accrued when they are allowed to choose up to q_{ss} balls from each bin where $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$. The advantage is then calculated by taking the expectation of this total over the coins of the random oracle.

With this in place, all we must do is show that with overwhelming probability when we are allowed to choose at most q_{ss} balls from each bin, the resulting set contains at least $q_{\mathrm{ss}} \cdot \frac{\beta_{\max}}{2}$ balls. Intuitively this must be the case. We would *expect* each bin to contain $\frac{\delta \cdot |\mathrm{supp}(W)|}{\beta_{\max}}$ balls, and this value must be much larger than q_{ss} or the attack would be trivial. As such to *not* hit our total, a very high proportion of the bins must contain a number of balls which has diverged wildly from the mean. However, formalizing this intuition is non-trivial. A theorem statement to this end can be easily derived as a special case of those of Theorem 4; we defer the formal statement and analysis to Appendix D.

6.2 PPH versus FRS2 and LHH

In this section, we show that $\mathsf{PBAS\text{-}PPH}$ offers a better time / security trade-off than $\mathsf{PBAS\text{-}SS}$ implemented with the FRS and LHH sketch constructions of Section 3.

To facilitate the comparison, we first set the hashing cost parameter $\mathsf{c}_{\mathrm{PPH}}$ such that $\mathsf{PBAS\text{-}PPH}$ achieves the same runtime as $\mathsf{PBAS\text{-}SS}$ with associated hashing cost c_{ss}. With this parameter setting, $\mathsf{PBAS\text{-}SS}$ has checking run time $\mathsf{RT} = 2 \cdot \mathsf{c}_{\mathrm{ss}}$, so Lemma 4 implies that setting $\mathsf{c}_{\mathrm{PPH}} = 2 \cdot \mathsf{c}_{\mathrm{ss}}$ ensures $\mathsf{PBAS\text{-}PPH}$ achieves run time RT also. With this in place, we now upper-bound the success probability of an optimal attacker against $\mathsf{PBAS\text{-}PPH}$ with these parameters; the proof is given in Appendix D.2.

Lemma 5. *Let $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ be an error setting, where $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = \tilde{\mu}$. Let $\mathsf{PBAS\text{-}PPH}$ be the population-proportional hashing scheme where random oracle $\mathbf{H}$ has associated cost $\mathsf{c}_{\mathrm{PPH}} = 2 \cdot \mathsf{c}_{\mathrm{ss}}$. Let $\mathcal{A}$ be an adversary in game $\mathrm{MR}^{\mathcal{A}}_{\mathsf{PBAS\text{-}PPH,E}}$ running in time at most T. Then,*

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) \le q_{\mathrm{ss}} \cdot 2^{-\tilde{\mu}} .$$

where $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$.

To give the above term some context, consider the equivalent upper bounds on success probability for sketch-based schemes $\mathsf{PBAS\text{-}SS[FRS2]}$, and $\mathsf{PBAS\text{-}SS[LHH]}$ (which are derived by substituting the parameters of the error setting into Theorem 1 and Theorem 3 respectively).[3] For any adversary $\mathcal{B}$ running in time at most T it holds that,

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS2],E}}(\mathcal{B}) \le q_{\mathrm{ss}} \cdot \frac{\mathrm{H}_0(W) \cdot 2^{-(\tilde{\mu}-1)}}{\delta} \text{ , and}$$

[3] Since they are stated in terms of $\tilde{\mu}$, we use the (looser) upper bounds here for ease of comparison. It is straightforward to derive similar statements showing the tighter bound are poorer too; see the proof of Theorem 4.

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[LHH],E}}(\mathcal{B}) \leq q_{\mathrm{ss}} \cdot \frac{2^{-(\tilde{\mu}-1)}}{\delta} \ .$$

By comparison with Lemma 5, it is immediately clear that PBAS-PPH enjoys better security upper bounds than either construction. Of course it could be that the upper bounds on the sketches can be improved.

We therefore, in the following theorem, lower bound the success probability of an optimal attack against PBAS-SS[FRS2] and PBAS-SS[LHH] in terms of the advantage of any adversary against PBAS-PPH. This rules out improving the upper bounds enough to make the sketch-based schemes better than PBAS-PPH. We first state the theorem, then discuss its significance.

Theorem 4. *Let* $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ *be an error setting with* $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = \tilde{\mu}$*. Let* $\Pi\text{-}\mathsf{S} = (\Pi\text{-}\mathsf{SS}, \Pi\text{-}\mathsf{Rec})$ *be the secure sketch for the same error setting where* $\Pi \in \{\mathsf{FRS2}, \mathsf{LHH}\}$*, achieving* $1-\delta$ *correctness for some* $0 < \delta < 1$*. We model the (strongly) universal hash functions used by the sketch as random oracles. Let* PBAS-SS[Π-S] *be the sketch-assisted* PBAS *built from* Π-S*, using random oracle* $\mathbf{H}$ *with associated cost* c_{ss}*. Let* PBAS-PPH *be the popularity-proportional hashing* PBAS *for this error setting, with random oracle* $\mathbf{H}'$ *with associated cost* $\mathsf{c}_{\mathrm{PPH}}$ *set such that* $\mathsf{RT}(\text{Chk-SS}, \mathsf{c}_{\mathrm{ss}}) \geq \mathsf{RT}(\text{Chk-PPH}, \mathsf{c}_{\mathrm{PPH}})$*. Then for any adversary* $\mathcal{A}$ *against* PBAS-PPH *running in time at most* T*, there exists an adversary* $\mathcal{B}$ *against* PBAS-SS[Π-S] *such that*

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[\Pi\text{-}S],E}}(\mathcal{B}) + \left(\frac{e \cdot \delta}{2}\right)^{q_{\mathrm{ss}}}$$

and, moreover, $\mathcal{B}$ *runs in time* T *and so can make at most* $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$ *queries.*

We have stated the theorem in the form of a reduction to highlight that PBAS-PPH provides at least as good a time / security trade-off as the seemingly more sophisticated sketch-based scheme. Given that q_{ss} will be large (this is the number of hash computations an attacker can make), then, provided that $\delta < 2/e \approx 0.736$ the second term in the bound is infinitesimally far from zero. Since δ represents the error rate of the sketch, in practice any useable sketch will require δ much smaller than .736.

The proof of the theorem proceeds by specifying a concrete adversary $\mathcal{B}$ against PBAS-SS[Π-SS] for $\Pi \in \{\mathsf{FRS2}, \mathsf{LHH}\}$, where the underlying (strongly) universal hash function family is modeled as a family of random oracles $\mathcal{H} = \{\mathsf{h}\}$. It works as one would expect: the adversary is given some sketch s and access to the oracle h used in the computation of the sketch. The attack queries the q_{ss} heaviest points in the preimage set

$$\mathcal{X}_s = \{w \in \mathrm{supp}(W) \ : \ \Pr\left[W = w \wedge \Pi\text{-}\mathsf{SS}^{\mathsf{h}}(W) = s \right] > 0\}$$

to the PBKDF $\mathbf{H}$, where $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$. This is the optimal attack.

We note that $\mathcal{B}$ need not compute the entire preimage set before submitting his guesses to the oracle $\mathbf{H}$ — rather his most efficient strategy is to compute the hashes of candidate points under h in descending order of weight, looking for points which lie in the preimage set. Intuitively this will be efficient because, assuming the sketch behaves uniformly, we would expect to find preimage set points at fairly regular intervals. For example, if (for simplicity) the sketch was simply $\mathsf{h}(w) = y$, then the expected run time for $\mathcal{B}$ to find q_{ss} matches (over the coins of h) is $q_{\mathrm{ss}} \cdot |y|$ computations of h.

The proof then must show a lower bound on the success of $\mathcal{B}$. This analysis turns out to be quite tricky, involving a nuanced balls-in-bins argument. We make things easier by targeting only a rather loose lower bound that suffices to show the desired relationship with PBAS-PPH. We believe that better lower bounds can be found. Better lower bounds would signify an even bigger gap between the security of PBAS-PPH and PBAS-SS, making PBAS-PPH look even better in comparison.

We note that while the above result shows that PBAS-PPH always offers a better time / security trade-off than sketch-based schemes using FRS or LHH, the same cannot be shown to hold for PBAS-BF. For example, consider an error setting such that W consists of 2^{49} points of weight 2^{-50} and 2^{99} points of weight 2^{-100}, for which all balls contain a single point, except for one large ball containing 2^{20} of the lower weight points. As such $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = 50$, and so by Theorems 2 and 3 it is easy to see that the security of the sketch-based schemes will degrade linearly and gracefully as T grows. On the other hand, the huge ball of 2^{20} points means that for matching run-times we must set $\mathsf{c}_{\mathrm{bf}} = 2^{-19} \cdot \mathsf{c}_{\mathrm{ss}}$ — so low that security for PBAS-BF (at least initially; see Section 6.3) degrades dramatically compared to PBAS-SS.

This counterexample may be contrived, but for more realistic distributions there remains a technical challenge in comparing PBAS-BF and PBAS-SS for FRS and LHH directly. The fact that the latter schemes are parameterized by $\tilde{\mu} = \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)$ means that the natural derived security bounds are in terms of $\tilde{\mu}$ also, whereas for PBAS-BF, security is dictated by the sum of the weights of the points at the head of the distribution. Therefore any balls-in-bins analysis of the form described above involves a complicated comparison between two somewhat orthogonal terms. To overcome this, we can use PPH (whose success probability is also a function of $\tilde{\mu}$) as a bridge, first invoking Theorem 4 to show that PBAS-PPH offers a better time / security trade-off than the sketch-based PBAS and then assessing, using results in Section 6.3, whether PBAS-BF offers a better trade-off still.

6.3 Brute-force Checking versus PPH

In the following theorem, we quantify precisely the conditions on an error setting E under which PBAS-PPH represents a better time / security trade-off than PBAS-BF. We fix a run time RT for the checking algorithms of both schemes, and set the associated hashing cost c_{bf} and hashing cost parameter $\mathsf{c}_{\mathrm{PPH}}$ in a way that ensures both schemes work within this run time. We then consider the success probabilities of optimal adversaries attacking the schemes, both running in some time T.

The following theorem formally captures our comparison of the two schemes. Roughly speaking, the result indicates that there exists a crossover point: for T smaller than this point, PBAS-PPH provides better security than PBAS-BF, and for T larger than this point, the inverse is true. The crossover point is dictated by the error setting. As we discuss in more detail below, the crossover point for typical error distributions seen with human-chosen passwords is actually pretty small, meaning that PBAS-BF would appear to dominate for distributions of practical interest. Whether PBAS-PPH can be improved is an open question.

Theorem 5. *Let $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ be an error setting, where $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = \tilde{\mu}$ and the largest ball is of size $\beta_{\max}$. Let PBAS-BF = (Reg, Chk-BF) be the brute-force PBAS for this error setting, using oracle $\mathbf{H}$ with associated cost c_{bf} and run time budget RT. Let PBAS-PPH = (Reg-PPH, Chk-PPH) be the popularity-proportional hashing PBAS for this error setting using an oracle $\mathbf{H}'$ with associated cost parameter $\mathsf{c}_{\mathrm{PPH}}$ set such that $\mathsf{RT}(\text{Chk-BF}, \mathsf{c}_{\mathrm{bf}}) \geq \mathsf{RT}(\text{Chk-PPH}, \mathsf{c}_{\mathrm{PPH}})$. Let $\mathcal{A}$ and $\mathcal{B}$ be optimal attackers in games $\mathrm{MR}^{\mathcal{A}}_{\mathsf{PBAS\text{-}PPH,E}}$ and $\mathrm{MR}^{\mathcal{B}}_{\mathsf{PBAS\text{-}BF,E}}$ respectively running in time T. Let $q_{\mathrm{bf}} = T/\mathsf{c}_{\mathrm{bf}} = T \cdot \beta_{\max}/\mathsf{RT}$. Then if T is such that*

$$T \leq \left(2^{-\mathrm{H}^{q_{\mathrm{bf}}}_{\infty}(W)} \cdot 2^{(\tilde{\mu}-1)}\right) \cdot \mathsf{RT} \ ,$$

it holds that $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{B})$. For all error settings such that,

$$T \geq \left(2^{-\mathrm{H}^{q_{\mathrm{bf}}}_{\infty}(W)} \cdot 2^{\tilde{\mu}} + 1\right) \cdot \mathsf{RT} \ ,$$

it holds that $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{B}) \leq \mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A})$.

The proof works by upper and lower bounding the success probability of an optimal attacker against PPH, and comparing this to the success probability of an optimal attacker against the brute-force checker. The proof is given in Appendix D.2.

At a high level, the first bound in the theorem shows that PBAS-PPH favors error settings for which the weight of points decreases slowly (relative to the attack run time) as we move down through the distribution starting from the heaviest point. In such error settings PBAS-PPH allows us to securely correct larger balls — and accordingly more errors — than brute-force checking, provided balls are constructed such that the fuzzy min-entropy is high. This latter requirement is not much of a restriction, since a well designed error setting will seek to maximize the utility for a given level of security by defining balls to have many points but low aggregate mass. For most such error settings, while there will be a point after which PBAS-BF offers the better time / security trade-off, this will be for an attack run time too large to be of concern. This class of distributions includes those described in Section 6.2 for which brute-force checking degrades security dramatically.

On the other hand, the second bound shows that if the weight of points decreases quickly as we move down through the distribution, then PBAS-BF offers the better time / security trade-off. Intuitively this is because, as the weight of points decreases, the gap between the (higher) hashing cost under PPH decreases until it is, in fact, lower than the hashing cost used with PBAS-BF. As such the crossover point after which brute-force checking offers the better trade-off falls within the attack run times of concern. Since password distributions are typically very 'top-heavy', with the weights of points decreasing rapidly to leave a long tail, they fall into the class for which brute-force checking offers the better time / security trade-off.

The theorem gives both upper and lower bounds on T, with a small gap between them, meaning the crossover point is not precisely pinned down. This is due to a small amount of slack in upper and lower bounding the success probability of an optimal attacker against PBAS-PPH for general error settings. For specific error settings, one can sharpen the analysis.

7 Conclusion

In this work we investigated error correction for cryptographic secrets in the known-distribution setting. Using typo-tolerant password checking as a guiding case study, we provided several improvements on both theory and practice. On the theory side, we introduced a new information-theoretic security goal for secure sketches that better matches the needs of applications that may allow an attacker to make multiple guesses about the secret. While for high-entropy settings the distinction is moot, for passwords it is critical. We then provided analysis of the best known schemes in this setting, due to Fuller et al. [15].

Our first main contribution was the design and analysis of a new secure sketch construction, the layer-hiding hash (LHH). We proved that it provides better security than prior schemes. We then introduced a new distribution-sensitive brute-force based technique called property-proportional hashing (PPH) that, unlike the prior brute-force checking approach of Chatterjee et al. [12], varies the run time of the hash function according to the popularity of the password being hashed.

We gave a framework for comparing different approaches to typo-tolerant authentication, and used it to show that PPH outperforms sketch-based solutions to typo-tolerance, even when using the layer-hiding hash sketch. We determine the conditions under which PPH improves on the brute-force checking approach of Chatterjee et al. [12], along with the conditions under which their simpler brute-force checking offers a better trade-off. Put all together, our results indicate that brute-force based approaches perform better than the best known secure sketches. We now finish with a few important points and open questions.

Complexity beyond time. Most in-use slow hashes only target extending the time required for a single hash computation. Increasingly, however, practitioners are transitioning to slow hashing that targets memory-hardness [1–3, 6, 26], meaning that computing a hash requires that the space-time complexity (the product of memory and time utilized) is lower bounded. Our constructions work with memory-hard hash functions as well, though our comparisons of different approaches currently only considers time complexity. Future work may also consider parallel computation models, which could be useful when a password checking system can use multiple cores to simultaneously check multiple possible corrections.

Additional applications. While we motivated and used as a running example the setting of password-based authentication, our constructions are generic. They hold for any distribution-sensitive setting in which one has efficiently enumerable balls (the same general setting considered by FRS). The FRS, LHH, and PPH approaches will not work for error settings with large balls, such as attempting to correct large Hamming or edit distances. In these contexts, existing secure sketch constructions [13, 14] seem to be the only solution. We note that their entropy loss is significantly worse than the FRS or LHH constructions, and so they would not seem useful for passwords.

We have focused on authentication, but our results and comparisons are applicable to any cryptographic application in which noisy secrets are used to derive a key for which one can efficiently test correctness. This includes at least all authentication primitives, such as message authentication codes and authenticated encryption. Similarly, our new sketch constructions can also be used to build a fuzzy extractor using the construction from [13], which inherits the security improvement over the fuzzy extractor from FRS.

Secure sketches in the multi-guess setting. In the previous section, we proved that PBAS-SS never offers a better time / security trade-off than PBAS-PPH/PBAS-BF when implemented with the FRS sketches, and the new — and nearly optimal, in the single-guess setting — LHH sketch. The key open question is whether *any* distribution-sensitive secure sketch can perform better in this context. The challenge is to design a sketch which preserves much of the min-entropy of the underlying distribution in the face of an attacker who can make q guesses for varying and large values of q. This is an important requirement in many practical settings, yet has been overlooked in existing literature.

Intuitively, the correctness requirement means that the sketch must include sufficient information to disambiguate between points in the heaviest ball(s). As such any efficiently-computable sketch — (we disregard those which, for example, solve an NP-hard problem to create an optimal arrangement of points into sketch preimage sets) — is likely to leak more information than is strictly necessary for correctness in less heavy balls. This additional leakage can then be exploited by an attacker. More generally we would expect that the larger q is, the wider the gap between the security of a sketch $\mathsf{S} = (\mathsf{SS}, \mathsf{Rec})$ for that error setting $\tilde{\mathrm{H}}^{q}_{\infty}(W|\mathsf{SS}(W))$, and the theoretical best case security bound $\mathrm{H}^{q,\mathrm{fuzz}}_{t,\infty}(W) - \log(1-\delta)$.

We conjecture that for a significant class of error settings — especially those such as passwords, which inherently contain large balls — no efficient distribution sensitive secure sketch can offer a better time / security trade-off than brute-force based approaches. Indeed it seems likely that any intuition leading to an improvement in secure sketch performance over LHH may also be utilized to create a brute-force approach which improves on PBAS-PPH (similar to the way in which the same layered approach is used by both LHH and PPH, with better performance in the latter). Refining and improving upon the brute-force based approaches described here is an interesting open problem.

Acknowledgements

This work was supported in part by NSF grants 1619158, 1319051, 1314568, 1514163, United States Army Research Office (ARO) grant W911NF-16-1-0145, EPSRC and UK government grant EP/K035584/1, and gifts from VMware Labs, Google, and Microsoft.

References

1. Alwen, J., Blocki, J.: Efficiently computing data-independent memory-hard functions. In: Advances in Cryptology – CRYPTO (2016)
2. Alwen, J., Chen, B., Kamath, C., Kolmogorov, V., Pietrzak, K., Tessaro, S.: On the complexity of scrypt and proofs of space in the parallel random oracle model. In: Advances in Cryptology –EUROCRYPT (2016)
3. Alwen, J., Chen, B., Pietrzak, K., Reyzin, L., Tessaro, S.: On the complexity of scrypt and proofs of space in the parallel random oracle model. In: Advances in Cryptology –EUROCRYPT (2017)
4. Bellare, M., Ristenpart, T., Tessaro, S.: Multi-instance security and its application to password-based cryptography. In: Advances in Cryptology – CRYPTO 2012, pp. 312–329. Springer Berlin Heidelberg (2012)
5. Bennett, C.H., Brassard, G., Robert, J.M.: Privacy amplification by public discussion. SIAM journal on Computing 17(2), 210–229 (1988)
6. Biryukov, A., Dinu, D., Khovratovich, D.: Argon and argon2: password hashing scheme. Tech. rep., Technical report (2015)
7. Bonneau, J.: Guessing human-chosen secrets. Ph.D. thesis, University of Cambridge (May 2012), `http://www.cl.cam.ac.uk/~jcb82/doc/2012-jbonneau-phd_thesis.pdf`
8. Bonneau, J.: The science of guessing: analyzing an anonymized corpus of 70 million passwords. In: IEEE Symposium on Security and Privacy (SP). pp. 538–552. IEEE (2012)
9. Bowes, R.: Skull Security, Passwords, `https://wiki.skullsecurity.org/Passwords`

10. Boztas, S.: Entropies, guessing, and cryptography. Department of Mathematics, Royal Melbourne Institute of Technology, Tech. Rep 6, 2–3 (1999)
11. Castelluccia, C., Dürmuth, M., Perito, D.: Adaptive password-strength meters from markov models. In: NDSS (2012)
12. Chatterjee, R., Athalye, A., Akhawe, D., Juels, A., Ristenpart, T.: In: pASSWORD tYPOS and How to Correct Them Securely. Security and Privacy (SP), 2015 IEEE Symposium on (2016)
13. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In: Cachin, C., Camenisch, J. (eds.) Eurocrypt 2004. pp. 523–540. Springer-Verlag (2004), lNCS no. 3027
14. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors. In: Security with Noisy Data, pp. 79–99. Springer (2007)
15. Fuller, B., Reyzin, L., Smith, A.: When are fuzzy extractors possible? In: Advances in Cryptology – ASIACRYPT. pp. 277–306. Springer (2016)
16. Juels, A., Ristenpart, T.: Honey Encryption: Beyond the Brute-force Barrier. In: Advances in Cryptology – EUROCRYPT. pp. 523–540. Springer (2014)
17. Juels, A., Wattenberg, M.: A fuzzy commitment scheme. In: Tsudik, G. (ed.) Sixth ACM Conference on Computer and Communications Security. pp. 28–36. ACM Press (1999)
18. Kaliski, B.: PKCS #5: Password-Based Cryptography Specification Version 2.0 (2000), rFC 2289
19. Ma, J., Yang, W., Luo, M., Li, N.: A study of probabilistic password models. In: Security and Privacy (SP), 2014 IEEE Symposium on. pp. 689–704. IEEE (2014)
20. Mehler, A., Skiena, S.: Improving usability through password-corrective hashing. In: String Processing and Information Retrieval. pp. 193–204. Springer (2006)
21. Melicher, W., Ur, B., Segreti, S.M., Komanduri, S., Bauer, L., Christin, N., Cranor, L.F.: Fast, lean and accurate: Modeling password guessability using neural networks. In: Proceedings of USENIX Security (2016)
22. Monrose, F., Reiter, M.K., Li, Q., Wetzel, S.: Cryptographic key generation from voice. In: Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on. pp. 202–213. IEEE (2001)
23. Monrose, F., Rubin, A.: Authentication via keystroke dynamics. In: Proceedings of the 4th ACM conference on Computer and communications security. pp. 48–56. ACM (1997)
24. Monrose, F., Rubin, A.D.: Keystroke dynamics as a biometric for authentication. Future Generation computer systems 16(4), 351–359 (2000)
25. Ostrovsky, R., Rabani, Y.: Low distortion embeddings for edit distance. Journal of the ACM (JACM) 54(5), 23 (2007)
26. Percival, C., Josefsson, S.: The scrypt password-based key derivation function. Tech. rep. (2016)
27. PKCS #5: Password-Based Cryptography Standard (RFC 2898). RSA Data Security, Inc. (Sep 2000), version 2.0
28. Renner, R., Wolf, S.: The exact price for unconditionally secure asymmetric cryptography. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 109–125. Springer (2004)
29. Siegler, M.: One Of the 32 Million With A RockYou Account? You May Want To Change All Your Passwords. Like Now. Tech Crunch (14 Dec 2009)
30. Škoric, B., Tuyls, P.: An efficient fuzzy extractor for limited noise. In: Symposium on Information Theory in the Benelux. pp. 193–200 (2009)
31. Wadhwa, T.: Why Your Next Phone Will Include Fingerprint, Facial, and Voice Recognition. Forbes (29 March 2013)
32. Weir, M., Aggarwal, S., de Medeiros, B., Glodek, B.: Password cracking using probabilistic context-free grammars. In: IEEE Symposium on Security and Privacy (SP). pp. 162–175 (2009)
33. Wheeler, D.L.: zxcvbn: Low-budget password strength estimation. In: Proc. USENIX Security (2016)
34. Yao, F., Yin, Y.: Design and analysis of password-based key derivation functions. In: Topics in Cryptology – CT-RSA 2005, pp. 245–261. Springer (2005)
35. Zhao, Q., Liu, F., Zhang, L., Zhang, D.: A comparative study on quality assessment of high resolution fingerprint images. In: International Conference on Image Processing (ICIP). pp. 3089–3092 (2010)

A The Approximately Known Distribution Setting

Our constructions target the distribution-sensitive setting. An important question to ask is what happens when the construction incorrectly estimates the distribution? In this case, the tuning used by FRS, LLH, and PPH may be off and an adversary that knows the true distribution will get some improvement in attacks. We show how to upper bound the effect of this using Lemma 5 as an illustrative example — the technique may readily be extended to other results. Let PBAS-PPH, E and $\mathcal{A}$ be as in Lemma 5, and recall that whatever set of queries $\mathcal{Q}^*$ the adversary $\mathcal{A}$ makes in game MR against $\mathsf{PBAS\text{-}PPH}$, the restriction on $\mathcal{A}$ to run in time T coupled with the fact that hashing a point in layer j incurs costs $\mathsf{c}^j_{\mathsf{PPH}} = \mathsf{c}_{\mathrm{ss}} \cdot 2^{\tilde{\mu}-j}$ implies that $\sum_{w \in \mathcal{Q}^*} \mathsf{c}_{\mathrm{ss}} \cdot 2^{\tilde{\mu}-(\mathsf{L}(w))} \leq T$, where L denotes the algorithm which maps points to their estimated layers. Rearranging and setting $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$ yields,

$$\sum_{w \in \mathcal{Q}^*} 2^{-\mathsf{L}(w)} \leq q_{\mathrm{ss}} \cdot 2^{-\tilde{\mu}} \ .$$

Now let L' be the function (in the possession of an adversary) which maps a point to its true layer. $\mathcal{A}$ can exploit points $w \in \mathcal{S}$ for which PPH *underestimates* the probability of w at the point of registration — that is to say w for which $\mathsf{L}(w) > \mathsf{L}'(w)$ — to query passwords for a lower cost than they would incur in the true model. Let $\alpha = \max_{w \in \mathcal{M}}(\mathsf{L}(w) - \mathsf{L}'(w))$. We can ensure that this difference is finite by modifying the registration algorithm so that if a user registers a password w which (due to a modeling error in L) is reported to have probability 0, it is hashed as if it lies in the layer of least probable points L_γ. Then for all $w \in \mathcal{M}$

$$\Pr[W = w] \leq 2^{-\mathsf{L}'(w)} \leq 2^{-\mathsf{L}(w)} \cdot 2^{\alpha} \ .$$

This immediately implies that whatever set of queries $\mathcal{Q}^*$ the adversary $\mathcal{A}$ chooses to output,

$$\begin{aligned}\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) &= \sum_{w \in \mathcal{Q}^*} \Pr[W = w] \\ &\leq \sum_{w \in \mathcal{Q}^*} 2^{-\mathsf{L}'(w)} \\ &\leq \sum_{w' \in \mathcal{Q}^*} 2^{-\mathsf{L}(w)} \cdot 2^{\alpha} \\ &= q_{\mathrm{ss}} \cdot 2^{\alpha - \tilde{\mu}} \ ,\end{aligned}$$

so we lose a factor of at most 2^α in security. We emphasise that this upper bound is loose; more accurately quantifying security loss in the case of approximately known distributions remains an interesting open problem.

B Proofs from Section 2

B.1 Proof of Lemma 1

Proof:

$$\begin{aligned}2^{-\tilde{\mathrm{H}}^q_\infty(W|Z)} &= \left(\sum_z \max_{w_1,\ldots,w_q} \sum_{i=1}^q \Pr[W = w_i \mid Z = z] \Pr[Z = z]\right) \\ &= \left(\sum_z \max_{w_1,\ldots,w_q} \sum_{i=1}^q \Pr[W = w_i \wedge Z = z]\right) \\ &\leq \left(\sum_z \max_{\substack{w_1,\ldots,w_q \\ z_1,\ldots,z_q}} \sum_{i=1}^q \Pr[W = w_i \wedge Z = z_i]\right) \\ &\leq 2^{\mathrm{H}_0(Z)} \cdot 2^{-\mathrm{H}^q_\infty(W,Z)}.\end{aligned}$$

Taking negative logs of both sides implies the result.
▮

C Proofs from Section 3

C.1 Proof of Lemma 2

Proof: Let $\mathcal{A}$ be an unbounded adversary who on input s proceeds as follows: $\mathcal{A}$ finds $\tilde{w}_1, \ldots, \tilde{w}_q$ such that the weight of the union of their balls achieves the q-fuzzy min-entropy. $\mathcal{A}$ then computes $w'_i \xleftarrow{\$} \mathsf{Rec}(s, \tilde{w}_i)$ for each $i \in [q]$, and submits $(w'_1, \ldots, w'_q)$ as its vector of guesses. Then,

$$\begin{aligned}
\Pr[\mathcal{A} \text{ wins }] &= \Pr\left[\bigvee_{i=1}^{q} W = w'_i\right] \\
&= \sum_{i=1}^{q} \Pr\left[W = w'_i \bigwedge W \notin \{w'_1, \ldots, w'_{i-1}\}\right] \\
&\geq \sum_{i=1}^{q} \Pr\left[W = w'_i \bigwedge W \notin \{\cup_{i=1}^{i-1} B_t(\tilde{w}_i)\}\right] \\
&\geq \sum_{i=1}^{q} \Pr\left[W = w'_i \bigwedge W \in B_t(\tilde{w}_i)/\{\cup_{i=1}^{i-1} B_t(\tilde{w}_i)\}\right] \\
&\geq \sum_{i=1}^{q} (1-\delta) \Pr\left[W \in B_t(\tilde{w}_i)/\{\cup_{i=1}^{i-1} B_t(\tilde{w}_i)\}\right] \\
&= (1-\delta) \sum_{w' \in \cup_{i=1}^{q} B_t(\tilde{w}_i)} \Pr[W = w'] \\
&= (1-\delta)\ 2^{-\mathrm{H}_{t,\infty}^{q,\mathrm{fuzz}}(W)} .
\end{aligned}$$

The first inequality follows from the fact that each $w'_i \in B_t(\tilde{w}_i)$, the next inequality from conditioning on the event that $W \in B_t(\tilde{w}_i)$, the final inequality from the correctness of the sketch, and the final equality from the definition of q-fuzzy min-entropy.
▮

C.2 Proof of Lemma 3

In subsequent proofs, we let $\mathsf{SS}(\cdot\ ;\ \mathsf{sa})$ denote the distribution of the output of SS in the event that sa is the chosen salt.

Proof: Let $R = \{0,1\}^{\log(\beta_{\max})+\log(1/\delta)}$, and let $\mathcal{X}_{(y,\mathsf{sa})} = \{w \in W\ :\ \mathsf{F}(w;\mathsf{sa}) = y\}$. We wish to upper bound:

$$\begin{aligned}
2^{-\tilde{\mathrm{H}}_{\infty}^{q}(W|\mathsf{FRS1\text{-}SS}(W))} &= \sum_{s} \max_{w_i,\ldots,w_q} \sum_{i=1}^{q} \Pr[W = w_i \wedge \mathsf{FRS1\text{-}SS}(W) = s] \\
&= \sum_{\mathsf{sa}} \sum_{y \in R} \max_{w_1,\ldots,w_q} \sum_{i=1}^{q} \Pr[W = w_i \wedge \mathsf{sa} \text{ chosen } \wedge \mathsf{FRS1\text{-}SS}(W;\mathsf{sa}) = (y,\mathsf{sa})] \\
&= \sum_{\mathsf{sa}} \sum_{y \in R} \max_{w_1,\ldots,w_q} \sum_{i=1}^{q} \Pr[\mathsf{FRS1\text{-}SS}(w_i;\mathsf{sa}) = (y,\mathsf{sa})] \cdot \Pr[W = w_i] \cdot 2^{-\ell} \\
&= \sum_{\mathsf{sa}} \sum_{y \in R} \max_{\substack{w_1,\ldots,w_q \\ \in \mathcal{X}_{(y,\mathsf{sa})}}} \sum_{i=1}^{q} \Pr[W = w_i] \cdot 2^{-\ell} ,
\end{aligned}$$

where the second to last equality follows from an application of Bayes' rule, and the final equality is because $\Pr[\mathsf{FRS1\text{-}SS}(w;\mathsf{sa}) = y] = 1$ if $w \in \mathcal{X}_{(y,sa)}$, and $\Pr[\mathsf{FRS1\text{-}SS}(w;\mathsf{sa}) = y] = 0$ otherwise.

Fix a salt sa, and notice that since W is flat with $\Pr[W = w] = 2^{-\mu}$, we may rewrite the inner sum

$$\begin{aligned}
\sum_{y \in R} \max_{\substack{w_1,\ldots,w_q \\ \in \mathcal{X}_{(y,\mathsf{sa})}}} \sum_{i=1}^{q} \Pr[W = w_i] &\leq \sum_{y \in R} q \cdot 2^{-\mu} \\
&= q \cdot 2^{\log(\beta_{\max})+\log(1/\delta)} \cdot 2^{-\mu} \\
&= q \cdot \frac{\beta_{\max} \cdot 2^{-\mu}}{\delta}
\end{aligned}$$

$$= q \cdot \frac{2^{-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)}}{\delta} \; .$$

where the final equality follows since $2^{-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)} = \beta_{\max} \cdot 2^{-\mu}$. Since this holds for each $\mathsf{sa} \in \{0,1\}^{\ell}$, taking the expectation of the choice of sa followed by logs of both sides implies the result.
▮

C.3 Proof of Theorem 2

Proof: Let $R_j = \{0,1\}^{j-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)+\log(1/\delta)+1}$ for each $j \in [\mu, \lambda-1]$. Let $\mathcal{X}_{(y,\mathsf{sa}_j,j)} = \{w \in L_j \;:\; \mathsf{F}_j(w;\mathsf{sa}_j) = y\}$. We wish to upper bound,

$$2^{-\tilde{\mathrm{H}}^q_\infty(W|\mathsf{FRS2\text{-}SS}(W))} = \sum_{j=\mu}^{\lambda} \sum_{\mathsf{sa}_j} \sum_{y \in R_j} \max_{w_1,\ldots,w_q} \sum_{i=1}^{q} \Pr[\, W = w_i \wedge \mathsf{FRS2\text{-}SS}(W) = (y, \mathsf{sa}_j, j)\,]$$

$$= \sum_{j=\mu}^{\lambda} \sum_{\mathsf{sa}_j} \sum_{y \in R_j} \max_{w_1,\ldots,w_q} \sum_{i=1}^{q} \Pr[\, W = w_i \wedge W \in L_j \wedge \mathsf{FRS2\text{-}SS}(W;\mathsf{sa}_j) = (y, \mathsf{sa}_j, j))\,] \cdot 2^{-\ell_j}$$

$$= \sum_{j=\mu}^{\lambda} \sum_{\mathsf{sa}_j} \sum_{y \in R_j} \max_{\substack{w_1,\ldots,w_q \\ w_i \in L_j}} \sum_{i=1}^{q} \Pr[\, \mathsf{FRS2\text{-}SS}(w_i;\mathsf{sa}_j) = y\,] \cdot \Pr[\, W = w\,] \cdot 2^{-\ell_j}$$

$$= \sum_{j=\mu}^{\lambda} \sum_{\mathsf{sa}_j} \sum_{y \in R_j} \max_{\substack{w_1,\ldots,w_q \\ w_i \in \mathcal{X}_{(y,\mathsf{sa}_j,j)}}} \sum_{i=1}^{q} \Pr[\, W = w_i\,] \cdot 2^{-\ell_j} \;;.$$

The second equality follows from conditioning on the event that sa_j is the chosen salt. The third equality follows from an application of Bayes' rule and the fact that $\Pr[\, W \in L_j \mid W = w\,] = 1$ if $w \in L_j$ and 0 otherwise. The final equality follows from the fact that $\Pr[\, \mathsf{FRS2\text{-}SS}(W, \mathsf{sa}_j, j) = (y, \mathsf{sa}_j)\,] = 1$ if $w \in \mathcal{X}_{(y,\mathsf{sa}_j,j)}$ and 0 otherwise. Fix an index $j \in [\mu, \lambda-1]$ and salt sa_j, and consider the sum,

$$\sum_{y \in R_j} \max_{\substack{w_1,\ldots,w_q \\ w_i \in \mathcal{X}_{(y,\mathsf{sa}_j,j)}}} \sum_{i=1}^{q} \Pr[\, W = w_i\,] \; .$$

Notice that since each $w \in L_j$ lies in precisely one preimage set, this sum will be maximised when each preimage set $\mathcal{X}_{(y,\mathsf{sa}_j,j)}$ for $y \in R_j$ contains precisely q of the points in $L_j(q \cdot |R_j|)$, (recall that $L_j(q \cdot |R_j|)$ denotes the set of the $\min\{q', |L_j|\}$ heaviest points in layer L_j). It follows that,

$$\sum_{y \in R_j} \max_{\substack{w_1,\ldots,w_q \\ w_i \in \mathcal{X}_{(w,\mathsf{sa}_j,j)}}} \sum_{i=1}^{q} \Pr[\, W = w_i\,]$$
$$\leq \Pr[\, W \in L_j(q \cdot |R_j|)\,] \; .$$

Furthermore, notice that since for each $w \in L_\lambda$ $\mathsf{FRS2\text{-}SS}(w) = (w, \perp, \lambda)$ it holds that $\mathcal{X}_{(w,\perp,\lambda)} = \{w\}$. Therefore the above equation becomes,

$$\sum_{w \in L_\lambda} \max_{\substack{w_1,\ldots,w_q \\ w_i \in \mathcal{X}_{(w,\perp,\lambda)}}} \sum_{i=1}^{q} \Pr[\, W = w_i\,]$$
$$= \sum_{w \in L_\lambda} \Pr[\, W = w\,]$$
$$= \Pr[\, W \in L_\lambda\,] \; .$$

Finally averaging over choice of sa_j for each $j \in [\mu, \lambda-1]$, and summing over the layers gives,

$$2^{-\tilde{\mathrm{H}}^q_\infty(W|\mathsf{FRS2\text{-}SS}(W))} \leq \Pr[\, W \in L_\lambda\,] + \sum_{j=\mu}^{\lambda-1} \Pr[\, W \in L_j(q \cdot |R_j|)\,] \;;$$

taking logs of both sides implies the result. To illustrate how this bound is tighter in the case that $q = 1$ than that given in Theorem 1, we show how to convert the above bound into that given in [15], following their proof. Let $\mathrm{H}_\infty(W) = \mu$. Then,

$$\begin{aligned}
2^{-\tilde{\mathrm{H}}_\infty(W|\mathsf{FRS2\text{-}SS}(W))} &\le \Pr[\, W \in L_\lambda \,] + \sum_{j=\mu}^{\lambda-1} \Pr[\, W \in L_j(|R_j|)\,] \\
&\le |L_\lambda| \cdot 2^{-\lambda} + \sum_{j=\mu}^{\lambda-1} |R_j| \cdot 2^{-j} \\
&\le 2^{\mathrm{H}_0(W)-\lambda} + \sum_{j=\mu}^{\lambda-1} 2^{-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)+\log(1/\delta)+1} \\
&< 2^{2-\mu} + (\lambda-\mu)\cdot 2^{-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)+\log(1/\delta)+1} \\
&\le (\lambda-\mu+1)\cdot 2^{-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)+\log(1/\delta)+1} \\
&\le \mathrm{H}_0(W)\cdot 2^{-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)+\log(1/\delta)+1} \;.
\end{aligned}$$

The second inequality follows from the fact that $w \in L_j$ implies $\Pr[\, W = w\,] \le 2^{-j}$, and that $|L_j(|R_j|)| \le |R_j|$. The third inequality follows from the fact that $|R_\lambda| \le 2^{\mathrm{H}_0(W)}$, and $|R_j| \le 2^{j-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)+\log(1/\delta)+1}$ for $j \in [\mu, \lambda-1]$. The third inequality follows since by definition $\lambda - \mu > \mathrm{H}_0(W) - 2$, and the fourth since $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) \le \mu$ and $\log(1/\delta) \ge 1$ imply that $2^{2-\mu} \le 2^{-\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)+\log(1/\delta)+1}$. The final inequality follows since $\lambda - \mu + 1 \le \mathrm{H}_0(W)$. Taking logs of both sides, we obtain the security bound of Theorem 1.
▮

C.4 Proof of Theorem 3

Proof: We begin by proving that the sketch is δ-correct. Let $\mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) = \tilde{\mu}$. We first observe that for each $j \in [\mu, \gamma]$, the family of hash functions $\mathsf{F}_j : \mathcal{S} \times \{0,1\}^\ell \to \{0,1\}^{j-\tilde{\mu}+\log(1/\delta)+1}$ defined by the rule

$$\mathsf{F}_j(w;\mathsf{sa}) = [\mathsf{F}(w;\mathsf{sa})]_1^{j-\tilde{\mu}+\log(1/\delta)+1} \;,$$

is itself strongly universal. Indeed, the strong universality of F implies that for any $w \ne w' \in \mathcal{S}$, and $y_1, y_1' \in \{0,1\}^{j-\tilde{\mu}+\log(1/\delta)+1}$

$$\begin{aligned}
\Pr[\, \mathsf{F}_j(w;\mathsf{sa}) = y_1 \,] &= \sum_{y_2 \in \{0,1\}^{\gamma-j}} \Pr[\, \mathsf{F}(w;\mathsf{sa}) = y_1 || y_2 \,] \\
&= 2^{\gamma-j} \cdot 2^{-(\gamma-\tilde{\mu}+\log(1/\delta)+1)} \\
&= 2^{-(j-\tilde{\mu}+\log(1/\delta)+1)} \;,
\end{aligned}$$

and

$$\begin{aligned}
&\Pr[\, \mathsf{F}_j(w;\mathsf{sa}) = y_1 \wedge \mathsf{F}_j(w';\mathsf{sa}) = y_1' \,] \\
&= \sum_{y_2, y_2' \in \{0,1\}^{\gamma-j}} \Pr[\, \mathsf{F}(w;\mathsf{sa}) = y_1||y_2 \wedge \mathsf{F}(w';\mathsf{sa}) = y_1'||y_2' \,] \\
&= 2^{2\cdot(\gamma-j)} \cdot 2^{-2\cdot(\gamma-\tilde{\mu}+\log(1/\delta)+1)} \\
&= 2^{-2\cdot(j-\tilde{\mu}+\log(1/\delta)+1)} \;.
\end{aligned}$$

All probabilities are over $\mathsf{sa} \xleftarrow{\$} \{0,1\}^\ell$. In both cases the second equality follows from the strong universality of F, and the fact that $|\{0,1\}^{\gamma-j}| = 2^{(\gamma-j)}$. Together these properties show that F_j is strongly universal as required. With this in place, fix $w \in \mathcal{M}$, $\tilde{w} \in \mathcal{S}$ such that $w \in B_t(\tilde{w})$. Letting

$\tilde{\mu} = \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W)$ it follows that

$$\begin{aligned}
&\Pr\left[\,\mathsf{LHH\text{-}Rec}(\tilde{w}, \mathsf{LHH\text{-}SS}(w)) \neq w\,\right] && (1)\\
&= \Pr\left[\exists w' \in B_t(\tilde{w}) \ :\ \mathsf{F}_{\mathsf{L}(w')}(w'; \mathsf{sa}) = [y]_1^{(\mathsf{L}(w') - \tilde{\mu} + \log(\frac{1}{\delta}) + 1)}\right] && (2)\\
&\leq \sum_{w' \in B_t(\tilde{w})} \Pr\left[\mathsf{F}_{\mathsf{L}(w')}(w'; \mathsf{sa}) = [y]_1^{(\mathsf{L}(w') - \tilde{\mu} + \log(\frac{1}{\delta}) + 1)}\right] && (3)\\
&\leq \sum_{w' \in B_t(\tilde{w})} 2^{-(\mathsf{L}(w') - \tilde{\mu} + \log(\frac{1}{\delta}) + 1)} && (4)\\
&= \delta \sum_{w' \in B_t(\tilde{w})} \frac{2^{-(\mathsf{L}(w')+1)}}{2^{-\tilde{\mu}}} && (5)\\
&< \delta \sum_{w' \in B_t(\tilde{w})} \frac{\Pr\left[\,W = w'\,\right]}{2^{-\tilde{\mu}}} && (6)\\
&\leq \delta\ . && (7)
\end{aligned}$$

All probabilities are over the coins of $\mathsf{LHH\text{-}SS}$. The inequality in line (3) follows from the application of a union bound. Line (4) follows from the definition of a strongly universal hash function. The inequality in line (6) follows from the fact that $w \in L_j$ implies that $2^{-(j+1)} < \Pr\left[\,W = w\,\right] \leq 2^{-j}$. Finally, line (7) follows from the fact that for any $\tilde{w} \in \mathcal{S}$, $\Pr\left[\,W \in B_t(\tilde{w})\,\right] \leq 2^{-\tilde{\mu}}$. Therefore,

$$\Pr\left[\,\mathsf{LHH\text{-}Rec}(\tilde{w}, \mathsf{SS}(w)) = w\,\right] \geq 1 - \delta\ ,$$

as required.

We now consider security. For each $y \in \{0,1\}^{\gamma - \tilde{\mu} + \log(1/\delta) + 1}$, let $\mathcal{X}_{(y,\mathsf{sa})} = \{w \in W \ :\ \mathsf{F}_{\mathsf{L}(w)}(w; \mathsf{sa}) = [y]_1^{(\mathsf{L}(w) - \tilde{\mu} + \log(1/\delta) + 1)}\}$. We wish to upper bound

$$\begin{aligned}
&2^{-\tilde{\mathrm{H}}^q_\infty(W \mid \mathsf{LHH\text{-}SS}(W))}\\
&= \sum_{\mathsf{sa}} \sum_{y} \max_{w_1,\ldots,w_q} \sum_{i=1}^{q} \Pr\left[\,W = w_i \wedge \mathsf{LHH\text{-}SS}(W) = (\mathsf{sa}, y) \wedge \mathsf{sa} \text{ is chosen }\,\right]\\
&= \sum_{\mathsf{sa}} \sum_{y} \max_{w_1,\ldots,w_q} \sum_{i=1}^{q} \Pr\left[\,W = w_i\,\right] \cdot \Pr\left[\,\mathsf{LHH\text{-}SS}(w; \mathsf{sa}) = y\,\right] \cdot 2^{-\ell}\\
&= \sum_{\mathsf{sa}} \sum_{y} \max_{\substack{w_1,\ldots,w_q\ :\\ w_i \in \mathcal{X}_{(y,\mathsf{sa})}}} \sum_{i=1}^{q} \Pr\left[\,W = w_i\,\right] \cdot 2^{-(\gamma - \mathsf{L}(w))} \cdot 2^{-\ell}\ ,
\end{aligned}$$

where the second equality follows from Bayes' rule, and the third since $\Pr\left[\,\mathsf{LHH\text{-}SS}(w; \mathsf{sa}) = y\,\right] = 2^{-(\gamma - \mathsf{L}(w))}$ if $\mathsf{F}_{\mathsf{L}(w)}(w; \mathsf{sa}) = [y]_1^{(\mathsf{L}(w) - \tilde{\mu} + \log(1/\delta) + 1)}$ and 0 otherwise.

Fix $\mathsf{sa} \in \{0,1\}^\ell$, and notice that each $w \in \mathcal{M}$ lies in precisely $2^{(\gamma - \mathsf{L}(w))}$ preimage sets. As such as we sum over the $y \in \{0,1\}^{(\gamma - \tilde{\mu} + \log(1/\delta) + 1)}$, picking at each step the q points in $\mathcal{X}_{(y,\mathsf{sa})}$ which maximise the RHS of the term, it follows that each $w \in \mathcal{M}$ may be selected for at most $2^{(\gamma - \mathsf{L}(w))}$ of the y, and each time contributes weight $\Pr\left[\,W = w\,\right] \cdot 2^{-(\gamma - \mathsf{L}(w))}$ to the total. As such the above total will be maximised when each preimage set $\mathcal{X}_{(y,\mathsf{sa})}$ contains precisely q of the heaviest $q \cdot 2^{\gamma - \tilde{\mu} + \log(1/\delta) + 1}$ points in W', where W' is the distribution constructed by taking each point $w \in \mathcal{M}$ and replacing it with $2^{(\gamma - j)}$ points, each of weight $\Pr\left[\,W = w\,\right] \cdot 2^{-(\gamma - j)}$, where $w \in L_j$. Therefore for each fixed sa we may bound the inner sum,

$$\sum_{y} \max_{\substack{w_1,\ldots,w_q\ :\\ w_i \in \mathcal{X}_{(y,\mathsf{sa})}}} \sum_{i=1}^{q} \Pr\left[\,W = w\,\right] \cdot 2^{-(\gamma - \mathsf{L}(w))} \leq 2^{-\mathrm{H}_\infty^{q \cdot 2^{\gamma - \tilde{\mu} + \log(1/\delta) + 1}}(W')}\ .$$

Averaging over the salt and taking logs of both sides implies to result. In order to form a straightforward comparison with the security of $\mathsf{FRS2}$, we upper bound this term via the following claim:

Claim. For any error setting $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$, it holds that,

$$2^{-\mathrm{H}^q_\infty(W, \mathsf{LHH\text{-}SS}(W))} \leq q \cdot 2^{-(\gamma+\ell)} \ .$$

Proof:

$$2^{-\mathrm{H}^q_\infty(W, \mathsf{LHH\text{-}SS}(W))} = \max_{\substack{w_1,\dots,w_q \\ (y_1,\mathsf{sa}_1),\dots,(y_q,\mathsf{sa}_q)}} \sum_{i=1}^{q} \Pr\left[\, W = w_i \wedge \mathsf{LHH\text{-}SS}(W) = (y_i, \mathsf{sa}_i)\,\right]$$

$$= \max_{\substack{w_1,\dots,w_q \\ (y_1,\mathsf{sa}_1),\dots,(y_q,\mathsf{sa}_q)}} \sum_{i=1}^{q} \Pr\left[\, \mathsf{LHH\text{-}SS}(w_i; \mathsf{sa}_i) = y_i \,\right] \Pr\left[\, W = w_i \,\right] \Pr\left[\, \mathsf{sa}_i \text{ is chosen } \right]$$

$$\leq \max_{\substack{w_1,\dots,w_q \\ (y_1,\mathsf{sa}_1),\dots,(y_q,\mathsf{sa}_q)}} \sum_{i=1}^{q} 2^{-\gamma} \cdot 2^{-\ell}$$

$$\leq q \cdot 2^{-(\gamma+\ell)} \ .$$

The first equality follows from the definition of the min-entropy of a joint distribution. The second equality is implied by Bayes' rule. The third line follows since for any $w \in \mathcal{M}$, it holds that $\Pr\left[\, W = w \,\right] \cdot \Pr\left[\, \mathsf{LHH\text{-}SS}(w_i; \mathsf{sa}_i) = y_i \,\right] \leq 2^{-\mathsf{L}(w)} \cdot 2^{-(\gamma - \mathsf{L}(w))} = 2^{-\gamma}$; summing over q proves the claim.

Finally, the chain rule (Lemma 1) implies that,

$$\begin{aligned} \tilde{\mathrm{H}}^q_\infty(W|\mathsf{SS}(W)) &\geq \mathrm{H}^q_\infty(W, \mathsf{SS}(W)) - \log|\mathrm{supp}(\mathsf{SS}(W))| \\ &= \gamma + \ell - \log(q) - \left(\gamma - \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) + \log(1/\delta) + 1 + \ell\right) \\ &= \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log(q) - \log(1/\delta) - 1 \ . \end{aligned}$$

In particular, in the case where $q = 1$, this gives,

$$\tilde{\mu}_1 \geq \mathrm{H}^{\mathrm{fuzz}}_{t,\infty}(W) - \log(1/\delta) - 1 \ .$$

▮

D Proofs from Section 6

D.1 PBAS-BF versus FRS1 for Flat Distributions

Theorem 6. *Let $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$ be an error setting where W is flat over $\mathcal{S}$, and let $\beta_{\max}$ be the maximal ball size. Let* PBAS-BF *be the brute-force scheme for this error setting using a random oracle $\mathbf{H}$ with associated hashing cost c_{bf}. Let* $\mathsf{FRS1} = (\mathsf{FRS1\text{-}SS}, \mathsf{FRS1\text{-}Rec})$ *be the FRS secure sketch for flat distributions from Lemma 3, achieving $1 - \delta$ error correction for some $0 < \delta < 1$. We model the universal hash function used by the sketch as a random oracle. Let* PBAS-SS[FRS1] *be the sketch-assisted* PBAS *scheme built from* FRS1*, and using random oracle $\mathbf{H}'$ with associated cost c_{ss} set such that* $\mathsf{RT}(\mathrm{Chk\text{-}SS}, \mathsf{c}_{\mathrm{ss}}) = \mathsf{RT}(\mathrm{Chk\text{-}BF}, \mathsf{c}_{\mathrm{bf}})$. *Then for any adversary $\mathcal{A}$ against* PBAS-BF *running in time at most T, there exists an adversary $\mathcal{B}$ against* PBAS-SS[FRS1] *such that*

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF},\mathsf{E}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS1]},\mathsf{E}}(\mathcal{B}) + \left(\frac{\delta \cdot e}{2}\right)^{q_{\mathrm{ss}}}$$

and, moreover, $\mathcal{B}$ runs in time T and so can make at most $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$ queries.

Proof: To ensure $\mathsf{RT}(\mathrm{Chk\text{-}SS}, \mathsf{c}_{\mathrm{ss}}) = \mathsf{RT}(\mathrm{Chk\text{-}BF}, \mathsf{c}_{\mathrm{bf}})$, set $\mathsf{c}_{\mathrm{ss}} = \mathsf{c}_{\mathrm{bf}} \cdot \frac{\beta_{\max}}{2}$. Now let $\mathcal{A}$ and $\mathcal{B}$ be adversaries in games $\mathrm{MR}^{\mathcal{A}}_{\mathsf{PBAS\text{-}BF},\mathsf{E}}$ and $\mathrm{MR}^{\mathcal{B}}_{\mathsf{PBAS\text{-}SS[FRS1]},\mathsf{E}}$ respectively, both running in time T. Let q_{bf}, (resp. q_{ss}) denote the maximum number of distinct oracle queries $\mathcal{A}$ (resp. $\mathcal{B}$) may make in the given time. Then $q_{\mathrm{bf}} = T/\mathsf{c}_{\mathrm{bf}} = q_{\mathrm{ss}} \cdot \frac{\beta_{\max}}{2}$. Since there are at most $|\mathcal{M}|$ distinct points in the support of W, it follows that $q_{\mathrm{bf}} \leq |\mathcal{M}|$.

It is straightforward to see that since W is flat with $\mathrm{H}_\infty(W) = \mu$, it must be the case that,

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{A}) \leq q_{\mathrm{bf}} \cdot 2^{-\mu} = q_{\mathrm{ss}} \cdot \beta_{\max} \cdot 2^{-(\mu+1)}.$$

Now let $\mathcal{B}$ be the adversary in game $\mathrm{MR}^{\mathcal{B}}_{\mathsf{PBAS\text{-}SS[FRS1],E}}$ who on input sketch $s = (y, \mathsf{h})$ computes the preimage set of s, $\mathcal{X}_s = \{w \in \mathcal{M} : \mathsf{h}(w) = y\}$ and queries q_{ss} points from $\mathcal{X}_s$ to $\mathbf{H}$ where $\mathsf{h} \xleftarrow{\$} \mathcal{H}$. Then,

$$\begin{aligned}
&\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS1],E}}(\mathcal{B}) \\
&= \sum_{\mathsf{h}\in\mathcal{H}} \sum_{y} \max_{w_1,\ldots,w_{q_{\mathrm{ss}}}} \sum_{i=1}^{q_{\mathrm{ss}}} \Pr\left[W = w_i \wedge \mathsf{FRS1\text{-}SS}^{\mathsf{h}}(W) = y \wedge \mathsf{h} \text{ chosen} \right] \\
&= \sum_{\mathsf{h}\in\mathcal{H}} \sum_{y} \max_{w_1,\ldots,w_{q_{\mathrm{ss}}}} \sum_{i=1}^{q_{\mathrm{ss}}} \Pr\left[\mathsf{h}(w_i) = y \right] \cdot \Pr\left[W = w_i \right] \cdot \frac{1}{|\mathcal{H}|} \\
&= \sum_{\mathsf{h}\in\mathcal{H}} \sum_{y} \max_{\substack{w_1,\ldots,w_{q_{\mathrm{ss}}}: \\ w_i \in \mathcal{X}_s}} \sum_{i=1}^{q_{\mathrm{ss}}} 2^{-\mu} \cdot \frac{1}{|\mathcal{H}|}
\end{aligned}$$

The second to last equality follows from the fact that $\Pr\left[\mathsf{h}(w) = y \right] = 1$ if and only if $w \in \mathcal{X}_s$, and the fact that W is flat.

Notice that we can lower bound the above sum via a balls-in-bins experiment. We view each point in the support of W as a ball of equal weight, and each value in the range of the sketch as a bin. Formally, the experiment takes a set of $|\mathcal{M}|$ ordered balls $b_1, \ldots, b_{|\mathcal{M}|}$ and a set of $\beta_{\max}/\delta$ bins, $B_1, \ldots, B_{\beta_{\max}/\delta}$. Each ball b_k is assigned weight $2^{-\mu}$. The choice of the random oracle h fixes a "throwing" of the balls into the bins, where $\Pr\left[b_k \text{ falls in } B_j \right] = \frac{\delta}{\beta_{\max}}$ over the coins of the random oracle. $\mathcal{B}$ is allowed to choose at most q_{ss} ball from each of the bins, (since it is of course possible that a bin may receive less than q_{ss} balls), and the success probability for that particular h is equal to the sum of the weights of those balls. Finally, $\mathcal{B}$'s advantage is computed by taking the average of this success probability over the coins of the random oracle.

Notice that if $\mathcal{B}$ chooses to select balls from each bin in the order in which they were thrown (which makes no difference to the final sum, since all balls are of equal weight), then any ball b_k will be included in the success probability sum unless the chosen random oracle h places it in a bin already holding $\geq q_{\mathrm{ss}}$ balls. We show that with high probability, each of the first $q_{\mathrm{ss}} \cdot \frac{\beta_{\max}}{2}$ points thrown are among the first q_{ss} which land in their bin. Let $\mathcal{Q}$ denote the set of points which are among the first q_{ss} in their bin as dictated by the coins of h. Let X_k be an indicator variable which is set to 1 if and only if $b_k \in \mathcal{Q}$. Adding h to the subscript of these terms represents the same event conditioned on h being the chosen random oracle. With this in place, we may rewrite our original sum,

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS1],E}}(\mathcal{B}) &= \sum_{\mathsf{h}} \sum_{y} \max_{\substack{w_1,\ldots,w_{q_{\mathrm{ss}}}: \\ w_i \in \mathcal{X}_{(y,\mathsf{h})}}} \sum_{i=1}^{q_{\mathrm{ss}}} 2^{-\mu} \cdot \frac{1}{|\mathcal{H}|} \\
&= \sum_{\mathsf{h}} \left(|\mathcal{Q}_{\mathsf{h}}| \cdot 2^{-\mu} \right) \frac{1}{|\mathcal{H}|} \\
&= \sum_{\mathsf{h}} \left(\sum_{k=1}^{|\mathcal{M}|} X_{k,\mathsf{h}} \cdot 2^{-\mu} \right) \frac{1}{|\mathcal{H}|} \\
&= 2^{-\mu} \cdot \sum_{k=1}^{|\mathcal{M}|} \left(\sum_{\mathsf{h}\in\mathcal{H}} X_{k,\mathsf{h}} \frac{1}{|\mathcal{H}|} \right) \\
&= 2^{-\mu} \cdot \sum_{k=1}^{|\mathcal{M}|} \Pr\left[X_k = 1 \right].
\end{aligned}$$

It remains to bound $\Pr[X_k = 1]$. Let $E_{k,j}$ denote the event that ball b_k falls in bin B_j, and notice that

$$\begin{aligned}\Pr[X_k = 1] &= 1 - \Pr[b_k \text{ falls in a bin containing } \geq q_{\text{ss}} \text{ balls}] \\ &= 1 - \sum_{j=1}^{\frac{\beta_{\max}}{\delta}} \Pr[B_j \text{ contains } \geq q_{\text{ss}} \text{ balls} \mid E_{k,j}] \Pr[E_{k,j}] ,\end{aligned}$$

where all probabilities are over the coins of h. Notice that for $b_1, \dots, b_{q_{\text{ss}}}$, and each B_j, $\Pr[X_k = 1] = 1$, since $\Pr[B_j \text{ contains } \geq q_{\text{ss}} \text{ balls}] = 0$ for $k \in [q_{\text{ss}}]$. Furthermore, for all $k \geq [q_{\text{ss}} + 1, q_{\text{ss}} \cdot \frac{\beta_{\max}}{2}]$,

$$\begin{aligned}\Pr[X_k = 1] &= 1 - \sum_{j=1}^{\frac{\beta_{\max}}{\delta}} \Pr[B_j \text{ contains } \geq q_{\text{ss}} \text{ balls} \mid E_{k,j}] \frac{\delta}{\beta_{\max}} \\ &\geq 1 - \sum_{j=1}^{\frac{\beta_{\max}}{\delta}} \binom{(k-1)}{q_{\text{ss}}} \cdot \left(\frac{\delta}{\beta_{\max}}\right)^{q_{\text{ss}}} \frac{\delta}{\beta_{\max}} \\ &\geq 1 - \left(\frac{e \cdot (k-1)}{q_{\text{ss}}}\right)^{q_{\text{ss}}} \left(\frac{\delta}{\beta_{\max}}\right)^{q_{\text{ss}}} \\ &\geq 1 - \left(\frac{\delta \cdot e}{2}\right)^{q_{\text{ss}}} ,\end{aligned}$$

where all probabilities are over the coins of choosing the salt. The first equality follows since $\Pr[E_{k,j}] = \frac{\delta}{\beta_{\max}}$ for all k, j. The first inequality follows from taking the probability that the bin ball b_k falls in contains precisely q_{ss} of the $(k-1)$ heavier points. The second inequality follows from Stirling's approximation, and cancelling the $\frac{\beta_{\max}}{\delta}$ equal terms in the summation with the the RHS $\frac{\delta}{\beta_{\max}}$ term. The final inequality follows since $k - 1 < q_{\text{ss}} \cdot \frac{\beta_{\max}}{2}$. Putting this all together we conclude that,

$$\begin{aligned}\mathbf{Adv}^{\text{mr}}_{\mathsf{PBAS\text{-}SS[FRS1],E}}(\mathcal{B}) &= \sum_{k=1}^{|\mathcal{M}|} \Pr[W = w_i] \cdot \Pr[X_k = 1] \\ &\geq \sum_{k=1}^{q_{\text{ss}} \cdot \frac{\beta_{\max}}{2}} \Pr[W = w_i] \cdot \left(1 - \left(\frac{\delta \cdot e}{2}\right)^{q_{\text{ss}}}\right) \\ &= 2^{-\mu} \sum_{k=1}^{q_{\text{ss}} \cdot \frac{\beta_{\max}}{2}} \cdot \left(1 - \left(\frac{\delta \cdot e}{2}\right)^{q_{\text{ss}}}\right) \\ &= 2^{-\mu} \cdot q_{\text{ss}} \cdot \frac{\beta_{\max}}{2} \cdot \left(1 - \left(\frac{\delta \cdot e}{2}\right)^{q_{\text{ss}}}\right) \\ &\geq 2^{-\mu} \cdot q_{\text{ss}} \cdot \frac{\beta_{\max}}{2} - \left(\frac{\delta \cdot e}{2}\right)^{q_{\text{ss}}} \\ &\geq \mathbf{Adv}^{\text{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{A}) - \left(\frac{\delta \cdot e}{2}\right)^{q_{\text{ss}}} . \qquad \blacksquare\end{aligned}$$

The first inequality follows since $|supp(W)| = |\mathcal{M}|$, so $\sum_{j=1}^{|\mathcal{M}|} \Pr[W = w_i] \geq \sum_{j=1}^{q_{\text{ss}} \cdot \frac{\beta_{\max}}{2}} \Pr[W = w_i]$; also for $k \in [q_{\text{ss}} \cdot \frac{\beta_{\max}}{2}]$ it holds that $\Pr[X_k = 1] \geq 1 - \left(\frac{\delta \cdot e}{2}\right)^{q_{\text{ss}}}$. The next equality follows since $\Pr[W = w_i] = 2^{-\mu}$ for all $w_i \in \mathcal{M}$. The second to last inequality follows since $2^{-\mu} \cdot q_{\text{ss}} \cdot \frac{\beta_{\max}}{2} \leq 1$. The final inequality follows since $\mathbf{Adv}^{\text{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{A}) = 2^{-\mu} \cdot q_{\text{ss}} \cdot \frac{\beta_{\max}}{2}$.

D.2 Proof of Lemma 5

Proof: Let $\mathcal{A}$ be an adversary in game $\text{MR}^{\mathcal{A}}_{\mathsf{PBAS\text{-}PPH,E}}$ running in time T. We shall upper bound $\mathcal{A}$'s success probability. It is straightforward to see that $\mathcal{A}$'s optimal strategy is to query the set of points $\mathcal{Q}^*$ to the relevant oracles, which maximizes

$$\sum_{w' \in \mathcal{Q}^*} \Pr[W = w],$$

subject to the condition that

$$\sum_{w' \in \mathcal{Q}^*} \mathsf{c}_{\text{PPH}}^{\mathsf{L}(w')} \leq T.$$

Since $\mathsf{c}_{\text{PPH}}^{j} = \mathsf{c}_{\text{PPH}} \cdot 2^{\tilde{\mu}-(j+1)}$ we may rewrite the above expression

$$\sum_{j=\mu}^{\gamma} |\mathcal{Q}^* \cap L_j| \cdot \mathsf{c}_{\text{PPH}} \cdot 2^{\tilde{\mu}-(j+1)} \leq T$$

Dividing both sides by $\frac{1}{2} \cdot \mathsf{c}_{\text{PPH}} \cdot 2^{\tilde{\mu}}$ (and noting that $\mathsf{c}_{\text{ss}} = \frac{1}{2} \cdot \mathsf{c}_{\text{PPH}}$, and $q_{\text{ss}} = T/\mathsf{c}_{\text{ss}}$) yields

$$\sum_{j=\mu}^{\gamma} |\mathcal{Q}^* \cap L_j| \cdot 2^{-j} \leq q_{\text{ss}} \cdot 2^{-\tilde{\mu}}$$

As $w \in L_j$ implies $\Pr[W = w] \leq 2^{-j}$, we conclude that

$$\mathbf{Adv}^{\text{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) = \Pr[W \in \mathcal{Q}^*] \leq \sum_{j=\mu}^{\gamma} |\mathcal{Q}^* \cap L_j| \cdot 2^{-j} \leq q_{\text{ss}} \cdot 2^{-\tilde{\mu}} \ .$$

∎

Notice that since $\mathcal{Q}^*$ is the set of points which maximises the above sum, notice that each set $\mathcal{Q}^* \cap L_j$ must either be empty, or contain precisely the $|\mathcal{Q}^* \cap L_j|$ most probable points in L_j. Were this not the case, one could achieve a higher success probability for the same computational cost by swapping any lower point in $\mathcal{Q}^* \cap L_j$ for one of the unused more probable points (which, being from the same layer, will incur the same computational cost to test as the point it replaced), violating the claim that $\mathcal{Q}^*$ maximises $\Pr[W \in \mathcal{Q}^*]$. Since testing a point from level L_j takes time $\mathsf{c}_{\text{PPH}}^{j} = \mathsf{c}_{\text{ss}} \cdot 2^{\tilde{\mu}-j}$, the constraint that $\mathcal{A}$ runs in time T implies that for each $j \in [\mu, \gamma]$, it must be the case that $|\mathcal{Q}^* \cap L_j| \leq T/\mathsf{c}_{\text{ss}} \cdot 2^{(j-\tilde{\mu})} = q_{\text{ss}} \cdot 2^{(j-\tilde{\mu})}$. We shall utilise this observation in subsequent proofs.

D.3 Proof of Theorem 4 in the case that $\Pi = \mathsf{FRS2}$

Proof: To ensure equal run times, we set parameters such that $\mathsf{c}_{\text{bf}} = \mathsf{c}_{\text{ss}} \cdot 2$. Now let $\mathcal{A}$ and $\mathcal{B}$ be adversaries in games $\text{MR}^{\mathcal{A}}_{\mathsf{PBAS\text{-}PPH,E}}$ and $\text{MR}^{\mathcal{B}}_{\mathsf{PBAS\text{-}SS[FRS2],E}}$ respectively, both running in time T. Recall that we model the universal hash function family for each layer j as a family of random oracles $\{\mathcal{H}_j\}$. Let $\mathcal{Q}^*$ denote the set of points which are queried to $\mathbf{H}$ by $\mathcal{A}$.

Let q_{ss} denote the maximum number of oracle queries $\mathcal{B}$ may make in the given time, so $q_{\text{ss}} = T/\mathsf{c}_{\text{ss}}$. On input sketch $s = (y, \mathsf{h}_j, j)$, adversary $\mathcal{B}$ proceeds as follows: if $j = \lambda$, simply return y. Otherwise compute the preimage set of (y, h_j, j) under h_j, defined $\mathcal{X}_s = \{w \in \mathcal{M} \ : \ \mathsf{h}_j(w) = y\}$, and query the q_{ss} most probable points in $\mathcal{X}_s$ to $\mathbf{H}$.

Let $R_j = \{0,1\}^{j-\tilde{\mu}+\log(1/\delta)+1}$ for $j \in [\mu, \lambda-1]$; that is to say, the range of the hash functions for layer j. Then,

$$\begin{aligned}
&\mathbf{Adv}^{\text{mr}}_{\mathsf{PBAS\text{-}SS[FRS2],E}}(\mathcal{B}) - \Pr[W \in L_\lambda] \\
&= \sum_{j=\mu}^{\lambda-1} \sum_{\mathsf{h}_j \in \mathcal{H}_j} \sum_{y \in R_j} \max_{w_1,\ldots,w_{q_{\text{ss}}}} \sum_{i=1}^{q_{\text{ss}}} \Pr\left[W = w_i \wedge \mathsf{FRS2\text{-}SS}^{\mathsf{h}_j}(W) = y\right] \cdot \frac{1}{|\mathcal{H}_j|} \\
&= \sum_{j=\mu}^{\lambda-1} \sum_{\mathsf{h}_j \in \mathcal{H}_j} \sum_{y \in R_j} \max_{w_1,\ldots,w_{q_{\text{ss}}}} \sum_{i=1}^{q_{\text{ss}}} \Pr\left[\mathsf{FRS2\text{-}SS}^{\mathsf{h}_j}(w_i) = y\right] \cdot \Pr[W = w_i] \cdot \frac{1}{|\mathcal{H}_j|} \\
&= \sum_{j=\mu}^{\lambda-1} \sum_{\mathsf{h}_j \in \mathcal{H}_j} \sum_{y \in R_j} \max_{\substack{w_1,\ldots,w_{q_{\text{ss}}}: \\ w_i \in \mathcal{X}_s}} \sum_{i=1}^{q_{\text{ss}}} \Pr[W = w_i] \cdot \frac{1}{|\mathcal{H}_j|} \ .
\end{aligned}$$

The right hand side of the first equation follows from the fact that $\mathsf{FRS2\text{-}SS}$ reveals the points in layer L_λ as part of the sketch, so $\Pr\left[\, W = w \wedge \mathsf{FRS2\text{-}SS}^{\mathsf{h}_j}(W) = y \,\right] = \Pr\left[\, W = w \,\right]$ for all $w \in L_\lambda$. The second equality follows from Bayes' rule, and the final line follows since $\Pr\left[\, \mathsf{FRS2\text{-}SS}^{\mathsf{h}_j}(w_i) = y \,\right] = 1$ if $w_i \in \mathcal{X}_{(y,\mathsf{h}_j,j)}$, and 0 otherwise.

We lower bound the above sum via a balls-in-bins experiment. For each $j \in [\mu, \lambda-1]$, the choice of random oracle h_j "throws" the $|L_j|$ balls in $2^{j+1-\tilde{\mu}}/\delta$ bins; we then calculate the expected total weight of the heaviest q_{ss} balls in each bin. The sum of these over the layers plus the weight of the lowest layer amount to $\mathcal{B}$'s expected success probability. Let $b_{k,j}$ denote the ball representing the k^{th} heaviest point in layer j, and let $p_{k,j}$ denote the weight of that ball.

Since within a given layer each ball is heavier than its successors, it follows that $b_{k,j}$ will be included in the success probability sum unless it falls in a bin already holding q_{ss} or more (heavier) balls. Consider the set of points $\mathcal{Q}_j$ that are included from level j, and let $X_{k,j}$ be an indicator variable which is equal to 1 if and only if ball $b_{k,j}$ is among the q_{ss} heaviest in its bin. Adding h to the subscript of these terms represents the same event conditioned on h being the chosen random oracle. Then we can rewrite the above sum,

$$
\begin{aligned}
&\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS2],E}}(\mathcal{B}) - \Pr\left[\, W \in L_\lambda \,\right] \\
&= \sum_{j=\mu}^{\lambda-1} \left(\sum_{\mathsf{h}_j \in \mathcal{H}_j} \left(\Pr\left[\, W \in \mathcal{Q}_{j,\mathsf{h}_j} \,\right] \right) \cdot \frac{1}{|\mathcal{H}|} \right) \\
&= \sum_{j=\mu}^{\lambda-1} \left(\sum_{\mathsf{h}_j \in \mathcal{H}_j} \left(\sum_{k=1}^{|L_j|} p_{k,j} \cdot X_{k,j,\mathsf{h}_j} \right) \cdot \frac{1}{|\mathcal{H}|} \right) \\
&= \sum_{j=\mu}^{\lambda-1} \sum_{k=1}^{|L_j|} p_{k,j} \cdot \left(\sum_{\mathsf{h}_j \in \mathcal{H}_j} X_{k,j,\mathsf{h}_j} \cdot \frac{1}{|\mathcal{H}|} \right) \\
&= \sum_{j=\mu}^{\lambda-1} \sum_{k=1}^{|L_j|} p_{k,j} \cdot \Pr\left[\, X_{k,j} = 1 \,\right] ,
\end{aligned}
$$

where the probability in the final line is over the coins of the random oracle.

We show that with overwhelming probability this sum exceeds $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A})$, by showing that with high probability each of the points in $\mathcal{Q}^*$ — the set of points output by the optimal attacker $\mathcal{A}$ in game $\mathrm{MR}^{\mathcal{A}}_{\mathsf{PBAS\text{-}PPH,E}}$ — is included in the computation of the sum. Notice that any points $w \in \mathcal{Q}^*$ for which $\Pr\left[\, W = w \,\right] \le 2^{-\lambda}$ will be guessed by $\mathcal{B}$ with probability 1 and so are already counted in the above total.

We now consider the lower layers. Let $w_{k,j}$ denote the k^{th} heaviest point in layer j, with weight $p_{k,j}$. Fix an index $j \in [\mu, \lambda - 1]$. By the same argument as before, for all $w_k \in \mathcal{Q}^*$ such that $k \in [q_{\mathrm{ss}}]$, $\Pr\left[\, X_{k,j} = 1 \,\right] = 1$ (since these points must land in a bin containing less than q_{ss} balls), and for all $w_{k.j} \in \mathcal{Q}^*$ such that $k > q_{\mathrm{ss}}$, it holds that,

$$
\begin{aligned}
\Pr\left[\, X_{k,j} = 1 \,\right] &\ge 1 - \binom{k-1}{q_{\mathrm{ss}}} \left(\frac{\delta}{2^{j+1-\tilde{\mu}}} \right)^{q_{\mathrm{ss}}} \\
&\ge 1 - \left(\frac{e \cdot (k-1)}{q_{\mathrm{ss}}} \right)^{q_{\mathrm{ss}}} \left(\frac{\delta}{2^{j+1-\tilde{\mu}}} \right)^{q_{\mathrm{ss}}} \\
&\ge 1 - \left(\frac{\delta \cdot e}{2} \right)^{q_{\mathrm{ss}}} ,
\end{aligned}
$$

where all probabilities are over the coins of the random oracle. The first inequality follows from a union bound. The second inequality is implied by Stirling's approximation. The final inequality follows from the discussion in Appendix D.2 showing that the restraint on attack run time implies that for any layer j, $|\mathcal{Q}^* \cap L_j| q_{\mathrm{ss}} \cdot 2^{j-\tilde{\mu}}$. For the optimal attacker, the points in $\mathcal{Q}^* \cap L_j$ must be the $|\mathcal{Q}^* \cap L_j|$ heaviest in their layers; thus for $\mathcal{A}$, $X_{k,j} = 1$ only if $(k-1) \le q_{\mathrm{ss}} \cdot 2^{j-\tilde{\mu}}$.

Putting this altogether, we conclude that

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS2],E}}(\mathcal{B}) &= \Pr\left[\, W \in L_\lambda \,\right] + \sum_{j=\mu}^{\lambda-1}\sum_{k=1}^{|L_j|} p_{k,j} \cdot \Pr\left[\, X_{k,j} = 1 \,\right] \\
&\geq \Pr\left[\, W \in L_\lambda \,\right] + \sum_{j=\mu}^{\lambda-1} \sum_{\substack{k\ :\\ w_{j,k} \in \mathcal{Q}^*}} p_{k,j} \cdot \left(1 - \left(\frac{\delta \cdot e}{2}\right)^{q_{\mathrm{ss}}}\right) \\
&\geq \Pr\left[\, W \in L_\lambda \,\right] + \sum_{j=\mu}^{\lambda-1} \left(\sum_{\substack{k\ :\\ w_{j,k} \in \mathcal{Q}^*}} p_{k,j} \right) - \left(\frac{\delta \cdot e}{2}\right)^{q_{\mathrm{ss}}} \\
&\geq \mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) - \left(\frac{\delta \cdot e}{2}\right)^{q_{\mathrm{ss}}} ,
\end{aligned}$$

as required. ▌

D.4 Proof of Theorem 4 in the case that $\Pi = \mathsf{LHH}$

Proof: To ensure equal run times, we set parameters such that $\mathsf{c}_{\mathrm{bf}} = \mathsf{c}_{\mathrm{ss}} \cdot 2$. Now let $\mathcal{A}$ and $\mathcal{B}$ be adversaries in games $\mathrm{MR}^{\mathcal{A}}_{\mathsf{PBAS\text{-}PPH,E}}$ and $\mathrm{MR}^{\mathcal{B}}_{\mathsf{PBAS\text{-}SS[LHH],E}}$ respectively, both running in time T. Let q_{ss} denote the maximum number of oracle queries $\mathcal{B}$ may make in the given time, so $q_{\mathrm{ss}} = T/\mathsf{c}_{\mathrm{ss}}$. Lemma 5 immediately implies,

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) \leq q_{\mathrm{ss}} \cdot 2^{-\tilde{\mu}}$$

Adversary $\mathcal{B}$ proceeds as follows on input sketch $s = (y, \mathsf{h})$: first $\mathcal{B}$ computes the preimage set $\mathcal{X}_s$

$$\mathcal{X}_s = \left\{ w \in \mathcal{M} \ :\ [\mathsf{h}(w)]_1^{\mathsf{L}(w) - \tilde{\mu} + \log(\frac{1}{\delta}) + 1} = [y]_1^{\mathsf{L}(w) - \tilde{\mu} + \log(1/\delta) + 1} \right\} .$$

For each $w \in \mathcal{X}_s$, $\mathcal{B}$ computes $2^{-(\gamma - \mathsf{L}(w))} \cdot \Pr\left[\, W = w \,\right]$, and submits the q_{ss} points to $\mathbf{H}$, for which this value is greatest. Letting $R = \{0,1\}^{\gamma - \tilde{\mu} + \log(\frac{1}{\delta}) + 1}$, it follows that

$$\begin{aligned}
&\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[LHH],E}}(\mathcal{B}) \\
&= \sum_{\mathsf{h} \in \{\mathcal{H}\}} \sum_{y \in R} \max_{w_1, \ldots, w_{q_{\mathrm{ss}}}} \sum_{i=1}^{q_{\mathrm{ss}}} \Pr\left[\, \mathsf{LHH\text{-}SS}^{\mathsf{h}}(w_i) = y \,\right] \cdot \Pr\left[\, W = w_i \,\right] \cdot \frac{1}{|\mathcal{H}|} \\
&= \sum_{\mathsf{h} \in \{\mathcal{H}\}} \sum_{y \in R} \max_{\substack{w_1, \ldots, w_{q_{\mathrm{ss}}} : \\ w \in \mathcal{X}_s}} \sum_{i=1}^{q_{\mathrm{ss}}} 2^{-(\gamma - \mathsf{L}(w))} \cdot \Pr\left[\, W = w \,\right] \cdot \frac{1}{|\mathcal{H}|},
\end{aligned}$$

where the first equality follows from Bayes' rule, and the second equality follows from the fact that $\Pr\left[\, \mathsf{LHH\text{-}SS}^{\mathsf{h}}(w) = y \,\right] = 2^{-(\gamma - \mathsf{L}(w))}$ if $w \in \mathcal{X}_s$, and 0 otherwise.

We lower bound the above sum via a balls-in-bins experiment. For each $j \in [\mu, \gamma]$, the coins of the random oracle h "throw" $|L_j|$ balls into $2^{\gamma - \tilde{\mu} + 1}/\delta$ bins corresponding to each $y \in \{0,1\}^{\gamma - \tilde{\mu} + \log(\frac{1}{\delta}) + 1}$. If a ball lands in a bin corresponding to some prefix $y_1 \in \{0,1\}^{j - \mu + \log(1/\delta) + 1}$, it is then placed in *all* of the $2^{(\gamma - j)}$ preimage sets $\mathcal{X}_s$ which share that prefix, and is assigned weight $2^{-(\gamma - j)} \cdot \Pr\left[\, W = w \,\right]$ in each (recall $\Pr\left[\, \mathsf{LHH\text{-}SS}^{\mathsf{h}}(w) = y \,\right] = 2^{-(\gamma - \mathsf{L}(w))}$ if $w \in \mathcal{X}_s$ and 0 otherwise). We then calculate the expected total weight of the heaviest q_{ss} balls in each preimage set. The sum of these over $y \in \{0,1\}^{\gamma - \tilde{\mu} + \log(1/\delta) + 1}$ is equal to $\mathcal{B}$'s success probability. Let b_k denote the ball representing the k^{th} heaviest point when ordered in terms of $2^{-(\gamma - \mathsf{L}(w))} \cdot \Pr\left[\, W = w \,\right]$. For brevity, we let $\bar{p}_k$ denote this probability for the k^{th} point in this ordering, and note that $\bar{p}_k > 2^{-(\gamma + 1)}$ for all k.

To simplify subsequent analysis, we shall further lower bound this sum by considering a slightly different experiment. For each $j \in [\mu, \gamma]$ we throw balls as before. However now when a ball falls in some bin corresponding to a hash y, we *only* place that ball in the unique preimage set corresponding to that y (as opposed to *all* of the $2^{\gamma - \mathsf{L}(w)}$ preimage sets sharing the prefix

$[y]^{\mathsf{L}(w)-\tilde{\mu}+\log(\frac{1}{\delta})+1}$ as before). However we still assign the ball weight $2^{-(\gamma-j)} \cdot \Pr[W = w]$ as before. We then calculate the expected total weight of the heaviest q_{ss} balls in each preimage set in the modified experiment. This clearly represents a lower bound on the same value in the previous experiment, since in the latter the set of balls from which we may choose from in each preimage set is a subset of those in the previous experiment.

Consider the balls thrown in order of k. Since each ball is assigned a heavier weight in the preimage sets than its successors, it follows that b_k will be included in the success probability sum unless it is placed it in a preimage set already holding q_{ss} or more (heavier) balls. For each k, let X_k be an indicator variable which is equal to 1 if and only if ball b_k is among the q_{ss} heaviest in its preimage set. Let $\bar{\mathcal{Q}}$ denote the set of points which are among the q_{ss} heaviest in their preimage sets. Adding h to the subscript of the terms represents the same event conditioned on h being the chosen random oracle. Then we can rewrite the above sum,

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[LHH]},\mathsf{E}}(\mathcal{B}) &= \sum_{\mathsf{h}\in\mathcal{H}} \Bigg(\sum_{k\,:\,w_k\in\bar{\mathcal{Q}}} \bar{p}_k \Bigg) \cdot \frac{1}{|\mathcal{H}|} \\
&\geq \sum_{\mathsf{h}\in\mathcal{H}} \Bigg(\Bigg(\sum_{k=1}^{|\mathcal{M}|} \bar{p}_k \cdot X_{k,\mathsf{h}} \Bigg) \cdot \frac{1}{|\mathcal{H}|} \Bigg) \\
&= \sum_{k=1}^{|\mathcal{M}|} \bar{p}_k \Bigg(\sum_{\mathsf{h}\in\mathcal{H}} \Big(X_{k,\mathsf{h}} \Big) \cdot \frac{1}{|\mathcal{H}|} \Bigg) \\
&= \sum_{k=1}^{|\mathcal{M}|} \bar{p}_k \cdot \Pr[X_k = 1] ,
\end{aligned}$$

where the third line follows from linearity of expectation, and the probability in the final line is over the coins of the random oracle. We show that with high probability, each point w_k for $k \in [q_{\mathrm{ss}} \cdot 2^{\gamma-\tilde{\mu}+1}]$ points is among the q_{ss} heaviest in their preimage set, and so included in this total. Since $\bar{p}_k > 2^{-(\gamma+1)}$ for all k, this would imply that the total weight of the points is greater than $q_{\mathrm{ss}} \cdot 2^{\gamma-\tilde{\mu}+1} \cdot 2^{-(\gamma+1)} = q_{\mathrm{ss}} \cdot 2^{-\tilde{\mu}}$ and we are done. Notice that for $k \in [q_{\mathrm{ss}}]$,

$$\Pr[X_k = 1] ,$$

and for any $k \in [q_{\mathrm{ss}} + 1, q_{\mathrm{ss}} \cdot 2^{\gamma-\tilde{\mu}+1}]$, it holds that,

$$\begin{aligned}
\Pr[X_k = 1] &\geq 1 - \binom{k-1}{q_{\mathrm{ss}}} \left(\frac{\delta}{2^{\gamma+1-\tilde{\mu}}} \right)^{q_{\mathrm{ss}}} \\
&\geq 1 - \left(\frac{e \cdot (k-1)}{q_{\mathrm{ss}}} \right)^{q_{\mathrm{ss}}} \left(\frac{\delta}{2^{\gamma+1-\tilde{\mu}}} \right)^{q_{\mathrm{ss}}} \\
&\geq 1 - \left(\frac{\delta \cdot e}{2} \right)^{q_{\mathrm{ss}}} ,
\end{aligned}$$

where the probabilities are over the coins of the random oracle. The first inequality follows from a union bound and the fact that $\Pr[\mathsf{h}(w) = y] = 2^{-(\gamma-\tilde{\mu}+\log(\frac{1}{\delta})+1)}$. The second inequality follows from Stirling's approximation. The final inequality follows since $k - 1 < q_{\mathrm{ss}} \cdot 2^{\gamma+1-\tilde{\mu}}$. Putting this altogether we conclude

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}SS[FRS2]},\mathsf{E}}(\mathcal{B}) &\geq \sum_{k=1}^{|\mathcal{M}|} \bar{p}_k \cdot \Pr[X_k = 1] \\
&\geq \sum_{k=1}^{q_{\mathrm{ss}} \cdot 2^{\gamma-\tilde{\mu}+1}} \bar{p}_k \cdot \Pr[X_k = 1] \\
&\geq \sum_{k=1}^{q_{\mathrm{ss}} \cdot 2^{\gamma-\tilde{\mu}+1}} \bar{p}_k \cdot \left(1 - \left(\frac{\delta \cdot e}{2} \right)^{q_{\mathrm{ss}}} \right)
\end{aligned}$$

$$
\begin{aligned}
&\geq \left(\sum_{k=1}^{q_{\mathrm{ss}} \cdot 2^{\gamma - \tilde{\mu} + 1}} \bar{p}_k \right) - \left(\frac{\delta \cdot e}{2} \right)^{q_{\mathrm{ss}}} \\
&> \left(\sum_{k=1}^{q_{\mathrm{ss}} \cdot 2^{\gamma - \tilde{\mu} + 1}} 2^{-(\gamma+1)} \right) - \left(\frac{\delta \cdot e}{2} \right)^{q_{\mathrm{ss}}} \\
&\geq q_{\mathrm{ss}} \cdot 2^{-\tilde{\mu}} - \left(\frac{\delta \cdot e}{2} \right)^{q_{\mathrm{ss}}} \\
&\geq \mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF},W}(\mathcal{A}) - \left(\frac{\delta \cdot e}{2} \right)^{q_{\mathrm{ss}}},
\end{aligned}
$$

and we are done.
▌

D.5 Proof of Theorem 5

Proof: Let $\mathsf{E} = (\mathcal{S}, W, \mathsf{dist}, t)$. We begin by setting $\mathsf{c}_{\mathrm{PPH}}$ such that Chk-PPH achieves the same run time as Chk-BF with associated cost parameter c_{bf}. Since the latter scheme achieves run time $\mathsf{RT} = \mathsf{c}_{\mathrm{bf}} \cdot \beta_{\max}$, Lemma 4 implies that setting $\mathsf{c}_{\mathrm{PPH}} = \mathsf{RT}$ ensures $\mathsf{PBAS\text{-}PPH}$ achieves run time RT also.

Next we show that for an optimal adversary $\mathcal{A}$ in game $\mathrm{MR}^{\mathcal{A}}_{\mathsf{PBAS\text{-}PPH,E}}$ running in time at most T, it holds that

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) \leq \frac{T}{\mathsf{RT}} \cdot 2^{-(\tilde{\mu}-1)} .$$

It is straightforward to see that $\mathcal{A}$'s optimal strategy is to query to oracle $\mathbf{H}'$ the set of points $\mathcal{Q}^*$ which maximizes

$$\sum_{w' \in \mathcal{Q}^*} \Pr[W = w'] ,$$

subject to the condition that

$$T \geq \sum_{w' \in \mathcal{Q}^*} \mathsf{RT} \cdot 2^{\tilde{\mu} - (\mathsf{L}(w')+1)} .$$

The latter condition implies that

$$T \geq \sum_{j=\mu}^{\gamma} |\mathcal{Q}^* \cap L_j| \; \mathsf{RT} \cdot 2^{\tilde{\mu} - (j+1)} ,$$

and multiplying both sides by $\frac{2^{-(\tilde{\mu}-1)}}{\mathsf{RT}}$ gives

$$\frac{T}{\mathsf{RT}} \cdot 2^{-(\tilde{\mu}-1)} \;\geq\; \sum_{j=\mu}^{\gamma} |\mathcal{Q}^* \cap L_j| \, 2^{-j} \;\geq\; \Pr[W \in \mathcal{Q}^*] ,$$

where the final inequality follows since $w \in L_j$ implies $\Pr[W = w] \leq 2^{-j}$.

Furthermore, we claim that $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) \geq \min\{(\frac{T}{\mathsf{RT}} - 1) \cdot 2^{-\tilde{\mu}}, 1\}$. Indeed, Lemma 4 implies that any set of points of weight at most $2^{-\tilde{\mu}}$ can be hashed in time RT. As such, an attacker $\mathcal{A}$ running in time T can hash points of weight up to $\frac{T}{\mathsf{RT}} \cdot 2^{-\tilde{\mu}}$. If this term is greater than one (the case if T is sufficiently large that $\mathcal{A}$ may hash every point in the distribution), then clearly $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) = 1$ and brute-force checking can offer no worse a trade-off for this T. If this is not the case, then finding a set of points which exactly equals this value boils down to solving an optimization problem which is likely to be inefficient, and for which an exact solution may not be possible for some T. However, $\mathcal{A}$ can efficiently get close to this advantage term by hashing points $w_1, \ldots, w_{k'}$ where $k' = \max\{k \; : \; 2^{-\mathrm{H}^k_\infty(W)} \leq \frac{T}{\mathsf{RT}} \cdot 2^{-\tilde{\mu}}\}$. Since any point $w \in W$ is such that $\Pr[W = w] \leq 2^{-\tilde{\mu}}$, it follows that $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) \geq \sum_{i=1}^{k'} \Pr[W = w_i] \geq \left(\frac{T}{\mathsf{RT}} - 1 \right) \cdot 2^{-\tilde{\mu}}$

(otherwise, we would have $\sum_{i=1}^{k'+1} \Pr\left[\, W = w_i \,\right] < \left(\frac{T}{\mathsf{RT}} - 1\right) \cdot 2^{-\tilde{\mu}} + 2^{-\tilde{\mu}} = \frac{T}{\mathsf{RT}} \cdot 2^{-\tilde{\mu}}$, contradicting the maximality of k'), proving the claim.

With this in place, let $\mathcal{B}$ be the optimal adversary in game $\mathrm{MR}^{\mathcal{B}}_{\mathsf{PBAS\text{-}BF,E}}$ who queries the $q_{\mathrm{bf}} = T/\mathsf{c}_{\mathrm{bf}} = \frac{T}{\mathsf{RT}} \cdot \beta_{\max}$ heaviest points in W to oracle $\mathbf{H}$. It is clear that $\mathcal{B}$ runs in time T, and that

$$\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{B}) = 2^{-\mathrm{H}^{q_{\mathrm{bf}}}_{\infty}(W)} \ .$$

Notice that while both the upper and lower bounds on $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A})$ grow linearly with T, $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{B})$ can only grow more slowly as T increases. As such, once we hit T such that the former advantage term is larger than the latter, then they will remain this way for all greater T. Therefore by comparing the upper and lower bounds it follows that if T is such that

$$\frac{T}{\mathsf{RT}} \cdot 2^{-(\tilde{\mu}-1)} \leq 2^{-\mathrm{H}^{q_{\mathrm{bf}}}_{\infty}(W)} \ ,$$

it holds that $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{B})$. For all error settings such that,

$$\left(\frac{T}{\mathsf{RT}} - 1\right) \cdot 2^{-\tilde{\mu}} \geq 2^{-\mathrm{H}^{q_{\mathrm{bf}}}_{\infty}(W)} \ ,$$

it holds that $\mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}BF,E}}(\mathcal{B}) \leq \mathbf{Adv}^{\mathrm{mr}}_{\mathsf{PBAS\text{-}PPH,E}}(\mathcal{A})$. Rearranging these terms yields those given in the theorem. ▌