The Query-Complexity of Preprocessing Attacks

Ashrujit Ghoshal(✉) and Stefano Tessaro

Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, WA, USA
{ashrujit,tessaro}@cs.washington.edu

Abstract. A large number of works prove lower bounds on space-time trade-offs in preprocessing attacks, i.e., trade-offs between the size of the advice and the time needed to break a scheme given such advice. We contend that the question of how much *time* is needed to produce this advice is equally important, and often highly non-trivial. However, this question has received significantly less attention. In this paper, we present lower bounds on the complexity of preprocessing attacks that depend on both offline and online time. As in the case of space-time trade-offs, we focus in particular on settings with ideal primitives, where both the offline and online time-complexities are approximated by the number of queries to the given primitive. We give generic results that highlight the benefits of salting to generically increase the offline costs of preprocessing attacks. The majority of our paper presents several results focusing on *salted* hash functions. In particular, we provide a fairly involved analysis of the pre-image- and collision-resistance security of the (two-block) Merkle-Damgård construction in our model.

1 Introduction

Preprocessing attacks leverage a suitably pre-computed *advice* string that only depends on some underlying primitive (e.g., a hash function, a block cipher, or an elliptic curve) to break a scheme using fewer resources than the best attack without such advice. For example, Hellman's [14] seminal work shows that, for a given permutation $\pi : [N] \to [N]$, one can compute a suitable S-bit advice that allows inverting the permutation on any point in time $T \approx N/S$.

A number of works, starting from [5,6,9,10,21], prove lower bounds that establish inherent trade-offs between the *size* of the advice and the *online* time needed to break the scheme (often referred to as "space-time trade-offs"). A question that has received less attention, however, concerns the study of trade-offs between the *offline* time needed to compute the advice and the online time. To the best of our knowledge, the only such result, due to Corrigan-Gibbs and Kogan [7], focuses on the discrete logarithm problem in the generic-group model.

Initiating a more comprehensive and grounded study of such *offline-online time trade-offs* is the main goal of this paper. As in prior works on space-time trade-offs, we focus on proving lower bounds relying on ideal primitives, thus

H. Handschuh and A. Lysyanskaya (Eds.): CRYPTO 2023, LNCS 14082, pp. 482–513, 2023.
https://doi.org/10.1007/978-3-031-38545-2_16

approximating time with query-complexity. In addition to providing a generic overview of how salting makes preprocessing attacks expensive, we revisit under a new lens the recent works [2,3,6,12] on preprocessing attacks against *salted* hash functions and the Merkle-Damgård construction.

Why does time-complexity matter? One main reason to study preprocessing attacks is to model non-uniform security. In this case, it is indeed irrelevant *how long* it takes to compute a good advice string since its mere *existence* suffices for an attack, although such an attack may well never be explicitly found. This perspective has emerged from the debate around the use of non-uniformity in security [4,16], although often the issue can be bypassed entirely via cleverer uniform reductions, as in e.g. [11]. The importance of non-uniform attacks in practice has also been a source of debate [19].

Here, we are concerned with a more practical and less formal perspective where preprocessing is used in practical, explicit attacks for one of two reasons:

1. An attack needs to run very quickly in the online stage (e.g., must succeed before a time-out occurs in an Internet protocol) but can afford to run much slower in an offline stage. For instance, Adrian et al. [1] use offline computation to break 512-bit finite-field discrete logs in less than a minute of online time, hence compromising legacy versions of the Diffie-Hellman handshake.
2. The advice is computed once and for all and is re-used to attack multiple instances. This is the scenario of *Rainbow Tables* [18], which leverage Hellman-type trade-offs to speed up attacks against unsalted password hashes.

In both cases, it is imperative that the offline time remains within a feasible range.

When is preprocessing worth it? Different preprocessing attacks exhibit very different offline-online time trade-offs. The main goal is to ensure that, thanks to preprocessing, the online complexity of an attack is better than the best preprocessing-free attack. For example, in Rainbow Tables, for a password dictionary of size N, the preprocessing takes time $T_1 \approx N$ to produce advice of size S, for which the online complexity is then $T_2 \approx N/S$. The online complexity bests the optimal online-only attacks, which is $\Omega(N)$; moreover, the preprocessing time is optimal since the *sum* of the offline and online time cannot beat the complexity of the best online only attack.

A more interesting example is finding collisions in the *salted* Merkle-Damgård (MD) construction, as studied in a line of recent works [2,3,12]. Given a (random) compression function $h : [N] \times [M] \to [N]$, the offline phase of the optimal attack for two-block collisions finds S collisions of the form $h(a_i, m_i) = h(a_i, m_i')$ for $m \neq m'$ and salts $a_1, \ldots, a_S$, for which offline time $T_1 \approx S \cdot \sqrt{N}$ is necessary. Then, the online phase, given the salt a, uses time $T_2 = N/S$ to find m such that $h(a, m) = a_i$ for some i, which yields a collision $m \| m_i$, $m \| m_i'$. This attack achieves the trade-off $T_1 \times T_2 = N^{3/2}$, and the online time beats the naïve Birthday attack whenever $T_1 = \Omega(N)$. It is not clear, however, whether the trade-off is optimal, and this is indeed one of the questions we are addressing below.

Our contributions. This paper initiates an in-depth investigation of the time-complexity of preprocessing attacks, and we focus primarily on salted constructions using hash functions, which is where the most interesting technical questions emerge. In particular:

- **Generic salting.** We propose a generic technique to analyze the common practice of salting to mitigate the effects of preprocessing attacks. We consider a model where every call to the underlying primitive is salted. Qualitatively, our result implies that in most settings, to beat the best online-only attack, one needs to invest an offline effort proportional to compromising the primitive on *every salt*.
- **Concrete bounds for random oracles.** Our generic technique can be combined with a recent work by Jaeger and Tessaro [15] to provide concrete quantitative upper bounds on the advantage of an offline-online adversary. These bounds are not always optimal, and we prove more refined bounds. We exemplify this situation by studying the pre-image-resistance and collision-resistance of a salted random oracle.
- **Merkle Damgård construction.** The technical bulk of this paper studies the salted Merkle-Damgård (MD) construction with a random ideal compression function. Here, salting achieves a more limited effect and still allows for non-trivial trade-offs between the offline and online complexity of an attack. We deliver quantitative upper bounds on the advantage of breaking pre-image-resistance and collision-resistance of the two-block salted MD for offline-online adversaries.

Salting defeats preprocessing. We start with a result that generically justifies the practice of salting cryptographic primitives to defeat preprocessing attacks. Such results were proved for space-time-complexity in [6], but we give an analogue result for offline-online query-complexity.

Concretely, we assume that we have a scheme Π^g that relies on a random function $g : [M] \to [N]$, and that the advantage in breaking Π^g, as a function of the number of queries to g, is a well understood quantity. (In particular, here we assume that the security depends only on the number of queries to g.) Now, we replace $g(\cdot)$ with a *salted* hash function $h(a, \cdot)$, where $h : [S] \times [M] \to [N]$. We aim to quantify security for an attacker $\mathcal{A}$ which, during an offline phase, is allowed to issue T_1 queries to $h(\cdot, \cdot)$. Then, after learning the random salt $a \leftarrow\!\!\$\ [S]$, $\mathcal{A}$ attacks $\Pi^{h(a,\cdot)}$. In this online stage, $\mathcal{A}$ can issue T_2 queries to $h(\cdot, \cdot)$.

We show that if (roughly) T^* queries to g are needed to break Π^g in the *worst case* with very high probability, then for the attacker $\mathcal{A}$ to succeed with high probability as well, $T_1 \geqslant S \cdot T^*/2$ or $T_2 \geqslant T^*/2$ must hold. In other words, the *only* way to beat the best online-only attack is to invest an amount of preprocessing equivalent to that of breaking the scheme *for every choice of the salt*. At the core of this proof is a simple argument that shows how to build an adversary $\mathcal{B}$ against Π^g, achieving the same advantage as $\mathcal{A}$, with *expected* query-complexity $T_1/S + T_2$.

Quantitative bounds for salted random oracles. The above generic result holds for adversaries achieving high advantage. Overall, we would like to go one step further to obtain quantitative precise upper bounds on the advantage of $\mathcal{A}$ as a function of T_1 and T_2 and to characterize the whole advantage curve. As our first result, we combine the above reduction to an adversary with expected query-complexity with the work by Jaeger and Tessaro [15]. This allows us to show that any adversary attempting to break pre-image-resistance of a random oracle with a s-bit salt and n-bit outputs succeeds with probability at most

$$\frac{T_1}{S \cdot N} + \frac{T_2}{N} \ ,$$

where $N = 2^n$, $S = 2^s$ and, once again, T_1 and T_2 are the offline- and online-query-complexity. This bound ends up being nearly *exact* in that there are offline- and online-only attacks achieving each of the two terms.

Unfortunately, the same approach via [15] yields only suboptimal bounds for other properties, such as the collision resistance of a salted random oracle. Here, we give a direct proof that shows a bound of order

$$\frac{T_1}{S \cdot \sqrt{N}} + \frac{T_2^2}{N} \ .$$

This proof is of independent interest, and uses a compression argument to bound the number of salts for which a collision is found in the preprocessing stage. And indeed, the first term is matched by an attack that finds collisions for $\Omega(\lceil T_1/\sqrt{N} \rceil)$ salts, issuing $\sqrt{N}$ queries for each of these salts.

Trade-offs for Merkle-Damgård. Our first set of results aims to show that salting prevents offline-online trade-offs–the best attack is either fully offline or fully online. However, we show that this is not true if we cannot afford to salt *each* call to a primitive. We focus in particular on the salted Merkle-Damgård (MD) construction [8,17], which has also been central to a recent wave of works in the context of space-time trade-offs [2,3,12]. Here, we are given a compression function $h : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^n$ and a message M that consists of B blocks $M_1, \ldots, M_B \in \{0,1\}^\ell$. To hash M, one sets the initial value y_0 to equal the salt a and computes the final hash y_B by iterating

$$y_i \leftarrow h(y_{i-1}, M_i) \ .$$

Here, we focus on the case of messages of length at most two, which, as in the case of space-time trade-offs [2], already captures many of the challenges. (In fact, we believe that going beyond requires significantly new techniques than those we explore in this paper.) For *pre-image-resistance*, we prove a bound (which we show to be tight) of the form (when ignoring constant factors and lower-order terms)

$$\frac{T_2}{N} + \frac{T_1 T_2}{N^2} + \frac{T_1^2}{N^3} \ .$$

The most interesting term is the middle one: it is leading, e.g., for $T_1 = N^{6/5}$ and $T_2 = N^{4/5}$, and suggests an inherent trade-off between offline and online

query-complexities. Indeed, this advantage is (roughly) matched by an actual attack that first evaluates $h(a_i, M)$ on N distinct M's for T_1/N different salts $a_1, \ldots, a_{T_1/N}$. Then, upon learning the value y to invert on, as well as the salt a, the online adversary spends its T_2 queries looking for $M \in \{0,1\}^n$ such that $h(a, M) = a_i$ for some $i \in [T_1/N]$, succeeding with probability $\Omega(T_1 T_2/N)$. Then, given it succeeds, the attacker knows N evaluations of $h(a_i, \cdot)$ and is thus likely able to find $M' \in \{0,1\}^n$ such that $h(a_i, M') = y$. Hence, (M, M') is a pre-image of y.

Collision-resistance of MD. Our most involved result is the analysis of the *collision-resistance* of the two-block MD construction, which in particular relies on a number of sophisticated compression arguments and results in a bound that we know to be only partially tight. Ignoring lower order terms and constant factors, we show a bound of order

$$\frac{T_2^2}{N} + \frac{T_1 T_2}{N^{3/2}} + \frac{T_1}{N^{5/4}} + \frac{T_1^2}{N^{7/3}} \ .$$

Here, we show matching attacks for all terms except the last one. This (potential) lack of tightness of the last term is due to our combinatorial analysis of a special type of offline-only attack. Namely, an offline attacker could repeatedly attempt to find a special type of collision called a *diamond* for a (potential) salt a, namely, four (distinct) queries $h(a, x_1) = y_1$, $h(a, x_2) = y_2$, $h(y_1, x_1') = z_1$, $h(y_2, x_2') = z_2$ such that $z_1 = z_2$. If the attacker finds a diamond for k salts, then in the online phase it wins with probability k/N (with no further query). Therefore, this boils down to proving a tail inequality on the number of salts for which a diamond is found with T_1 queries. This is challenging since in the regime $T_1 \gg N$ the combinatorics of random functions are not very well understood. The challenge stems from the fact that the "outer" queries $h(y_1, x_1') = z_1$ and $h(y_2, x_2') = z_2$ in one diamond could, individually, be part of diamonds for different salts. Our proof uses compression arguments to provide a suitable tail bound, but we leave it as an open problem to improve our analysis (or show it is tight).

Combining space and time. In conclusion, we observe that our approach is entirely dual to that of space-time trade-offs. The latter completely ignores the issue of *time* to produce advice, whereas we completely ignore the issue of advice size. The obvious question is whether both can be combined, and we currently lack good techniques to combine space and query-complexity.

Relationship with Multi-Instance Security. We note that a remark in [7] observed that a lower bound against multiple-discrete-log algorithms also yields lower bounds on the preprocessing time for discrete-log algorithms with preprocessing (observation attributed to Dan Bernstein). We can extend this approach to relate the advantage of an offline-online adversary with the advantage of an adversary playing a multi-instance game. However, we find that this does not give tight bounds for the advantage of offline-online adversaries that succeed with small (sub-constant) probability. In the full version [13], we illustrate this via an example.

2 Preliminaries

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ denote the set of all natural numbers and $\mathbb{N}_{>0} = \mathbb{N} \setminus \{0\}$. For $N \in \mathbb{N}_{>0}$, let $[N] = \{1, 2, \ldots, N\}$. For a set X, let $|X|$ be its size and X^+ denote one or more elements of X. For a set S and $r \in \mathbb{N}_{>0}$ such that $r \leqslant |\mathsf{S}|$, we denote using $\binom{\mathsf{S}}{r}$ the set of subsets of S with r elements. We denote $\mathsf{Fcs}(\mathsf{D}, \mathsf{R})$ the set of all functions mapping elements in D to the elements of R. Security notions are defined via games; for an example see Fig. 2. The probability that a game G outputs true is denoted using $\Pr[\mathsf{G}]$.

We let $x \xleftarrow{\$} \mathcal{D}$ denote sampling x according to the distribution $\mathcal{D}$. If D is a set, we overload notation and let $x \xleftarrow{\$} \mathsf{D}$ denote uniformly sampling from the elements of D. For a bit-string s we use $|s|$ to denote the number of bits in s. For a random variable X, we use $\mathsf{E}[X]$ to denote its expectation.

Merkle-Damgård. We recall the Merkle-Damgård hashing mechanism. For $n, \ell \in \mathbb{N}_{>0}$, let $h : \{0,1\}^n \times (\{0,1\}^\ell)^+ \to \{0,1\}^n$ be a compression function. We recursively define Merkle-Damgård (MD) hashing $\mathsf{MD}^h : \{0,1\}^n \times (\{0,1\}^\ell)^+ \to \{0,1\}^n$ as

$$\mathsf{MD}^h(a, M) = h(a, M)$$

for $a \in \{0,1\}^n, M \in \{0,1\}^\ell$ and

$$\mathsf{MD}^h(a, (M_1, M_2, \ldots, M_B)) = h(\mathsf{MD}^h(a, (M_1, M_2, \ldots, M_{B-1})), M_B)$$

for $a \in \{0,1\}^n$ and $M_1, \ldots, M_B \in \{0,1\}^\ell$. We refer to a as the *salt*.

The compression lemma. The compression lemma states that it is impossible to compress a random element in set $\mathcal{X}$ to a string shorter than $\log |\mathcal{X}|$ bits long, even relative to a random string.

Proposition 1 (E.g., [9]). *Let* Encode *be a randomized map from* $\mathcal{X}$ *to* $\mathcal{Y}$ *and let* Decode *be a randomized map from* $\mathcal{Y}$ *to* $\mathcal{X}$ *such that*

$$\Pr_{x \xleftarrow{\$} \mathcal{X}}[\mathsf{Decode}(\mathsf{Encode}(x)) = x] \geqslant \epsilon.$$

Then, $\log |\mathcal{Y}| \geqslant \log |\mathcal{X}| - \log(1/\epsilon)$.

Markov's inequality. We use Markov's inequality multiple times in this paper. We state it here for the sake of completeness.

Proposition 2. *Let* X *be a non-negative random variable and* $a > 0$. *Then*

$$\Pr[X \geqslant a] \leqslant \frac{\mathsf{E}[X]}{a}.$$

3 Offline-Online Trade-offs and the Role of Salting

We present some basic facts about offline-online trade-offs and discuss the role of salting. To do this, we define a notational framework that captures the generality of our statements.

3.1 A General Framework for Offline-Online Attacks

Games. We formalize security guarantees in cryptography using *games*, which we also use for security proofs. A game G describes an environment an adversary $\mathcal{A}$ can interact with, and the combination of G and $\mathcal{A}$ results in a random experiment $\mathsf{G}(\mathcal{A})$ (we refer to this as $\mathcal{A}$ "playing" the game G) which produces a Boolean output. We also denote this output as $\mathsf{G}(\mathcal{A})$.

Games with ideal primitives. We are interested in a special class of games that depend on an *ideal primitive*, such as a random oracle, random permutation, ideal cipher, etc. We model this via a distribution $\mathcal{I}$ on a set of functions. For example, a *random oracle* with (finite) domain D and range R would be modeled by the uniform distribution on $\mathsf{Fcs}(\mathsf{D},\mathsf{R})$.[1] Similarly, an ideal cipher with key space K and domain X can be modeled as a uniformly chosen function e from the set $\mathsf{Fcs}(\mathsf{K}\times\mathsf{X}\times\{-1,1\},\mathsf{X})$ such that $e(k,\cdot,1)$ is a permutation on X for all $k\in\mathsf{K}$, and $e(k,\cdot,-1)$ is its inverse. We can also model a variant of the generic-group model (GGM) [20] by looking at the uniform distribution of functions $f\in\mathsf{Fcs}(\mathbb{Z}_p\times\mathsf{X}\times\mathsf{X},\mathsf{X}\times\mathsf{X})$, where $|\mathsf{X}|=p$, and

$$\pi(x,l_1,l_2)=(\phi(x),\phi(\phi^{-1}(l_1)+\phi^{-1}(l_2)))\ ,$$

where $\phi:\mathbb{Z}_p\to\mathsf{X}$ is a bijective function.

Games with primitives. An *oracle* game G^{π} is one where both the adversary $\mathcal{A}$ and the game procedures are given access to an oracle π, from an understood set of possible functions π, which we refer to as *compatible* with the game G. We denote by $\mathsf{G}^{\pi}(\mathcal{A}^{\pi})$ both the experiment where $\mathcal{A}$ plays the game, and is given access to the same π as the game as well as the random variable denoting the output. We say that an (oracle) game G is *compatible* with an ideal primitive $\mathcal{I}$, if the range of $\mathcal{I}$ is a subset of the compatible oracles for G. We write specifically

$$\mathsf{Adv}^{\mathsf{G}}_{\mathcal{I}}(\mathcal{A})=\Pr\left[\mathsf{G}^{\pi}(\mathcal{A}^{\pi})\right] \tag{1}$$

where $\pi\leftarrow\!\!\$\ \mathcal{I}$. One could define a more general notion that permits other advantage formats (e.g., to model distinguishing notions). This is straightforward, and outside the scope of this paper.

Defining offline-online attacks. With the above formalism, given an oracle game G, we introduce a new oracle game $\mathsf{pre\text{-}G}$, which enhances the original game to model offline-online attacks. Both games preserve compatibility with any oracle. In particular, an adversary $\mathcal{A}$ is split into two parts, the offline adversary $\mathcal{A}_1$ and the online adversary $\mathcal{A}_2$. Initially, in the *offline stage*, $\mathcal{A}_1$ is given access *solely* to the game oracle π. At the end of this stage, $\mathcal{A}_1$ outputs a state st. Then, in the *online stage*, adversary $\mathcal{A}_2$ is initialized with state st and run in the game G. Both $\mathcal{A}_2$ and G are given access to the oracle π. Crucially,

[1] As usual, one must be more precise when formally defining a random oracle with $\mathsf{D}=\{0,1\}^*$, but we remain intentionally informal on this front; all of our examples can be assumed to work on a finite domain.

Game $\mathsf{pre\text{-}G}^{\pi}(\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2))$	Game $s\text{-}\mathsf{pre\text{-}G}^{\pi}(\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2))$
$\mathsf{st} \leftarrow \mathcal{A}_1^{\pi}$	$\mathsf{st} \leftarrow \mathcal{A}_1^{\pi}$
Return $\mathsf{G}^{\pi}(\mathcal{A}_2^{\pi}(\mathsf{st}))$	$a \leftarrow\!\!\$\ \{0,1\}^s$
	Return $\mathsf{G}^{\pi_a}(\mathcal{A}_2^{\pi}(\mathsf{st},a))$

Fig. 1. Offline-Online security games for an original game G. Left: the unsalted case. Right: the salted case. Here, π is meant to be sampled from a salted ideal primitive $\mathcal{I}_s$, and $\pi_a(\cdot) = \pi(a,\cdot)$, i.e., the primitive with salt fixed to a.

the game G might give $\mathcal{A}_2$ additional oracles plus additional initialization values, etc., which are not available in the offline stage. (This is formalized in Fig. 1 on the left). We colloquially refer to $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ as an offline-online adversary. Further, we say that $\mathcal{A}$ is a (T_1, T_2)-adversary if $\mathcal{A}_1$ makes at most T_1 queries and $\mathcal{A}_2$ makes at most T_2 queries. (Note that $\mathcal{A}_2$ could make additional game-dependent queries, which we would specify separately if necessary.) We overload notation and define advantage in terms of (T_1, T_2) as follows.

$$\mathsf{Adv}^{\mathsf{G}}_{\mathcal{I}}(T_1,T_2) = \max_{(T_1,T_2)\text{-adversaries } \mathcal{A}} \mathsf{Adv}^{\mathsf{G}}_{\mathcal{I}}(\mathcal{A}) \ . \tag{2}$$

Some basic facts. The following elementary fact, while straightforward, establishes some important baselines for when offline-online attacks are interesting. It relies on the basic observation that one can consider $\mathcal{A}_1$ and $\mathcal{A}_2$ to be a single online adversary.

Lemma 1. *Let G be a game compatible with the ideal primitive $\mathcal{I}$. For any (T_1, T_2)-adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{pre\text{-}G}}_{\mathcal{I}}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{G}}_{\mathcal{I}}(\mathcal{B}) \ .$$

Here, $\mathcal{B}$ makes $T_1 + T_2$ queries to the primitive, and its time-complexity is the sum of the time-complexities of $\mathcal{A}_1$ and $\mathcal{A}_2$. Further, for any game-dependent type of query, $\mathcal{B}$ makes the same number of queries as $\mathcal{A}_2$.

For an ideal primitive $\mathcal{I}$, we let $Q(\mathcal{I})$ be an upper bound on the number of queries needed by an adversary, given oracle access to $\pi \leftarrow\!\!\$\ \mathcal{I}$, to reconstruct π with probability 1. Then, the following also holds true and formalizes the fact that one can simulate an offline-online adversary by having the offline adversary first reconstruct π and then store it in st.

Lemma 2. *Let $\mathcal{I}$ be a primitive that is compatible with game G. For all offline-online adversaries $\mathcal{A}$, there exists a $(Q(\mathcal{I}), 0)$-adversary $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{pre\text{-}G}}_{\mathcal{I}}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{pre\text{-}G}}_{\mathcal{I}}(\mathcal{B})$$

Note that the adversary $\mathcal{B}$ could be much less efficient than $\mathcal{A}$; however, in many cases, we will study games that only target information-theoretic security, and time-complexity will not matter. It also naïvely follows that there always exists an *optimal* adversary that is a $(Q(\mathcal{I}), 0)$-adversary.

WHICH GAMES ARE INTERESTING? Some games are more interesting than others in the context of offline-online trade-offs, and the above two lemmas already provide some guidance.

Consider the problem of inverting a random permutation $\pi : \{0,1\}^n \to \{0,1\}^n$. It is well known that the best adversary takes time $\Omega(N)$ queries, where $N = 2^n$, to invert with constant probability. Then, Lemma 1 implies that any (T_1, T_2)-offline adversary needs $T_1 + T_2 = \Omega(N)$ to invert with constant probability. Thus, to get $T_2 = o(N)$ and beat the naïve inversion attack in the online phase, we need $T_1 = \Omega(N)$. Further, we already have an $(N, 0)$-adversary by Lemma 2, so we cannot really expect interesting trade-offs. The question becomes interesting only if we limit the state size between $\mathcal{A}_1$ and $\mathcal{A}_2$, which is exactly what is considered by prior works on space-time trade-offs.

This is in contrast to the setting of the discrete logarithm problem with preprocessing [7]. There, for a group of order p, by combining Lemma 1 with the well-known result by Shoup [20], we get that $T_1 + T_2 = \Omega(\sqrt{p})$. Lemma 2 guarantees only a $(p, 0)$ attacker, so we can expect that $T_2 = o(\sqrt{p})$ while still having $T_1 = o(p)$. And indeed, one can achieve the trade-off $T_1 \cdot T_2 \geqslant p$, as indicated in [7].

3.2 The Power of Salting

A special case of interest is that of *salting*, where the cryptographic primitive $\mathcal{I}$ permits an additional input–called a *salt*–that is chosen in the online phase of an attack.

GENERIC SALTING. Let $\mathcal{I}$ be an ideal primitive, whose range is a subset of $\mathsf{Fcs}(\mathsf{D}, \mathsf{R})$. We define its s-bit *salted* version, denoted $\mathcal{I}_s$, as the ideal primitive with range $\mathsf{Fcs}(\{0,1\}^s \times \mathsf{D}, \mathsf{R})$; sampling a function $\pi : \{0,1\}^s \times \mathsf{D} \to \mathsf{R}$ occurs by first sampling 2^s independent copies $\pi_a \xleftarrow{\$} \mathcal{I}$ for each $a \in \{0,1\}^s$ and then letting

$$\pi(a, x) = \pi_a(x) \ ,$$

for all $a \in \{0,1\}^s$ and $x \in \mathsf{D}$.

For any game G compatible with an ideal primitive $\mathcal{I}$, we can now define an s-bit salted version of the game, s-pre-G, that is compatible with $\mathcal{I}_s$. This is given on the right of Fig. 1. Essentially, we now sample a primitive $\pi \xleftarrow{\$} \mathcal{I}_s$ to which the adversary $\mathcal{A}$ is given access. However, in the online phase, the games themselves have access only to π_a for a randomly sampled salt a, which is revealed to only the adversary $\mathcal{A}_2$.

FROM SALTED TO UNSALTED GAMES. The following theorem relates the advantage of an offline-online adversary for the salted game with that of an adversary

for the original game. We provide an interpretation below (and use this lemma quantitatively in Sect. 4).

Theorem 1. *Let* G *be a game compatible with an ideal primitive* $\mathcal{I}$*. Let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a* (T_1, T_2) *offline-online adversary. Then, there exists an adversary* $\mathcal{B}$ *playing* G *such that*

$$\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{G}}_{\mathcal{I}_s}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{G}}_{\mathcal{I}}(\mathcal{B}) \ .$$

The adversary $\mathcal{B}$ *makes a number of queries, expressed as a random variable* T *with expectation* $\mathsf{E}[T] \leqslant T_1/2^s + T_2$*.*

Proof. Given access to π in the range of $\mathcal{I}$, the adversary $\mathcal{B}$ samples a random salt a from $\{0,1\}^s$ and then samples $\pi_{a'} \leftarrow\!\!\$\, \mathcal{I}$ for $a' \neq a$. It then simulates an execution of $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in s-pre-G as follows: it answers the ideal-primitive query of $\mathcal{A}$ for salt a using oracle queries to π, and those for salt $a' \neq a$ using the local evaluation of $\pi_{a'}$. It is immediate that $\mathcal{B}$ perfectly simulates the execution of s-pre-G to $\mathcal{A}$. Therefore, $\mathcal{A}$ wins if and only if $\mathcal{B}$ does and the claim about advantages follows.

Observe that $\mathcal{B}$ queries its ideal primitive, sampled from $\mathcal{I}$ only when it receives an ideal object query from $\mathcal{A}$ that is prefixed by the actual salt a. Since it is not given access to a when simulating $\mathcal{A}_1$, its queries are independent of a, and each of them is indeed on salt a with probability $1/2^s$. In contrast, $\mathcal{A}_2$ makes queries after learning a, and therefore, those queries are on salt a with probability upper bounded by one. By linearity of expectation, the total number of queries T to $\pi = \pi_a$ made by $\mathcal{B}$ satisfies $\mathsf{E}[T] \leqslant T_1/|\mathsf{S}| + T_2$, as we intended to show. □

Salting generically defeats preprocessing attacks. We illustrate one first main application of Theorem 1, i.e., the fact that salting generically defeats preprocessing in a *qualitative* sense. Here, "qualitative" means that we only look at the power of attacks that achieve large advantage. Subsequent sections (Sects. 4 and 5) take a more quantitative angle on this, studying the whole advantage curve.

We now say that a game G compatible with $\mathcal{I}$ is (T^*, ϵ)-hard if $\mathsf{Adv}^{\mathsf{G}}_{\mathcal{I}}(\mathcal{A}) \leqslant \epsilon$ for all T^*-query $\mathcal{A}^*$. We say that game G is (T^*, ϵ)-expected-hard if the same holds for all adversaries running in *expected* time at most T^*. The following fact is helpful.

Lemma 3. *If* G *is* $(T^*, 0.4)$*-hard, then it is* $(T^*/2, 0.9)$*-expected-hard.*

Proof. By contradiction, let $\mathcal{A}$ run in expected time at most $T^*/2$, and $\mathsf{Adv}^{\mathsf{G}}_{\mathcal{I}}(\mathcal{A}) > 0.9$. Then, build $\mathcal{B}$ that runs $\mathcal{A}$ for T^* queries and then aborts with some default answer if $\mathcal{A}$ did not finish running. Let T be the running time of $\mathcal{A}$. Then, for $\pi \leftarrow\!\!\$\, \mathcal{I}$,

$$\begin{aligned} \Pr[\mathsf{G}^{\pi}(\mathcal{B}^{\pi})] &= \Pr[\mathsf{G}^{\pi}(\mathcal{A}^{\pi}) \ \wedge \ T \leqslant T^*] \\ &\geqslant \Pr[\mathsf{G}^{\pi}(\mathcal{A}^{\pi})] - \Pr[T > T^*] > 0.9 - 0.5 = 0.4 \ , \end{aligned}$$

where we used Markov's inequality and the fact that $\mathsf{E}[T] \leqslant T^*/2$. □

Now, say that s-pre-G is (T_1, T_2, ϵ)-hard if for all (T_1, T_2)-adversaries $\mathcal{A}$, we have that $\mathsf{Adv}^{s\text{-pre-}\mathsf{G}}_{\mathcal{I}}(\mathcal{A}) \leqslant \epsilon$. Then, Lemma 3 and Theorem 1 yield the following corollary.

Corollary 1. *If* G *is* $(T^*, 0.4)$*-hard, then* s-pre-G *is* $(T_1, T_2, 0.9)$*-hard for any* T_1, T_2 *such that* $T_1/2^s + T_2 \leqslant T^*/2$.

This means that if a (T_1, T_2)-adversary is to achieve advantage larger than 0.9 in s-pre-G, then $T_1 \geqslant 2^s T^*/4$ or $T_2 \geqslant T^*/4$. In other words, in order to win s-pre-G with an advantage larger than 0.9, an attacker needs to either use online time which is (almost) as large as that of the best online attack achieving advantage 0.4 or run 2^s times that amount of time in the offline stage.

Moving on. We can easily revisit the remainder of this paper using what we saw in this section. First of all, our conclusion about salting applies only to large advantage adversaries since otherwise we cannot prove an analogue of Lemma 3. Section 4 examines *tight* exact bounds on the advantage of (T_1, T_2)-adversaries that hold for each choice of T_1 and T_2. Second, this conclusion applies only to the case where G salts *every call* to the primitive. In Sect. 5, we characterize the pre-image- and the collision-resistance of the salted MD construction against offline-online attacks and show that salting, while still useful, has a more limited effect.

4 Offline-Online Security of Salted Random Oracles

In this section, we study the security of salted monolithic random oracles against offline-online adversaries. Specifically, we consider the security properties of pre-image-resistance and collision-resistance. Our analysis begins by applying Theorem 1 in conjunction with [15, Theorem 1] to derive advantage upper bounds for offline-online adversaries against these properties. This approach already yields a tight bound for pre-image-resistance, but not for collision-resistance. We then use a non-generic technique to prove a tight bound for the latter.

4.1 Pre-image-resistance of a Salted Random Oracle

Oracle game PR^h in Fig. 2 formalizes the preimage-resistance of oracle h, which has co-domain $\{0,1\}^n$. In the game the adversary is given as input y, which is randomly sampled from $\{0,1\}^n$. It has oracle access to h and wins if it manages to output x such that $h(x) = y$.

We aim to upper bound the advantage of offline-online adversaries $\mathcal{A}$ against pre-image-resistance of salted random oracles. Let $H_{s,\ell,n}$ be the uniform distribution over $\mathsf{Fcs}(\{0,1\}^s \times \{0,1\}^\ell, \{0,1\}^n)$. The quantity of interest is $\mathsf{Adv}^{s\text{-pre-}\mathsf{PR}}_{H_{s,\ell,n}}(T_1, T_2)$. Using Theorem 1 and [15, Theorem 1], we get the following corollary.

Corollary 2. *Let* $T_1, T_2, n, s, \ell \in \mathbb{N}_{>0}$. *Then*

$$\mathsf{Adv}^{s\text{-pre-PR}}_{H_{s,\ell,n}}(T_1, T_2) \leqslant 5\left(\frac{T_1}{2^{s+n}} + \frac{T_2}{2^n}\right) .$$

Proof. We fix the adversary (T_1, T_2)-adversary $\mathcal{A}$ that maximizes $\mathsf{Adv}^{s\text{-pre-PR}}_{H_{s,\ell,n}}(T_1, T_2)$. From Theorem 1 we have that there exists an adversary $\mathcal{B}$ that makes at most T queries to its h oracle, where $\mathsf{E}[T] \leqslant T_1/2^s + T_2$ and

$$\mathsf{Adv}^{s\text{-pre-PR}}_{H_{s,\ell,n}}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{PR}}_{H_{s,\ell,n}}(\mathcal{B}) .$$

Game $\mathsf{PR}^h(\mathcal{B})$	Game $\mathsf{CR}^h(\mathcal{A})$
$x \leftarrow\!\!\$\ \{0,1\}^*$	$(M, M') \leftarrow \mathcal{A}^h$
$y \leftarrow h(x)$	If $M \neq M'$ and $h(M) = h(M')$
$x' \leftarrow \mathcal{B}^h(y)$	Return true
If $h(x') = y$:	Return false
Return true	
Return false	

Fig. 2. Left: Oracle Game PR^h for preimage-resistance of oracle h. Right: Oracle Game CR^h for collision-resistance of oracle h.

Using Theorem 1 in [15], we can show that $\mathsf{Adv}^{\mathsf{PR}}_{H_{s,\ell,n}}(\mathcal{B}) \leqslant \frac{5\mathsf{E}[T]}{2^n}$, which completes the proof.

Tightness. We remark that this bound is tight up to constant factors. To see the tightness of the term $T_2/2^n$, consider the online-only adversary that simply makes k distinct queries with the salt a. It fails only if all the queries have answer different from y, which happens with probability $(1 - 1/2^n)^{T_2} \leqslant e^{-T_2/2^n}$. Since $e^{-x} \leqslant 1 - x/2$ for $x \leqslant 1.5$, for $T_2 \leqslant 2^n$, $e^{-T_2/2^n} \leqslant 1 - T_2/2^{n+1}$. This means the adversary succeeds with probability at least $T_2/2^{n+1}$, meaning the second term in the bound is tight up to constant factors.

To see why the first term is tight, consider the adversary $\mathcal{A}_1$ which makes 2^n queries on different inputs for $k = T_1/2^n$ different salts (where T_1 is a multiple of 2^n). In the online phase, it simply checks whether it had made a query with salt a, that had output y; if so it returns the query input.

Let the set of k salts $\mathcal{A}_1$ had made queries on be S. For each salt in S, the probability that $\mathcal{A}_1$ had not made query with that salt that had answer y is at most $(1-1/2^n)^{2^n} \leqslant 1/e$. So, for each salt in S with probability at least $(1-1/e)$, $\mathcal{A}_1$ had made a query that had answer y. Now $\mathcal{A}$ wins if the a sampled is in S, and $\mathcal{A}_1$ has made a query with salt a that had answer y; this probability is at least $(1-1/e)k/2^s$ since a is sampled at random. Since $k = T_1/2^n$, it follows that the second term in the bound is tight as well.

4.2 Collision-Resistance of a Salted Random Oracle

Oracle game CR^h in Fig. 2 formalizes the collision-resistance of oracle h. In the game the adversary has oracle access to h and wins if it manages to output M, M' such that $M \neq M'$ and $h(M) = h(M')$.

We aim to upper bound the advantage of offline-online adversaries $\mathcal{A}$ against collision-resistance of salted random oracles. Let $H_{s,\ell,n}$ be the uniform distribution over $\mathsf{Fcs}(\{0,1\}^s \times \{0,1\}^\ell, \{0,1\}^n)$. We seek to tightly upper bound $\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{CR}}_{H_{s,\ell,n}}(T_1, T_2)$. Using Theorem 1 and [15, Theorem 1] we get the following corollary.

Corollary 3. *Let $T_1, T_2, n, s, \ell \in \mathbb{N}_{>0}$. Then*

$$\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{CR}}_{H_{s,\ell,n}}(T_1, T_2) \leqslant 5\sqrt{2}\left(\frac{T_1}{2^{s+n/2}} + \frac{T_2}{2^{n/2}}\right) .$$

Proof. We fix the adversary (T_1, T_2)-adversary $\mathcal{A}$ that maximizes $\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{CR}}_{H_{s,\ell,n}}(T_1, T_2)$. From Theorem 1 we have that there exists an adversary $\mathcal{B}$ such that it makes at most T queries to its h oracle, where $\mathsf{E}[T] \leqslant T_1/2^s + T_2$, and

$$\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{PR}}_{H_{s,\ell,n}}(\mathcal{A}) \leqslant \mathsf{Adv}^{\mathsf{PR}}_{H_{s,\ell,n}}(\mathcal{B}) .$$

Using [15, Theorem 1], we can show that $\mathsf{Adv}^{\mathsf{CR}}_{H_{s,\ell,n}}(\mathcal{B}) \leqslant 5\sqrt{\frac{2\mathsf{E}[T]^2}{2^n}}$, which concludes the proof.

Tight bound. The bound in Corollary 3 is suboptimal. In Theorem 2, we obtain a better bound for $\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{CR}}_{H_{s,\ell,n}}(T_1, T_2)$.

Theorem 2. *Let $n, s, \ell, T_1, T_2 \in \mathbb{N}_{>0}$. Let $H_{s,\ell,n}$ be the uniform distribution on $\mathsf{Fcs}(\{0,1\}^s \times \{0,1\}^\ell, \{0,1\}^n)$. Then, we have that*

$$\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{CR}}_{H_{s,\ell,n}}(T_1, T_2) \leqslant \frac{\binom{T_2}{2}}{2^n} + \frac{T_2 T_1}{2^{s+n}} + \frac{eT_1}{2^{s+n/2}} + \frac{n}{2^{s+1}} + \frac{1}{2^n} .$$

Tightness. We argue that this bound is tight up to constant factors. Initially, observe that for $T_2 \geqslant 2^{n/2}$, the right side becomes greater than one, and the bound always holds. For $T_2 \leqslant 2^{n/2}$, we have that $\frac{T_1 T_2}{2^{n+s}} \leqslant \frac{eT_1}{2^{s+n/2}}$. Therefore, the term $\frac{T_1 T_2}{2^{n+s}}$ is never the dominant term in the bound, and it suffices to show attacks that achieve advantage of the order $\frac{\binom{T_2}{2}}{2^n}$ and $\frac{eT_1}{2^{s+n/2}}$ to show that this bound is tight. A birthday style attack with T_2 queries achieves advantage of the order $\frac{\binom{T_2}{2}}{2^n}$. Finally, we prove the following theorem to show that term $\frac{eT_1}{2^{s+n/2}}$ is tight up to constant factors.

Theorem 3. *Let $T_1, s, n, \ell \in \mathbb{N}_{>0}$ such that n is a multiple of 2 and T_1 is a multiple of $2^{n/2+1}$. Let $H_{s,\ell,n}$ be the uniform distribution over $\mathsf{Fcs}(\{0,1\}^s \times \{0,1\}^\ell, \{0,1\}^n)$. Then there exists a $(T_1, 0)$-adversary $\mathcal{A}$ such that*

$$\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{CR}}_{H_{s,\ell,n}}(\mathcal{A}) \geqslant \frac{(1 - 1/e)T_1}{2^{s+n/2+1}} .$$

We defer the formal proof of this theorem to the full version [13]. We next prove Theorem 2.

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be the (T_1, T_2)-offline-online adversary that maximizes $\mathsf{Adv}^{s\text{-pre-CR}}_{H_{s,\ell,n}}(T_1, T_2)$. We can treat $\mathcal{A}$ as deterministic by fixing its randomness that maximizes its advantage.

We seek to upper bound the probability that the adversary $\mathcal{A}$ finds a one-block collision for the randomly chosen salt a that it gets as input in its online phase. We can assume without loss of generality that if $\mathcal{A}$ outputs a collision, it must have made the relevant queries either in the offline or the online phase. This is without loss of generality because if $\mathcal{A}$ does not make one of these queries, we can construct a $(T_1, T_2 + 2)$-offline-online adversary $\mathcal{A}'$ that does whatever $\mathcal{A}$ does, and at the end of its online phase, makes the two relevant queries if not made earlier after $\mathcal{A}$ outputs M, M'. The term T_2 would then be replaced by $T_2 + 2$ in our bounds; for ease of readability, we omit this.

Also, without loss of generality we can assume that no query across the offline and online phases is repeated because the adversary can simply remember the query answer since we do not restrict its memory or the amount of advice it can pass on from offline to the online phase.

We define the following three events.

1. $\mathsf{onecoll}_{\mathsf{on}}$: $\mathcal{A}_2$ makes two queries $h(a, M) = z$, $h(a, M') = z$ for some $M \neq M'$
2. $\mathsf{onecoll}_{\mathsf{offon}}$: $\mathcal{A}_1$ makes a query $h(a, M) = z$, and $\mathcal{A}_2$ makes a query whose answer is z
3. $\mathsf{onecoll}_{\mathsf{off}}$: $\mathcal{A}_1$ makes two queries $h(a, M) = z$, $h(a, M') = z$ for some $M \neq M'$

Observe that if none of $\mathsf{onecoll}_{\mathsf{on}}, \mathsf{onecoll}_{\mathsf{offon}}, \mathsf{onecoll}_{\mathsf{off}}$ happen, $\mathcal{A}$ cannot find a collision. We have that

$$\Pr\left[s\text{-pre-}\mathsf{CR}^h_{H_{s,\ell,n}}(\mathcal{A})\right] \leqslant \Pr\left[\mathsf{onecoll}_{\mathsf{on}}\right] + \Pr\left[\mathsf{onecoll}_{\mathsf{offon}}\right] + \Pr\left[\mathsf{onecoll}_{\mathsf{on}}\right] . \quad (3)$$

We upper bound the probability of these three events one by one.

Upper bounding $\Pr[\mathsf{onecoll}_{\mathsf{on}}]$. This event happens only if $\mathcal{A}_2$ makes two queries that collide. The probability of any two queries of $\mathcal{A}_2$ colliding is $1/2^n$. Using a union bound over all pairs of queries of $\mathcal{A}_2$, we have

$$\Pr\left[\mathsf{onecoll}_{\mathsf{on}}\right] \leqslant \frac{\binom{T_2}{2}}{2^n} . \quad (4)$$

Upper bounding $\Pr[\mathsf{onecoll}_{\mathsf{offon}}]$. Observe that $\mathsf{onecoll}_{\mathsf{offon}}$ happens only if there is an online query that has the same answer as one of the offline queries that had input salt a. There are a total of T_1 offline queries, and a is random. Therefore, the expected number of offline queries with salt a is $T_1/2^s$. We have that

$$\begin{aligned}\Pr\left[\mathsf{onecoll}_{\mathsf{offon}}\right] &\leqslant \sum_{k=0}^{T_1} \Pr\left[\text{There are } k \text{ offline queries with salt } a\right] \frac{kT_2}{2^n} \\ &= \mathsf{E}\left[\text{Number of offline queries with salt } a\right] \frac{T_2}{2^n} \\ &= \frac{T_1 T_2}{2^{s+n}} \,. \end{aligned} \tag{5}$$

Upper bounding $\Pr\left[\mathsf{onecoll}_{\mathsf{off}}\right]$. The main challenge of this proof is proving an upper bound on $\Pr\left[\mathsf{onecoll}_{\mathsf{off}}\right]$. We do it as follows: we define an event $\mathsf{off\text{-}oneblk}\text{-}k$ since there are k different salts for which a one-block collision has been found in the offline phase. We have that for any k,

$$\Pr\left[\mathsf{onecoll}_{\mathsf{off}}\right] \leqslant \Pr\left[\mathsf{onecoll}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}oneblk}\text{-}k\right] + \Pr\left[\mathsf{off\text{-}oneblk}\text{-}k\right] \,. \tag{6}$$

Since $\mathsf{onecoll}_{\mathsf{off}}$ happens if for the salt a that is chosen uniformly at random from $\{0,1\}^s$, $\mathcal{A}_1$ had queried $h(a,M), h(a,M')$ that have the same answer, we have that $\Pr\left[\mathsf{onecoll}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}oneblk}\text{-}k\right] \leqslant k/2^s$. We upper bound $\Pr\left[\mathsf{off\text{-}oneblk}\text{-}k\right]$ using a compression argument.

The encoding procedure encodes the random oracle h as follows.

1. It runs $\mathcal{A}_1^h$ and initializes a list L to the empty list and a set S to the empty set.
2. For every query $h(a,M)$ made by $\mathcal{A}_1$, it does the following:
   (a) Let $z = h(a,M)$. If there was *exactly one* earlier query by $\mathcal{A}_1$ of the form (a,M') for some $M' \neq M$ that had answer z, and $|\mathsf{S}| < 2k$, it adds the index of the query $h(a,M')$ and the current query to S.
   (b) Otherwise, it adds $h(a,M)$ to L.
3. It appends the evaluation of h on the points not queried by $\mathcal{A}_1$ to L in the lexicographical order of the inputs.
4. If $|\mathsf{S}| < 2k$ it outputs $\varnothing$; otherwise, it outputs L, S.

The decoding procedure works as follows.

1. If the encoding is $\varnothing$, it aborts.
2. It runs $\mathcal{A}_1^h$.
3. For every query $h(a,M)$ made by $\mathcal{A}_1$, it does the following:
   (a) If the index of the query is in S, and there is an earlier query $h(a,M')$ for some M' by $\mathcal{A}_1$ such that its index is in S, answer this query with $h(a,M')$.
   (b) Otherwise, it removes the element in front of L and answers with that.
4. It populates h on the points not queried by $\mathcal{A}_1$ in the lexicographical order by the remaining entries of L

Correctness of decoding: For adversary $\mathcal{A}_1$ that causes the event $\mathsf{off\text{-}oneblk}\text{-}k$ to happen, the encoding algorithm will never return $\varnothing$ because by the definition of $\mathsf{off\text{-}oneblk}\text{-}k$ there will be at least k different salts a_i for which $\mathcal{A}_1$ queries $h(a_i,M_i) = z_i$, $h(a_i,M_i') = z_i$; meaning the size of S will be $2k$. For such an

adversary $\mathcal{A}$ it is easy to verify the decoding algorithm will always produce the correct output because for the answers of h that were not added to L, the decoding algorithm recovers them using the set of indices S. Therefore, we have that

$$\Pr[\text{ Decoding is correct }] \geqslant \Pr[\mathsf{off\text{-}oneblk}\text{-}k] \ .$$

The size of the output space of the encoding algorithm is upper bounded by $\binom{T_1}{2k} \cdot (2^n)^{2^{s+\ell}-k}$. The size of the input space is $(2^n)^{2^{s+\ell}}$. From the compression lemma (Proposition 1), we have that

$$\Pr[\mathsf{off\text{-}oneblk}\text{-}k] \leqslant \Pr[\text{ Decoding is correct }] \leqslant \frac{\binom{T_1}{2k}}{2^{kn}} \ .$$

We $k = \max\left(\frac{eT_1}{2^{n/2}}, n/2\right)$. If $\max\left(\frac{eT_1}{2^{n/2}}, n/2\right) = \frac{eT_1}{2^{n/2}}$, then $k \geqslant n/2$. Therefore,

$$\frac{\binom{T_1}{2k}}{2^{nk}} \leqslant \left(\frac{eT_1}{2^{n/2}(2k)}\right)^{2k} \leqslant \left(\frac{1}{2}\right)^n \ .$$

If $\max\left(\frac{eT_1}{2^{n/2}}, n/2\right) = n/2$, then $k \geqslant \frac{eT_1}{2^{n/2}}$. Therefore,

$$\frac{\binom{T_1}{2k}}{2^{nk}} \leqslant \left(\frac{eT_1}{2^{n/2}(2k)}\right)^{2k} \leqslant \left(\frac{1}{2}\right)^n \ .$$

Therefore, for $k = \max\left(\frac{eT_1}{2^{n/2}}, n/2\right)$, $\Pr[\mathsf{off\text{-}oneblk}\text{-}k] \leqslant 1/2^n$. Therefore, from (6) we have

$$\Pr[\mathsf{onecoll}_{\mathsf{off}}] \leqslant \frac{eT_1}{2^{s+n/2}} + \frac{n}{2^{s+1}} + \frac{1}{2^n} \ . \tag{7}$$

Plugging (4), (5) and (7) into (3) gives us that

$$\mathsf{Adv}^{s\text{-}\mathsf{pre}\text{-}\mathsf{CR}}_{H_{s,\ell,n}}(\mathcal{A}) \leqslant \frac{\binom{T_2}{2}}{2^n} + \frac{T_1T_2}{2^{n+s}} + \frac{eT_1}{2^{s+n/2}} + \frac{n}{2^{s+1}} + \frac{1}{2^n} \ .$$

□

5 Offline-Online Security of Two-Block Merkle-Damgård

In previous sections, we focused on proving security guarantees against offline-online attacks on constructions where every query to the ideal primitive is salted. Here, we will see an example of a construction (Merkle Damgård) where only the first query to the underlying primitive (random oracle) is salted. Specifically, we will study the pre-image-resistance and collision-resistance for two-block Merkle-Damgård.

The main takeaway from this section is that for primitives that are not salted for every call, we can have parameter regimes where a term of the form T_1T_2

dominates the bound, meaning that in these regimes there are trade-offs between the number of offline and online queries for the advantage to be close to one. This contrasts with what we saw earlier in Sect. 3.2. There we demonstrated that for constructions that salt every query to the ideal primitive, an adversary must have either online time close to the online time of the best online-only attack that attains an advantage close to one, or offline time close to the best offline-only attack that attains an advantage close to one to achieve an advantage close to one.

5.1 Pre-image-resistance of Two-Block Merkle-Damgård

In this section we study the offline-online attacks against the pre-image-resistance of the two-block Merkle-Damgaard (MD) construction. Pre-image-resistance of two-block Merkle-Damgård is formalized in the game $\mathsf{2\text{-}PR\text{-}MD}_n^h$ in Fig. 3. A salt a and a value y are sampled uniformly at random from $\{0,1\}^n$ and given as input to $\mathcal{A}$. $\mathcal{A}$ can make queries to h and wins if it outputs a message M that is one or two blocks long whose MD evaluation with a is y.

Game $\mathsf{2\text{-}PR\text{-}MD}_n^h(\mathcal{A})$
$a \leftarrow\!\!\$\ \{0,1\}^n$
$y \leftarrow\!\!\$\ \{0,1\}^n$
$M \leftarrow \mathcal{A}^h(a,y)$
If $|M| \in \{\ell, 2\ell\}$ and $\mathsf{MD}^h(a,M) = y$
  Return true
Return false

Fig. 3. Oracle game $\mathsf{2\text{-}PR\text{-}MD}_n^h$ formalizing the pre-image-resistance of the two-block MD construction.

Let $H_{n,\ell,n}$ be the uniform distribution over $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^\ell, \{0,1\}^n)$. We are interested in proving an upper bound on $\mathsf{Adv}^{\mathsf{pre\text{-}2\text{-}PR\text{-}MD}}_{H_{n,\ell,n}}(T_1,T_2)$. We prove the following theorem.

Theorem 4. *Let $T_1, T_2, n, \ell \in \mathbb{N}_{>0}$. Let $H_{n,\ell,n}$ be the uniform distribution over $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^\ell, \{0,1\}^n)$. Then*

$$\mathsf{Adv}^{\mathsf{pre\text{-}2\text{-}PR\text{-}MD}}_{H_{n,\ell,n}}(T_1,T_2) \leqslant \frac{2T_2+1}{2^n} + \frac{T_1T_2 + nT_1 + T_1}{2^{2n}} + \frac{4eT_1^2}{2^{3n}}\ .$$

Offline-online trade-offs. Note that in the regime $T_1 = 2^{n(1+\epsilon)}, T_2 = 2^{n(1-\epsilon)}$ for $0 < \epsilon < 1/2$, the term $T_1T_2/2^{2n}$ dominates the bound, meaning there is an offline-online query trade-off in that regime.

Tightness. We show that this bound is tight up to factors of n. Observe that the dominant terms are of the order $T_2/2^n$, $T_1T_2/2^{2n}$, and $T_1^2/2^{3n}$. We briefly describe how we show this.

The tightness of $T_2/2^n$ follows easily – the online only attack simply makes T_2 queries with the given salt. The advantage of this attack is at least $(1-1/e)T_2/2^n$ for $T_2 \leqslant 2^n$, as argued in the tightness discussion for pre-image-resistance of the salted random oracle.

The following theorem proves that the term $T_1T_2/2^{2n}$ is tight.

Theorem 5. *Let $T_1, T_2, n, \ell \in \mathbb{N}_{>0}$ such that T_1 is a multiple of 2^n, $T_1T_2 \leqslant 2^{2n}$. Let $H_{n,\ell,n}$ be the uniform distribution over $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^\ell, \{0,1\}^n)$. Then there exists a (T_1, T_2)-adversary such that*

$$\mathsf{Adv}^{\text{2-PR-MD}}_{H_{n,\ell,n}}(\mathcal{A}) \geqslant \frac{T_1T_2(1-2/e)}{2^{2n+1}} .$$

We defer the proof of this theorem to the full version [13].

The following theorem proves that the term $T_1^2/2^{3n}$ is tight.

Theorem 6. *Let $T_1, n, \ell \in \mathbb{N}_{>0}$ such that T_1 is a multiple of 2^{n+1}, $T_1 \leqslant 2^{3n/2}$. Let $H_{n,\ell,n}$ be the uniform distribution over $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^\ell, \{0,1\}^n)$. Then there exists a $(T_1, 0)$-adversary such that*

$$\mathsf{Adv}^{\text{2-PR-MD}}_{H_{n,\ell,n}}(\mathcal{A}) \geqslant \frac{T_1^2(1-2/e)}{2^{3n+4}} .$$

We defer the proof of this theorem to the full version [13].

Game $\mathsf{H}(\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2))$	Oracle $\mathrm{H}(a', M')$
$h \leftarrow\!\!\$\ \mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^\ell, \{0,1\}^n)$	$\tau \leftarrow \tau \cup \{((a', M'), h(a', M'))\}$
win $\leftarrow$ false	If $\exists M' : ((a, M'), y) \in \tau$:
oneblkinv, twoblkinv $\leftarrow$ false	win $\leftarrow$ true
$\tau \leftarrow []$, $a, y \leftarrow \perp$	oneblkinv $\leftarrow$ true
st $\leftarrow \mathcal{A}_1^{\mathrm{H}}$	If $\exists M', M'', z$:
$a \leftarrow\!\!\$\ \{0,1\}^n$	$((a, M'), z), ((z, M''), y) \in \tau$:
$y \leftarrow\!\!\$\ \{0,1\}^n$	win $\leftarrow$ true
If $\exists M : ((a, M), y) \in \tau$:	twoblkinv $\leftarrow$ true
win $\leftarrow$ true	Return $h(a', M')$
oneblkinv $\leftarrow$ true	
If $\exists M, M', z$:	
$((a, M), z), ((z, M'), y) \in \tau$:	
win $\leftarrow$ true	
twoblkinv $\leftarrow$ true	
$\mathcal{A}_2^{\mathrm{H}}(\mathsf{st}, a, y)$	
Return win	

Fig. 4. H using in the analysis of pre-image-resistance of two-block MD. The events introduced in this game are marked in red. (Color figure online)

Proof (Theorem 4). Let $\mathcal{A}$ be the (T_1, T_2)-adversary that maximizes $\mathsf{Adv}^{\mathsf{2\text{-}PR\text{-}MD}}_{H_{n,\ell,n}}(T_1, T_2)$. Without loss of generality $\mathcal{A}$ is deterministic and does not repeat any queries. We also assume without loss of generality that $\mathcal{A}$ makes all the queries needed to compute the MD evaluation of the messages it outputs.

We can formulate an alternate version of the game for pre-image-resistance of MD in game H in Fig. 4. Note that whenever $\mathcal{A}$ wins $\mathsf{2\text{-}PR\text{-}MD}^h_n$, it has to win H because from our assumption that $\mathcal{A}$ makes all the queries needed to compute the MD evaluation of the messages it outputs, it follows that at least one of the following happens.

- $\mathcal{A}$ makes a query $h(a, M') = y$ – meaning it has found a one-block message M' whose MD evaluation with salt a is y.
- $\mathcal{A}$ queries $h(a, M') = z$ and $h(z, M'') = y$ – meaning it has found a two-block message (M', M'') whose MD evaluation with salt a is y.

In either of these cases the flag win is set in H, meaning $\mathcal{A}$ wins the game. Therefore,

$$\mathsf{Adv}^{\mathsf{2\text{-}PR\text{-}MD}}_{H_{n,\ell,n}}(\mathcal{A}) \leqslant \Pr\left[\mathsf{H}(\mathcal{A})\right] \ .$$

Note that win is set to true H only if one of $\mathsf{oneblkinv}, \mathsf{twoblkinv}$ is set to true. It follows that

$$\Pr\left[\mathsf{H}(\mathcal{A})\right] = \Pr\left[\mathcal{A} \text{ sets } \mathsf{win}\right] \leqslant \Pr\left[\mathsf{oneblkinv}\right] + \Pr\left[\mathsf{twoblkinv}\right] \ .$$

We show that

$$\Pr\left[\mathsf{oneblkinv}\right] \leqslant \frac{T_2}{2^n} + \frac{T_1}{2^{2n}} \ ,$$

and

$$\Pr\left[\mathsf{twoblkinv}\right] \leqslant \frac{T_2 + 1}{2^n} + \frac{T_1 T_2 + n T_1}{2^{2n}} + \frac{4 e T_1^2}{2^{3n}} \ .$$

Putting it all together would give us the theorem.

We next upper bound $\Pr\left[\mathsf{oneblkinv}\right], \Pr\left[\mathsf{twoblkinv}\right]$.

Towards upper bounding $\Pr\left[\mathsf{oneblkinv}\right]$, we define the two following events:

1. $\mathsf{oneblkinv}_{\mathsf{off}}$: $\mathcal{A}_1$ makes a query with input salt a and output y
2. $\mathsf{oneblkinv}_{\mathsf{on}}$: $\mathcal{A}_2$ makes a query which has output y

It is easy to see that if $\mathsf{oneblkinv}$ happens, then at least one of $\mathsf{oneblkinv}_{\mathsf{off}}$, $\mathsf{oneblkinv}_{\mathsf{on}}$ has to happen. Therefore

$$\Pr\left[\mathsf{oneblkinv}\right] \leqslant \Pr\left[\mathsf{oneblkinv}_{\mathsf{off}}\right] + \Pr\left[\mathsf{oneblkinv}_{\mathsf{on}}\right] \ .$$

We first upper bound $\Pr\left[\mathsf{oneblkinv}_{\mathsf{on}}\right]$. Each query by $\mathcal{A}_2$ has answer y with probability $1/2^n$. Using a union bound over all queries of $\mathcal{A}_2$, we have that

$$\Pr\left[\mathsf{oneblkinv}_{\mathsf{on}}\right] \leqslant T_2/2^n \ .$$

We next upper bound $\Pr[\mathsf{oneblkinv}_{\mathsf{off}}]$. Consider the set of (s, y) pairs such that there is a query by $\mathcal{A}_1$ with input salt s and answer y. There are at most T_1 such pairs. Note that, $\mathsf{oneblkinv}_{\mathsf{off}}$ happens only if (a, y) which is sampled uniformly at random, is among those at most T_1 pairs. Hence,

$$\Pr[\mathsf{oneblkinv}_{\mathsf{off}}] \leqslant T_1/2^{2n} .$$

Putting this together, we have the required bound on $\Pr[\mathsf{oneblkinv}]$.

We next upper bound $\Pr[\mathsf{twoblkinv}]$. We define the three following events.

1. $\mathsf{twoblkinv}_{\mathsf{off}}$: $\mathcal{A}_1$ makes queries $h(a, M) = z$ and $h(z, M') = y$ for some M, M', z
2. $\mathsf{twoblkinv}_{\mathsf{offon}}$: $\mathcal{A}_1$ makes a query $h(z, M) = y$ and $\mathcal{A}_2$ makes a query $h(a, M') = z$ for some M, M', z
3. $\mathsf{twoblkinv}_{\mathsf{on}}$: $\mathcal{A}_2$ makes a query with answer y

It is easy to see that if $\mathsf{twoblkinv}$ happens then at least one of $\mathsf{twoblkinv}_{\mathsf{off}}$, $\mathsf{twoblkinv}_{\mathsf{offon}}$, $\mathsf{twoblkinv}_{\mathsf{on}}$ has to happen. Therefore,

$$\Pr[\mathsf{twoblkinv}] \leqslant \Pr[\mathsf{twoblkinv}_{\mathsf{off}}] + \Pr[\mathsf{twoblkinv}_{\mathsf{offon}}] + \Pr[\mathsf{twoblkinv}_{\mathsf{on}}] . \quad (8)$$

We upper bound these probabilities one by one starting with $\Pr[\mathsf{twoblkinv}_{\mathsf{on}}]$. Observe that every query by $\mathcal{A}_2$ has probability $1/2^n$ of having answer y. Using a union bound over all queries of $\mathcal{A}_2$, it follows that

$$\Pr[\mathsf{twoblkinv}_{\mathsf{on}}] \leqslant \frac{T_2}{2^n} . \quad (9)$$

We next upper bound $\Pr[\mathsf{twoblkinv}_{\mathsf{offon}}]$. Observe that this event happens only if $\mathcal{A}_2$ makes a query that has answer z such that $\mathcal{A}_1$ made a query with salt z that had answer y. Therefore, using total probability

$$\begin{aligned}\Pr[\mathsf{twoblkinv}_{\mathsf{offon}}] &\leqslant \sum_{k=1}^{T_1} \Pr[\mathcal{A}_1 \text{ made } k \text{ queries with answer } y] \frac{kT_2}{2^n} \\ &= \frac{T_2}{2^n} \sum_{k=1}^{T_1} \mathsf{E}[\text{Number of queries of } \mathcal{A}_1 \text{ with answer } y] \\ &= \frac{T_2 T_1}{2^{2n}} . \end{aligned} \quad (10)$$

The last equality follows since each query of $\mathcal{A}_1$ has answer y with probability $1/2^n$.

Finally, we upper bound $\Pr[\mathsf{twoblkinv}_{\mathsf{off}}]$. For this we initially take a short detour and define the event $(m+1)$-col as the event that $\mathcal{A}_1$ has made $m+1$ distinct random oracle queries, all of which have the same answer. We claim that

$$\Pr\left[\mathsf{twoblkinv}_{\mathsf{off}} \,\middle|\, \neg(m+1)\text{-}\mathsf{col}\right] \leqslant \frac{mT_1}{2^{2n}} .$$

This is because if $(m+1)$-col does not happen, there can be at most mT_1 pairs of queries such that the answer of the one query of the pair is the input of the other query (as otherwise there would be a $m+1$-multi-collision since there are T_1 queries made by $\mathcal{A}_1$). Now, $\mathsf{twoblkinv}_{\mathsf{off}}$ happens only if (a, y) that is sampled uniformly at random is such that one of these at most mT_1 pairs have a as the salt for one query and y as the answer of the other query. This happens with probability at most $\frac{mT_1}{2^{2n}}$.

Finally, we upper bound $\Pr[(m+1)\text{-}\mathsf{col}]$. For any subset of $m+1$ queries made by $\mathcal{A}_1$, the probability that they have the same answer is $1/2^{nm}$. Using a union bound over all possible $m+1$ sized subsets of the queries of $\mathcal{A}_1$, we have that

$$\Pr[(m+1)\text{-}\mathsf{col}] \leqslant \frac{\binom{T_1}{m+1}}{2^{nm}} .$$

We let $m = \max\left(n, \frac{4eT_1}{2^n}\right)$. If $n \leqslant \frac{4eT_1}{2^n}$, we have that $m = \frac{4eT_1}{2^n} \geqslant n$. Therefore,

$$\Pr[(m+1)\text{-}\mathsf{col}] \leqslant \frac{\binom{T_1}{m+1}}{2^{mn}} \leqslant \left(\frac{eT_1}{(m+1)2^n}\right)^{m+1} 2^n \leqslant \left(\frac{1}{4}\right)^m 2^n \leqslant \left(\frac{1}{2}\right)^n .$$

Otherwise, if $n > \frac{4eT_1}{2^n}$, we have that $m = n > \frac{4eT_1}{2^n}$. Therefore,

$$\Pr[(m+1)\text{-}\mathsf{col}] \leqslant \frac{\binom{T_1}{m+1}}{2^{nm}} \leqslant \left(\frac{eT_1}{(m+1)2^n}\right)^{m+1} 2^n \leqslant \left(\frac{1}{4}\right)^{m+1} 2^n \leqslant \left(\frac{1}{2}\right)^n .$$

Therefore, we have that for $m = \max\left(n, \frac{4eT_1}{2^n}\right)$, $\Pr[(m+1)\text{-}\mathsf{col}] \leqslant 1/2^n$. Hence

$$\Pr[\mathsf{twoblkinv}_{\mathsf{off}}] \leqslant \frac{nT_1}{2^{2n}} + \frac{4eT_1^2}{2^{3n}} + \frac{1}{2^n} . \tag{11}$$

Plugging (9) to (11) into (8) gives us the required bound for $\Pr[\mathsf{twoblkinv}]$ and concludes the proof. □

5.2 Collision-Resistance of Two-Block Merkle-Damgård

In this section, we study the collision-resistance of two-block Merkle-Damgård (MD) against offline-online adversaries. Collision-resistance of two-block MD is formalized by the oracle game $2\text{-}\mathsf{CR}\text{-}\mathsf{MD}_n^h$ in Fig. 5. In this game a salt a is picked at random from $\{0,1\}^n$ that is given to the adversary $\mathcal{A}$. The adversary $\mathcal{A}$ has oracle access to h, and wins if it can output two messages M, M' that are distinct; both at most 2 blocks long, and satisfy $\mathsf{MD}^h(a, M) = \mathsf{MD}^h(a, M')$.

The game $\mathsf{pre}\text{-}2\text{-}\mathsf{CR}\text{-}\mathsf{MD}_n^h$ captures the collision-resistance of 2-block MD against offline-online adversaries. We prove the following upper bound on $\mathsf{Adv}^{\mathsf{pre}\text{-}2\text{-}\mathsf{CR}\text{-}\mathsf{MD},n}_{H_{n,\ell,n}}$.

Theorem 7. *Let $T_1, T_2, s, \ell, n \in \mathbb{N}_{>0}$. Let $H_{n,\ell,n}$ be the uniform distribution on* $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^\ell, \{0,1\}^n)$.

$$\mathsf{Adv}^{\mathsf{pre\text{-}2\text{-}CR\text{-}MD}}_{H_{s,\ell,n}}(\mathcal{A}) \leqslant \frac{2T_2^2 + nT_2/2 + 3n^2/2 + 99n/2 + 33}{2^n} + \left(\frac{T_1T_2}{2^{3n/2}}\right)(n^2 + 5n + 83) + \left(\frac{T_1}{2^{5n/4}}\right)(53n + 14n^{1/2} + 56n^{1/3} + 342) + 468\left(\frac{T_1^2}{2^{7n/3}}\right) .$$

Offline-online trade-offs. Note that, in the regime of parameters $T_1 = 2^{n(1+\epsilon)}, T_2 = 2^{n(1/2-\epsilon)}$ for $0 < \epsilon < 1/6$, the term $T_1T_2/2^{3n/2}$ dominates the bound, i.e., there is a trade-off between the number of offline and online queries in that regime.

Game $\mathsf{2\text{-}CR\text{-}MD}^h_n(\mathcal{A})$
$a \leftarrow\!\!\$\ \{0,1\}^n$
$(M, M') \leftarrow \mathcal{A}^h(a)$
If $|M|, |M'| \in \{\ell, 2\ell\}$ and $M \neq M'$ and $\mathsf{MD}^h(a, M) = \mathsf{MD}^h(a, M')$
  Return true
Return false

Fig. 5. Oracle game $\mathsf{2\text{-}CR\text{-}MD}^h_n$ formalizing collision-resistance of two-block MD.

Tightness of the bound. We show that the first three terms in the above bound are tight by giving matching attacks. We could not find an attack matching the last term in the bound and leave improving it or showing it tight to be future research.

We briefly describe how we show the other terms to be tight. The first term is dominated by $T_2^2/2^n$ – we can show that this is tight up to constant factors using the birthday attack, which achieves advantage of the order $T_2^2/2^n$.

In the second term, ignoring constants and powers of n, the dominant factor is $\frac{T_1T_2}{2^{3n/2}}$. We prove this theorem to show that it is tight.

Theorem 8. *Let $T_1, T_2, n, \ell \in \mathbb{N}_{>0}$ such that n is a multiple of 2, T_1 is a multiple of $2^{n/2+1}$, and $T_1T_2 \leqslant 2^{3n/2}$. Let $H_{n,\ell,n}$ be the uniform distribution over* $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^\ell, \{0,1\}^n)$. *There exists a (T_1, T_2) adversary $\mathcal{A}$ such that*

$$\mathsf{Adv}^{\mathsf{pre\text{-}2\text{-}CR\text{-}MD}}_{H_{n,\ell,n}}(\mathcal{A}) \geqslant \frac{(1 - 2/e)T_1T_2}{2^{3n/2+3}} .$$

The proof of this theorem is in the full version [13].

In the third term, ignoring constants and powers of n, the dominant factor is $\frac{T_1}{2^{5n/4}}$. We give an attack that achieves advantage of the order $\frac{T_1^2}{2^{5n/2}}$. While $\frac{T_1^2}{2^{5n/2}} \leqslant \frac{T_1}{2^{5n/4}}$ for $T_1 \leqslant 2^{5n/4}$, observe that both of them become one at $T_1 = 2^{5n/4}$. Formally, we prove the following theorem.

Theorem 9. *Let $T_1, T_2, n, \ell \in \mathbb{N}_{>0}$ such that n is a multiple of 2, and T_1 is a multiple of $2^{n/2+1}$. Let $H_{n,\ell,n}$ be the uniform distribution over $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^\ell, \{0,1\}^n)$. There exists a (T_1, T_2) adversary $\mathcal{A}$ such that*

$$\mathsf{Adv}^{\mathsf{pre\text{-}2\text{-}CR\text{-}MD}}_{H_{n,\ell,n}}(\mathcal{A}) \geqslant \frac{(1-2/e)T_1^2}{2^{5n/2+6}} .$$

The proof of this theorem is in the full version.

We now proceed to prove Theorem 7.

Proof. The proof of this theorem fixes the (T_1, T_2)-offline-online adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that maximizes $\mathsf{Adv}^{\mathsf{pre\text{-}2\text{-}CR\text{-}MD}}_{H_{n,\ell,n}}(T_1, T_2)$. We can treat $\mathcal{A}$ as deterministic by fixing its randomness that maximizes its advantage. Without loss of generality we can assume that $\mathcal{A}$ does not repeat any query across the offline and online phases because we have no restrictions on the memory of the adversary.

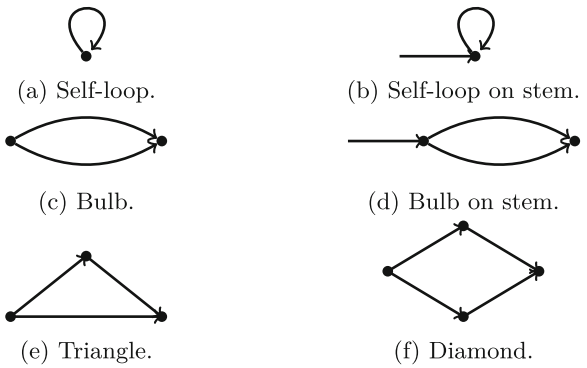

Fig. 6. The structure of the six different types of two-block MD collisions in the query graph. The nodes in the query graph are labelled with values in $\{0,1\}^n$, and there is an edge (a, a') labelled with M if the adversary made a query $h(a, M) = a'$. We omit the node and edge labels for simplicity.

We rewrite the collision-resistance game for two-block MD in game H in Fig. 7. Note that whenever $\mathcal{A}$ wins $\mathsf{G}^{\mathsf{2\text{-}PR\text{-}MD}}_{n,\ell}$, from our assumption that $\mathcal{A}$ makes all the queries needed to compute the MD evaluation of the messages it outputs, at least one of the following happens.

- $\mathcal{A}$ makes a query $h(a, M') = a$, meaning it has found a one-block message M' whose MD evaluation with salt a is a. This is sufficient for a two-block collision because for any $M'' \in \{0,1\}^\ell$, (M', M'') and M'' have the same MD evaluation with salt a.
- $\mathcal{A}$ makes queries $h(a, M') = z$ and $h(z, M'') = z$; this is a two-block collision because (M', M'') and M' have the same MD evaluation with salt a.
- $\mathcal{A}$ makes queries $h(a, M') = z$ and $h(a, M'') = z$ for $M' \neq M''$; this is a two-block collision because M', and M'' have the same MD evaluation with salt a.

- $\mathcal{A}$ makes queries $h(a, M') = z$, $h(y, M'') = z$ and $h(y, M''') = z$ for $M'' \neq M'''$; this is a two-block collision because (M', M'') and (M', M'') have the same MD evaluation with salt a.
- $\mathcal{A}$ makes queries $h(a, M') = y$, $h(y, M'') = z$ and $h(a, M''') = z$; this is a two-block collision because (M', M'') and M''' have the same MD evaluation with salt a.
- $\mathcal{A}$ makes queries $h(a, M') = y$, $h(y, M'') = z$, $h(a, M''') = y'$, and $h(y', M''') = z$ for $M' \neq M'''$ and $M'' \neq M''''$; this is a two-block collision because (M', M'') and (M''', M'''') have the same MD evaluation with salt a.

If any of these occur, win is set in H, meaning $\mathcal{A}$ wins the game. Therefore,

$$\mathsf{Adv}^{\mathsf{pre\text{-}2\text{-}CR\text{-}MD}}_{H_{n,\ell,n}}(\mathcal{A}) \leqslant \Pr\left[\mathsf{H}(\mathcal{A})\right] .$$

Game $\mathsf{H}(\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2))$	Oracle $\mathrm{H}(a, M)$
$h \leftarrow\!\!\$\ \mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^m, \{0,1\}^n)$	$\tau \leftarrow \tau \cup \{((a, M), h(a, M))\}$
win $\leftarrow$ false, $\tau \leftarrow []$, $a \leftarrow \perp$	If $\exists M' : ((a, M'), a) \in \tau$:
sl, sos, bulb, bos, tri, dia $\leftarrow$ false	win $\leftarrow$ true, sl $\leftarrow$ true
st $\leftarrow \mathcal{A}_1^{\mathrm{H}}$	If $\exists M', M'', z :$
$a \leftarrow\!\!\$\ \{0,1\}^n$	$((a, M'), z), ((z, M''), z) \in \tau$:
If $\exists M' : ((a, M'), a) \in \tau$:	win $\leftarrow$ true, sos $\leftarrow$ true
win $\leftarrow$ true, sl $\leftarrow$ true	If $\exists M' \neq M'', z :$
If $\exists M', M'', z :$	$((a, M'), z), ((a, M''), z) \in \tau$:
$((a, M'), z), ((z, M''), z) \in \tau$:	win $\leftarrow$ true, bulb $\leftarrow$ true
win $\leftarrow$ true, sos $\leftarrow$ true	If $\exists M', M'' \neq M''', y, z :$
If $\exists M' \neq M'', z :$	$((a, M'), y), ((y, M''), z) \in \tau$,
$((a, M'), z), ((a, M''), z) \in \tau$:	$((y, M'''), z) \in \tau$:
win $\leftarrow$ true, bulb $\leftarrow$ true	win $\leftarrow$ true, bos $\leftarrow$ true
If $\exists M', M'' \neq M''', y, z :$	If $\exists M', M'', M''', y, z :$
$((a, M'), y), ((y, M''), z) \in \tau$,	$((a, M'), y), ((y, M''), z) \in \tau$,
$((y, M'''), z) \in \tau$:	$((a, M'''), z) \in \tau$:
win $\leftarrow$ true, bos $\leftarrow$ true	win $\leftarrow$ true, tri $\leftarrow$ true
If $\exists M', M'', M''', y, z :$	If $\exists M' \neq M''', M'' \neq M'''', x, y, z :$
$((a, M'), y), ((y, M''), z) \in \tau$,	$((a, M'), x), ((a, M''), y) \in \tau$,
$((a, M'''), z) \in \tau$:	$((x, M'''), z), ((y, M''''), z) \in \tau$:
win $\leftarrow$ true, tri $\leftarrow$ true	win $\leftarrow$ true, dia $\leftarrow$ true
If $\exists M' \neq M''', M'' \neq M'''', x, y, z :$	$\mathcal{A}_2^{\mathrm{H}}(\mathsf{st}, a)$
$((a, M'), x), ((a, M''), y) \in \tau$,	Return $h(a, M)$
$((x, M'''), z), ((y, M''''), z) \in \tau$:	
win $\leftarrow$ true, dia $\leftarrow$ true	
$\mathcal{A}_2^{\mathrm{H}}(\mathsf{st}, a)$	
Return win	

Fig. 7. H using in the analysis of collision-resistance of two-block MD against offline-online adversaries. The events introduced in this game are marked in red. (Color figure online)

The game H defines events $\mathsf{sl}, \mathsf{sos}, \mathsf{bulb}, \mathsf{bos}, \mathsf{tri}, \mathsf{dia}$. We name the events this way because of the following alternative view of MD collisions via the query graph of $\mathcal{A}$: the nodes of the query graph are labelled with strings from $\{0,1\}^n$, and whenever $\mathcal{A}$ makes a query $h(a, M) = a'$, an edge (a, a') labelled M is added to the graph. Finding a two block collision can be viewed as finding one of the following structures in the query graph: self-loop, self-loop on stem, bulb, bulb-on-stem, triangle, and diamond; Fig. 6 shows these structures.

It follows from inspection that in game H, win is set only if one of these event among $\{\mathsf{sl}, \mathsf{sos}, \mathsf{bulb}, \mathsf{bos}, \mathsf{tri}, \mathsf{dia}\}$ happen. Therefore, using the union bound we have that

$$\begin{aligned}\Pr\left[\mathsf{H}(\mathcal{A})\right] &= \Pr\left[\mathcal{A} \text{ sets } \mathsf{win}\right] \\ &\leqslant \sum_{\mathsf{event} \in \{\mathsf{sl},\mathsf{sos},\mathsf{bulb},\mathsf{bos},\mathsf{tri},\mathsf{dia}\}} \Pr\left[\mathsf{event}\right] \end{aligned} \tag{12}$$

Our proof is divided into these following lemmas each of which upper bound the probability of these events.

Lemma 4

$$\Pr\left[\mathsf{sl}\right] \leqslant \frac{1}{2^n} + \frac{n}{2^n} + \frac{2eT_1}{2^{2n}} + \frac{T_2}{2^n} .$$

Lemma 5

$$\Pr\left[\mathsf{sos}\right] \leqslant \frac{T_2 + 3 + nT_2 + n^2}{2^n} + \frac{2eT_1T_2 + 6enT_1}{2^{2n}} + \frac{8e^2T_1^2}{2^{3n}} .$$

Lemma 6

$$\Pr\left[\mathsf{bulb}\right] \leqslant \frac{\binom{T_2}{2}}{2^n} + \frac{T_2T_1}{2^{2n}} + \frac{eT_1}{2^{3n/2}} + \frac{n}{2^{n+1}} + \frac{1}{2^n} .$$

Lemma 7

$$\begin{aligned}\Pr\left[\mathsf{bos}\right] \leqslant &\frac{\binom{T_2}{2} + nT_2/2 + n^2/2 + 4}{2^n} + \frac{eT_1T_2 + enT_1}{2^{3n/2}} + \frac{nT_1T_2 + T_1T_2^2 + 2enT_1}{2^{2n}} \\ &+ \frac{4e^2T_1^2}{2^{5n/2}} + \frac{4eT_1^2T_2}{2^{3n}} .\end{aligned}$$

Lemma 8

$$\begin{aligned}\Pr\left[\mathsf{tri}\right] \leqslant &\frac{\binom{T_2}{2} + 18n + 8}{2^n} + \frac{3(24e)^{1/2}T_1 + 3(8en)^{1/2}T_1}{2^{3n/2}} + \frac{9(2)^{1/3}eT_1^{4/3}}{2^{5n/3}} \\ &+ \frac{2nT_1T_2 + T_1T_2 + T_2^2T_1 + 12e(2)^{1/2}T_1^{3/2}}{2^{2n}} + \frac{8eT_1^2T_2 + 2T_1T_2}{2^{3n}} .\end{aligned}$$

Lemma 9

$$\begin{aligned}\Pr[\mathsf{dia}] \leqslant & \frac{\binom{T_2}{2} + 30n + 16}{2^n} + \frac{4eT_1^2T_2 + 2T_1T_2 + 4eT_1^2T_2^2}{2^{3n}} \\ & + \frac{nT_1T_2 + T_2^2T_1 + n^2T_1T_2 + nT_1T_2^2}{2^{2n}} + \frac{4eT_1^3T_2}{2^{4n}} \\ & + \frac{40(en)^{1/3}T_1}{2^{4n/3}} + \frac{(8e)^{1/2}nT_1 + 10(2e)^{1/2}nT_1}{2^{3n/2}} \\ & + \frac{240e^{2/3}T_1^2}{2^{7n/3}} + \frac{8(2)^{1/2}e^{3/2}T_1^2 + 40(2e)^{1/2}T_1^2}{2^{5n/2}} \ .\end{aligned}$$

Plugging all probability upper bounds into (12), rearranging terms and simplifying, we get the required bound. We show these calculations in the full version [13].

We prove Lemmas 4 to 6 in Sects. 5.3 to 5.5 respectively. Due to lack of space we defer the proofs of Lemmas 7 to 9 to the full version [13]. □

5.3 Proof of Lemma 4

Proof (Lemma 4). We define the two following events.

1. $\mathsf{sl}_{\mathsf{off}}$: $\mathcal{A}_1$ made a query $h(a, M) = a$
2. $\mathsf{sl}_{\mathsf{on}}$: $\mathcal{A}_2$ made a query $h(a, M) = a$

Notice that sl happens only if at least one of $\mathsf{sl}_{\mathsf{off}}$ or $\mathsf{sl}_{\mathsf{on}}$ happens. Therefore, we have that

$$\Pr[\mathsf{sl}] \leqslant \Pr[\mathsf{sl}_{\mathsf{off}}] + \Pr[\mathsf{sl}_{\mathsf{on}}] \ . \tag{13}$$

We first prove an upper bound on $\Pr[\mathsf{sl}_{\mathsf{on}}]$. Note that, for every query $h(a, M)$ made by $\mathcal{A}_2$, the probability that its answer is a is $1/2^n$. Therefore, using a union bound over all the queries of $\mathcal{A}_2$, we have that

$$\Pr[\mathsf{sl}_{\mathsf{on}}] \leqslant T_2/2^n \ .$$

To prove an upper bound on $\Pr[\mathsf{sl}_{\mathsf{off}}]$, we define the following event $\mathsf{off\text{-}sl}\text{-}k$: $\mathcal{A}_1$ makes at least k different queries such that the input salt of the query is the answer of the query. Using total probability, we have that for any k

$$\Pr[\mathsf{sl}_{\mathsf{off}}] \leqslant \Pr\left[\mathsf{sl}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}sl}\text{-}k\right] + \Pr[\mathsf{off\text{-}sl}\text{-}k] + \frac{k}{2^n} \ . \tag{14}$$

Note that, if the adversary $\mathcal{A}_1$ makes at most k different queries, such that the input salt of the query is the answer, $\mathsf{sl}_{\mathsf{off}}$ happens only if the salt a that is sampled uniformly at random is same as the salt for one of those at most k queries. Therefore, $\Pr\left[\mathsf{sl}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}sl}\text{-}k\right] \leqslant k/2^n$.

We upper bound $\Pr[\mathsf{off\text{-}sl}\text{-}k]$ as follows. Let B_j be the indicator random variable that indicates whether the j-th query of $\mathcal{A}_1$ is such that its answer

is same as its input salt. Since $\mathcal{A}_1$ does not repeat queries, all the B_j's are independent, and $\Pr[B_j] = 1/2^n$. From the definition of B_j's, it follows that

$$\Pr[\mathsf{off\text{-}sl}\text{-}k] = \Pr\left[\sum_{j=1}^{T_1} B_j \geqslant k\right] .$$

We rewrite the term on the right as

$$\Pr\left[\sum_{j=1}^{T_1} B_j \geqslant k\right] = \Pr[\exists S \subseteq [T_1], |S| = k : \forall j \in S, B_j = 1] .$$

Using a union bound over all subsets of T_1 of size k, we have

$$\Pr[\exists S \subseteq [T_1], |S| = k : \forall j \in S, B_j = 1] \leqslant \sum_{S \subseteq [T_1], |S|=k} \Pr[\forall j \in S, B_j = 1] .$$

Since there are $\binom{T_1}{k}$ subsets of $[T_1]$ of size k, and all the B_j's are independent and $\Pr[B_j = 1] = 1/2^n$, we have that

$$\Pr[\exists S \subseteq [T_1], |S| = k : \forall j \in S, B_j = 1] \leqslant \frac{\binom{T_1}{k}}{2^{nk}} .$$

Therefore

$$\Pr[\mathsf{off\text{-}sl}\text{-}k] \leqslant \frac{\binom{T_1}{k}}{2^{nk}} \leqslant \left(\frac{eT_1}{k2^n}\right)^k .$$

Plugging this in (14), we have that for any k,

$$\Pr[\mathsf{sl}_{\mathsf{off}}] \leqslant \left(\frac{eT_1}{k2^n}\right)^k + \frac{k}{2^n} .$$

We let $k = \max\left(n, \frac{2eT_1}{2^n}\right)$. If $n \leqslant \frac{2eT_1}{2^n}$, we have that $k = \frac{2eT_1}{2^n} \geqslant n$. Therefore,

$$\frac{\binom{T_1}{k}}{2^{nk}} \leqslant \left(\frac{eT_1}{k2^n}\right)^k \leqslant \left(\frac{1}{2}\right)^k \leqslant \frac{1}{2^n} .$$

Otherwise if $n > \frac{2eT_1}{2^n}$, we have that $k = n > \frac{2eT_1}{2^n}$. Therefore,

$$\frac{\binom{T_1}{k}}{2^{nk}} \leqslant \left(\frac{eT_1}{k2^n}\right)^k \leqslant \left(\frac{1}{2}\right)^k = \frac{1}{2^n} .$$

Hence,

$$\Pr[\mathsf{sl}_{\mathsf{off}}] \leqslant \frac{1}{2^n} + \frac{n}{2^n} + \frac{2eT_1}{2^{2n}} .$$

Plugging this back into (13) gives us

$$\Pr[\mathsf{sl}] \leqslant \frac{1}{2^n} + \frac{n}{2^n} + \frac{2eT_1}{2^{2n}} + \frac{T_2}{2^n} .$$

□

5.4 Proof of Lemma 5

Proof (Lemma 5). We first define the three following events.

1. $\mathsf{sos}_{\mathsf{off}}$: $\mathcal{A}_1$ made queries $h(a, M') = z$ and $h(z, M'') = z$
2. $\mathsf{sos}_{\mathsf{offon}}$: $\mathcal{A}_1$ made a query $h(y, M'') = y$ and $\mathcal{A}_2$ made a query $h(a, M') = y$
3. $\mathsf{sos}_{\mathsf{on}}$: $\mathcal{A}_2$ made a query $h(z, M') = z$, i.e., a query whose answer is the same as its input salt

From inspection one can verify that sos happens only if at least one of $\mathsf{sos}_{\mathsf{off}}$, $\mathsf{sos}_{\mathsf{on}}$, $\mathsf{sos}_{\mathsf{offon}}$ happen. It follows that

$$\Pr[\mathsf{sos}] \leqslant \Pr[\mathsf{sos}_{\mathsf{off}}] + \Pr[\mathsf{sos}_{\mathsf{on}}] + \Pr[\mathsf{sos}_{\mathsf{offon}}] \ . \tag{15}$$

We first upper bound $\Pr[\mathsf{sos}_{\mathsf{on}}]$. Note that for every query $h(a, M)$ made by $\mathcal{A}_2$, the probability that its answer is a is $1/2^n$. Therefore, using a union bound over all the queries of $\mathcal{A}_2$, we have that

$$\Pr[\mathsf{sos}_{\mathsf{on}}] \leqslant T_2/2^n \ . \tag{16}$$

We next upper bound $\Pr[\mathsf{sos}_{\mathsf{offon}}]$. Recall that the event $\mathsf{off\text{-}sl}\text{-}k$ defined in the proof of Lemma 4: $\mathcal{A}_1$ makes at least k different queries such that the input salt of the query is the answer. We have that

$$\Pr[\mathsf{sos}_{\mathsf{offon}}] \leqslant \Pr\left[\mathsf{sos}_{\mathsf{offon}} \,\middle|\, \neg\mathsf{off\text{-}sl}\text{-}k\right] + \Pr[\mathsf{off\text{-}sl}\text{-}k] \ .$$

In this case, $\mathsf{sos}_{\mathsf{offon}}$ happens only if $\mathcal{A}_2$ makes a query whose answer is the input salt of one of at most k such queries. Therefore, we have that for any k,

$$\Pr[\mathsf{sos}_{\mathsf{offon}}] \leqslant \Pr[\mathsf{off\text{-}sl}\text{-}k] + \frac{kT_2}{2^n} \ .$$

As seen in the proof of Lemma 4, setting $k = \max\left(n, \frac{2eT_1}{2^n}\right)$ makes $\Pr[\mathsf{off\text{-}sl}\text{-}k] \leqslant 1/2^n$. Therefore, by setting this value of k, we have that

$$\Pr[\mathsf{sos}_{\mathsf{offon}}] \leqslant \frac{nT_2}{2^n} + \frac{2eT_1T_2}{2^{2n}} + \frac{1}{2^n} \ . \tag{17}$$

We finally upper bound $\Pr[\mathsf{sos}_{\mathsf{off}}]$. For this we recall $(m+1)\text{-}\mathsf{col}$ as the event that we defined in the proof of Theorem 4. We say that $(m+1)\text{-}\mathsf{col}$ happens if the $\mathcal{A}_1$ has made $m+1$ distinct random oracle queries that all have the same answer. Using total probability, we have that for any k, m

$$\begin{aligned}\Pr[\mathsf{sos}_{\mathsf{off}}] &\leqslant \Pr\left[\mathsf{sos}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}sl}\text{-}k \wedge \neg(m+1)\text{-}\mathsf{col}\right] + \Pr[\mathsf{off\text{-}sl}\text{-}k \vee (m+1)\text{-}\mathsf{col}] \\ &\leqslant \Pr\left[\mathsf{sos}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}sl}\text{-}k \wedge \neg(m+1)\text{-}\mathsf{col}\right] + \Pr[\mathsf{off\text{-}sl}\text{-}k] + \Pr[(m+1)\text{-}\mathsf{col}] \ .\end{aligned} \tag{18}$$

We claim that

$$\Pr\left[\mathsf{sos}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}sl}\text{-}k \wedge \neg(m+1)\text{-}\mathsf{col}\right] \leqslant \frac{mk}{2^n} \ .$$

This is because if off-sl-k and $(m+1)$-col do not happen, there can be at most $k \cdot m$ salts a that satisfy that $\mathcal{A}_1$ makes a query $h(a, M') = z$ and $h(z, M'') = z$. The probability that the salt a that is sampled uniformly at random is among one of those at most $k \cdot m$ salts is at most $\frac{mk}{2^n}$.

From our calculations in the proof of Theorem 4, we have that for $m = \max\left(n, \frac{4eT_1}{2^n}\right)$, $\Pr\left[(m+1)\text{-col}\right] \leqslant 1/2^n$. We also know that for $k = \max\left(n, \frac{2eT_1}{2^n}\right)$, $\Pr\left[\text{off-sl-}k\right] \leqslant 1/2^n$. We set m, k to these values and obtain from (18) that

$$\Pr\left[\mathsf{sos}_{\mathsf{off}}\right] \leqslant \frac{2}{2^n} + \frac{n^2}{2^n} + \frac{6enT_1}{2^{2n}} + \frac{8e^2T_1^2}{2^{3n}} \ . \tag{19}$$

This is because for $m = \max\left(n, \frac{4eT_1}{2^n}\right)$, $k = \max\left(n, \frac{2eT_1}{2^n}\right)$,

$$k \cdot m \leqslant n^2 + 6enT_1/2^n + 8e^2T_1^2/2^{2n} \ .$$

Plugging (16), (17) and (19) into (15), we get that

$$\Pr\left[\mathsf{sos}\right] \leqslant \frac{T_2 + 3 + nT_2 + n^2}{2^n} + \frac{2eT_1T_2 + 6enT_1}{2^{2n}} + \frac{8e^2T_1^2}{2^{3n}} \ .$$

□

5.5 Proof of Lemma 6

Proof. We define the three following events.

1. $\mathsf{bulb}_{\mathsf{off}}$: $\mathcal{A}_1$ made queries $h(a, M) = y$, $h(a, M') = y$ for some $M \neq M'$ and y
2. $\mathsf{bulb}_{\mathsf{offon}}$: $\mathcal{A}_1$ made queries $h(a, M) = y$, and $\mathcal{A}_2$ made a query with answer y for some M, y
3. $\mathsf{bulb}_{\mathsf{on}}$: $\mathcal{A}_2$ made queries $h(a, M) = y$, $h(a, M') = y$ for some $M \neq M'$ and y

Observe that bulb happens only if at least one of $\mathsf{bulb}_{\mathsf{off}}$, $\mathsf{bulb}_{\mathsf{offon}}$, $\mathsf{bulb}_{\mathsf{on}}$ happen. Therefore

$$\Pr\left[\mathsf{bulb}\right] \leqslant \Pr\left[\mathsf{bulb}_{\mathsf{off}}\right] + \Pr\left[\mathsf{bulb}_{\mathsf{offon}}\right] + \Pr\left[\mathsf{bulb}_{\mathsf{on}}\right] \ . \tag{20}$$

The rest of the proof consists of upper bounding these probabilities one by one. We begin with $\Pr\left[\mathsf{bulb}_{\mathsf{on}}\right]$. Observe that $\mathsf{bulb}_{\mathsf{on}}$ happens only if $\mathcal{A}_2$ makes two queries that have the same answer. The probability of any two queries of $\mathcal{A}$ having the same answer is $1/2^n$. Using a union bound over all pairs of queries of $\mathcal{A}_2$, we have that

$$\Pr\left[\mathsf{bulb}_{\mathsf{on}}\right] \leqslant \frac{\binom{T_2}{2}}{2^n} \ . \tag{21}$$

We next upper bound $\Pr[\mathsf{bulb}_{\mathsf{offon}}]$. Let Q_a be the random variable denoting the number of queries $\mathcal{A}_1$ makes with salt a. Using total probability

$$\begin{aligned}\Pr[\mathsf{bulb}_{\mathsf{offon}}] &= \sum_{i=1}^{T_1} \Pr[Q_a = k] \Pr\left[\mathsf{bulb}_{\mathsf{offon}} \,\middle|\, Q_a = k\right] \\ &= \sum_{i=1}^{T_1} \Pr[Q_a = k] \cdot \frac{kT_2}{2^n} \\ &= \frac{T_2}{2^n} \cdot \mathsf{E}[Q_a] = \frac{T_1 T_2}{2^{2n}} \,. \end{aligned} \tag{22}$$

The second equality above follows because if $\mathcal{A}_1$ makes k queries with salt a, the probability that $\mathsf{bulb}_{\mathsf{offon}}$ happens is at most $kT_2/2^n$ using a union bound over all queries of $\mathcal{A}_2$. The final equality uses the fact that $\mathsf{E}[Q_a] = T_1/2^n$, because $\mathcal{A}_1$ makes T_1 queries and a is sampled uniformly at random.

Finally, we upper bound $\Pr[\mathsf{bulb}_{\mathsf{off}}]$. We define an event $\mathsf{off\text{-}bulbs}$-$k$ as follows: there is a set of at least k distinct salts $a_1, \ldots, a_k$ such that for each a_i, $\mathcal{A}_1$ has made a pair of queries $h(a_i, M_i) = z$ and $h(a_i, M_i') = z$ for $M_i \neq M_i'$.

We have that for any k,

$$\Pr[\mathsf{bulb}_{\mathsf{off}}] \leqslant \Pr\left[\mathsf{bulb}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}bulbs}\text{-}k\right] + \Pr[\mathsf{off\text{-}bulbs}\text{-}k] \,. \tag{23}$$

Since $\mathsf{bulb}_{\mathsf{off}}$ happens if for the salt a that is chosen uniformly at random from $\{0,1\}^n$, $\mathcal{A}_1$ had queried $h(a, M), h(a, M')$ that have the same answer, we have that $\Pr\left[\mathsf{bulb}_{\mathsf{off}} \,\middle|\, \neg\mathsf{off\text{-}bulbs}\text{-}k\right] \leqslant k/2^n$. We upper bound $\Pr[\mathsf{off\text{-}bulbs}\text{-}k]$ using a compression argument.

Note that the event $\mathsf{off\text{-}bulbs}$-$k$ is similar to the event $\mathsf{off\text{-}oneblk}$-$k$ we defined in the proof of Theorem 4, the only difference being the salt length was s there and is n here. However, $\Pr[\mathsf{off\text{-}oneblk}\text{-}k]$ did not depend on s, hence we can prove the same bound for $\Pr[\mathsf{off\text{-}bulbs}\text{-}k]$. We showed in that proof that for $k = \max\left(\frac{eT_1}{2^{n/2}}, n/2\right)$, $\Pr[\mathsf{off\text{-}oneblk}\text{-}k] \leqslant 1/2^n$.

Hence from (23) we have

$$\Pr[\mathsf{bulb}_{\mathsf{off}}] \leqslant \frac{eT_1}{2^{3n/2}} + \frac{n}{2^{n+1}} + \frac{1}{2^n} \,. \tag{24}$$

Plugging (21), (22) and (24) into (20) gives us that

$$\Pr[\mathsf{bulb}] \leqslant \frac{\binom{T_2}{2}}{2^n} + \frac{T_2 T_1}{2^{2n}} + \frac{eT_1}{2^{3n/2}} + \frac{n}{2^{n+1}} + \frac{1}{2^n} \,.$$

□

Acknowledgements. This research was partially supported by NSF grants CNS-2026774, CNS-2154174, a JP Morgan Faculty Award, a CISCO Faculty Award, and a gift from Microsoft.

References

1. Adrian, D., et al.: Imperfect forward secrecy: how Diffie-Hellman fails in practice. In: Ray, I., Li, N., Kruegel, C. (eds.) ACM CCS 2015, pp. 5–17. ACM Press, October 2015
2. Akshima, Cash, D., Drucker, A., Wee, H.: Time-space tradeoffs and short collisions in Merkle-Damgård hash functions. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part I. LNCS, vol. 12170, pp. 157–186. Springer, Heidelberg (2020). https://doi.org/10.1007/978-3-030-56784-2_6
3. Akshima, Guo, S., Liu, Q.: Time-space lower bounds for finding collisions in Merkle-Damgård hash functions. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part III. LNCS, vol. 13509, pp. 192–221. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-15982-4_7
4. Bernstein, D.J., Lange, T.: Non-uniform cracks in the concrete: the power of free precomputation. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 321–340. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42045-0_17
5. Coretti, S., Dodis, Y., Guo, S.: Non-uniform bounds in the random-permutation, ideal-cipher, and generic-group models. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 693–721. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_23
6. Coretti, S., Dodis, Y., Guo, S., Steinberger, J.: Random oracles and non-uniformity. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 227–258. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_9
7. Corrigan-Gibbs, H., Kogan, D.: The discrete-logarithm problem with preprocessing. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 415–447. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_14
8. Damgård, I.B.: A design principle for hash functions. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 416–427. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_39
9. De, A., Trevisan, L., Tulsiani, M.: Time space tradeoffs for attacks against one-way functions and PRGs. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 649–665. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_35
10. Dodis, Y., Guo, S., Katz, J.: Fixing cracks in the concrete: random oracles with auxiliary input, revisited. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part II. LNCS, vol. 10211, pp. 473–495. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56614-6_16
11. Gaži, P., Pietrzak, K., Rybár, M.: The exact PRF-security of NMAC and HMAC. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 113–130. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_7
12. Ghoshal, A., Komargodski, I.: On time-space tradeoffs for bounded-length collisions in Merkle-Damgård hashing. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part III. LNCS, vol. 13509, pp. 161–191. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-662-44371-2_7
13. Ghoshal, A., Tessaro, S.: The query-complexity of preprocessing attacks. Cryptology ePrint Archive, Paper 2023/856 (2023). https://eprint.iacr.org/2023/856

14. Hellman, M.E.: A cryptanalytic time-memory trade-off. IEEE Trans. Inf. Theory **26**(4), 401–406 (1980)
15. Jaeger, J., Tessaro, S.: Expected-time cryptography: generic techniques and applications to concrete soundness. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part III. LNCS, vol. 12552, pp. 414–443. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64381-2_15
16. Koblitz, N., Menezes, A.: Another look at HMAC. Cryptology ePrint Archive, Report 2012/074 (2012). https://eprint.iacr.org/2012/074
17. Merkle, R.C.: A fast software one-way hash function. J. Cryptol. **3**(1), 43–58 (1990)
18. Oechslin, P.: Making a faster cryptanalytic time-memory trade-off. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 617–630. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_36
19. Rogaway, P.: Formalizing human ignorance. In: Nguyen, P.Q. (ed.) VIETCRYPT 2006. LNCS, vol. 4341, pp. 211–228. Springer, Heidelberg (2006). https://doi.org/10.1007/11958239_14
20. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_18
21. Unruh, D.: Random oracles and auxiliary input. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 205–223. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74143-5_12