Offline-Online Indifferentiability of Cryptographic Systems

Ashrujit Ghoshal* Ilan Komargodski† Gil Segev‡

Abstract

The indifferentiability framework has become a standard methodology that enables us to study the security of cryptographic constructions in idealized models of computation. Unfortunately, while indifferentiability provides strong guarantees whenever the security of a construction is captured by a "single-stage" security game, it may generally provide no meaningful guarantees when the security is captured by a "multi-stage" one. In particular, the indifferentiability framework does not capture offline-online games, where the adversary can perform an extensive offline computation to later speed up the online phase. Such security games are extremely common, both in practice and in theory. Over the past decade, there has been numerous attempts to meaningfully extend the indifferentiability framework to offline-online games, however, they all ultimately met with little success.

In this work, our contribution is threefold. First, we propose an extension of the classical indifferentiability framework, we refer to as *offline-online-indifferentiability*, that applies in the context of attackers with an expensive offline phase (à la Ghoshal and Tessaro, CRYPTO '23). Second, we show that our notion lends itself to a natural and meaningful composition theorem for offline-online security games. Lastly, as our main technical contribution, we analyze the offline-online-indifferentiability of two classical variants of the Merkle-Damgård hashing mechanism, one where the key is fed only to the first block in the chain and the other where the key is fed to each block in the chain. For both constructions, we prove a *tight* bound on their offline-online-indifferentiability (i.e., an upper bound and an attack that matches it). Notably, our bound for the second variant shows that the construction satisfies *optimal* offline-online-indifferentiability.

1 Introduction

A cryptographic hash function is an efficient mapping between arbitrary length inputs to a fixed length digest. It is one of the most fundamental tools in applications of cryptography, underlying numerous widely used cryptosystems. For example, it facilitates the hash-and-sign paradigm, proofs-of-work for blockchains, and many more. While it is empirically believed that concrete

*Carnegie Mellon University. Email: aghoshal@andrew.cmu.edu. Supported in part by NSF awards 2128519 and 2044679, an ONR grant, a Packard Fellowship and a DARPA SIEVE grant under a subcontract from SRI.

†School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem, 9190401, Israel and NTT Research. Email: ilank@cs.huji.ac.il. Supported in part by an Alon Young Faculty Fellowship, by a grant from the Israel Science Foundation (ISF Grant No. 1774/20), by a grant from the US-Israel Binational Science Foundation and the US National Science Foundation (BSF-NSF Grant No. 2020643), and by the European Union (ERC, SCALE,101162665). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

‡School of Computer Science and Engineering, Hebrew University of Jerusalem, Jerusalem 91904, Israel. Email: segev@cs.huji.ac.il. Supported by the Israel Science Foundation (Grant No. 1336/22) and by the European Union (ERC, FTRC, 101043243). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

cryptographic hash functions such as SHA-2 and SHA-3 satisfy various useful security properties, formalizing this seems to be currently out of reach. Thus, in the context of provable security, cryptographic hash functions are usually modeled as completely random functions, i.e., in the Random Oracle Model (ROM) [BR93]. This allows us to analyze specific properties and argue about the concrete security of systems that use them.

Since with the work of Hellman [Hel80] and Fiat and Naor [FN99], and even more so in recent years, there have been significant efforts to extend various security guarantees of hash functions beyond the uniform setting (for example, [Unr07, DGK17, CDG18, CDGS18, ACDW20, GLLZ21, GLLZ21, GK23, AGL22, FGK22, FGK23]). Indeed, the uniform model assumes, in some sense, that the adversary is "fixed" before the hash function and so it clearly does not apply to many real-world applications. Thus, focus has shifted to analyzing analogous keyed constructions in a non-uniform setting, where we consider adversaries that have gone through an expensive preprocessing stage. Beyond being more realistic as an idea, it is widely accepted in the cryptography community that non-uniformity is the "right" modeling of attackers, despite being overly conservative and including potentially unrealistic attackers.

Unfortunately, we do not have sufficiently general systematic methods for analyzing cryptographic constructions in models that allow preprocessing. Indeed, all the above-mentioned recent works [DGK17, CDG18, CDGS18, ACDW20, GK23, AGL22, FGK22, FGK23] consider an attacker with such a preprocessing phase but eventually analyze only *a specific security property* of a given (keyed) construction. For instance, [ACDW20, GK23, AGL22, FGK22, FGK23] *only consider collision resistance*. Unfortunately, these proofs are very complex and do not easily port to other security properties (like one-wayness, pseudorandomness, unpredictability, and more). One could ask whether it is possible to carry out a single analysis that will capture a (wide) class of security properties simultaneously. Such a result would not only give a better understanding of the security or weakness of a given hash function construction, but as we shall see, it is also inherent if we want to apply composition and use these hash functions in larger systems.

The latter has been formalized in a number of frameworks, notably in the UC framework of Canetti [Can01, Can20], the reactive systems framework of Backes, Pfitzmann and Waidner [BPW04], and the indifferentiability framework of Maurer, Renner, and Holenstein [MRH04]. The latter is by now a standard methodology to study the soundness of cryptographic constructions, particularly symmetric ones such as hash functions [CDMP05, BDPA08, RSS11] and block-ciphers [CPS08, HKT11, ABD+13, DSSL16], in idealized models of computation. The indifferentiability framework is extremely popular because it allows to argue the soundness of various constructions that rely on compressing random functions. For concreteness, imagine a suggested hash function construction $C^{\pi}\colon \{0,1\}^{4n} \to \{0,1\}^{n}$ using an ideal random compressing function $\pi\colon \{0,1\}^{2n} \to \{0,1\}^{n}$. Intuitively, if C^{π} is indifferentiable from a random oracle $\Pi\colon \{0,1\}^{4n} \to \{0,1\}^{n}$, then any cryptosystem that uses C^{π} can be analyzed when C^{π} is replaced by Π, which is potentially much easier.

Unfortunately, as Riestenpart, Shacham, and Shrimpton [RSS11] pointed out, [MRH04]'s definition of indifferentiability does not generically capture security games where offline preprocessing is allowed. In [RSS11]'s language, the framework of [MRH04] applies only to *single-stage* games, but does not necessarily apply to *two-or-more-stage* games. Indeed, [RSS11] come up with concrete (and natural) multi-stage games where the indifferentiability composition theorem completely fails. Notably, security games with an offline phase are inherently two-stage: first, the adversary gets to see and somehow interact with the ideal object to produce an "advice", and only later the challenge is presented to the adversary and it needs to break the system in some way. Riestenpart et al. [RSS11] go further and suggest a stronger definition of indifferentiability, called *reset-indifferentiability*, that allows for composition even for multi-stage games. However, this no-

tion is probably too strong because no variable-input-length hash function construction can fulfill it [RSS11, LAMP12, DGHM13]. The challenge of obtaining a notion that is sufficient for games that include an offline phase and achievable at the same time was tackled by several works (e.g., [MPS12, DGHM13, Mit14, DFMT20]); nevertheless, they all ultimately fall short of being generally applicable to offline-online games in a meaningful manner. We refer to Section 1.2 for further details on the above attempts and how they compare to ours.

Recall, the indifferentiability of a construction C^π from an ideal object Π allows us to analyze the security of C^π in some game by replacing it with the ideal object Π. In the offline-online setting, analyzing an ideal object is extremely hard without reducing the problem to the analysis of an online-only attacker against the ideal object. Indeed, the few cases that managed to analyze ideal primitives (e.g., random oracles) without such a reduction in the offline-online setting are extremely complicated [DGK17, GT23]. So, ideally, we would like to reduce the task of analyzing a construction C^π with respect to some offline-online game to the (much easier) task of analyzing an ideal object Π but with respect to an online-only game. This, overall, will allow us to have relatively simple and widely applicable analysis, at the expense of looser bounds.

This work. Our contribution is threefold:

- We propose an extension of the classical indifferentiability framework, we refer to as *offline-online-indifferentiability*, that captures the "distance" between a given (keyed) construction and an ideal primitive in the context of attackers with an expensive offline phase. Concretely, our framework captures the offline-online tradeoffs model of Ghoshal and Tessaro [GT23], wherein both the offline and the online phases have a bound on their query complexity (unlike the time-space tradeoffs model in which there is a bound on the output size of the offline phase).

- We show that offline-online-indifferentiability lends itself to a natural and meaningful composition theorem, capturing security games where attackers could have an expensive offline phase. Remarkably, our composition theorem allows us to transform the analysis of an adversary with an offline phase to one that is online-only.

- Lastly, we prove that several popular (keyed) hash function constructions satisfy our notion of offline-online-indifferentiability. We complement our offline-online-indifferentiability bounds with matching attacks.

1.1 Our Results in Detail

Indifferentiability formalizes a set of necessary and sufficient conditions for the construction C^π to securely replace its ideal counterpart Π in a wide range of environments. Roughly this is obtained by requiring a simulator S for which the systems (C^π, π) and (Π, S^Π) are indistinguishable. The composition theorem proved by [MRH04] states that, if C^π is indifferentiable from Π, then C^π can securely replace Π in arbitrary single-stage games.

The most common model for analyzing security in the presence of an expensive offline phase is the so called (S, T)-model often referred to as "time-space tradeoffs". In this model the offline phase which has access to π is unbounded in queries or computational complexity, but is bounded in its output size, denoted S. The online phase gets the S-sized output of the offline phase as well as a challenge and needs to "break" the system while issuing at most T queries.

Considering such a setting in the context of indifferentiability is extremely challenging: imagine that the offline adversary, based on π, generates some (structured) advice which the online phase later utilizes. For indifferentiability, we would need to simulate the answers seen by the online

attacker such that they are consistent with respect to the advice. It is not clear that a computationally bounded simulator can simulate such a distribution, even given the advice. This is mainly due to the fact that, in the (S, T)-model, the offline attacker has unrestricted access to the ideal object and so the S-sized output of this phase may be an *arbitrary* function of the corresponding responses – with which a *query-bounded* simulator must be consistent. See Section 1.2 for a discussion on prior attempts.

Offline-online time tradeoffs. We consider a slightly different yet naturally motivated security game, called the (T_1, T_2)-model or "offline-online time tradeoffs," where the offline phase is bounded in queries rather than output size, a model that was recently put forward by Ghoshal and Tessaro [GT23]. In this model, the goal is to understand the complexity of attacks that depend on both offline and online *time*, where both the offline and online complexities are measured by the number of queries to the given primitive. We emphasize that in this model, there is no limitation on the amount of information conveyed from the offline phase to the online phase, but rather the online phase gets "free" access to the query-answer transcript sequence of the offline phase. The motivation behind the introduction of this model was capturing the actual *feasibility* of the offline phase of the offline-online attacks, because for many of the attacks in the time-space tradeoff model the offline phase is nearly infeasible. In this model, [GT23] study the security of salted hash functions and most notably prove bounds on the pre-image and collision-resistance security of the (2-block) Merkle-Damgård construction, the most popular technique for keyed domain extension for hash functions.

We call an attacker in our model a (T_1, T_2)-attacker to indicate that the offline phase is allowed to perform T_1 queries and the online phase is allowed to perform T_2 queries. We say that a keyed construction $\{C^\pi_\mathsf{k}\}_\mathsf{k}$ is (T_1, T_2)-*offline-online-indifferentiable* from an ideal counterpart Π if there is a simulator S such that for any (T_1, T_2)-adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, it cannot distinguish between $b = 0$ and $b = 1$ in the following game:

1. Generate the advice string τ, where
   - **real world** $(b = 0)$: $\tau \leftarrow \mathcal{A}_1^\pi$;
   - **ideal world** $(b = 1)$: $\tau \leftarrow \mathcal{A}_1^S$.
2. Sample the challenge $\mathsf{k} \leftarrow \mathsf{Samp}(1^n)$, where $\mathsf{Samp}(\cdot)$ is a sampler.
3. Let $\mathcal{A}_2$ attack the scheme, where
   - **real world** $(b = 0)$: Output $\mathcal{A}_2^{C^\pi_\mathsf{k}, \pi}(\tau, \mathsf{k})$;
   - **ideal world** $(b = 1)$: Output $\mathcal{A}_2^{\Pi, S^\Pi(\tau, \mathsf{k})}(\tau, \mathsf{k})$.

The classical indifferentiability definition [MRH04] is a special case of this definition with τ and k being empty (and so $\mathcal{A}_1$ does nothing). However, this definition is stronger (i.e., harder to satisfy and captures a wider class of games) if τ and k are allowed to be non-empty.

If the best distinguisher for the above game has only ϵ probability of distinguishing the case of $b = 0$ from $b = 1$, then we say that a construction $\{C^\pi_\mathsf{k}\}_\mathsf{k}$ is (T_1, T_2, ϵ)-offline-online-indifferentiable from Π. For such a construction, we prove a composition theorem saying that for any game $\mathsf{G}^{C^\pi}(\mathcal{A})$ involving the construction C^π, where C^π is (T_1, T_2, ϵ)-offline-online indifferentiable from Π, and for any (T_1, T_2)-offline-online adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, there is a $(0, T')$-offline-online adversary

$\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$, where T' is roughly proportional to T_2, such that

$$\Pr\left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\!\!\$\ \{0,1\}^n \\ 1 \leftarrow\!\!\$\ \mathsf{G}^{C_{\mathsf{k}}^{\pi}}(\mathcal{A}_2^{C_{\mathsf{k}}^{\pi},\pi}(\tau,\mathsf{k})) \end{array}\right] \leq \Pr\left[\begin{array}{c} \tau \leftarrow \mathcal{B}_1 \\ \mathsf{k} \leftarrow\!\!\$\ \{0,1\}^n \\ 1 \leftarrow\!\!\$\ \mathsf{G}^{\Pi}(\mathcal{B}_2^{\Pi}(\tau,\mathsf{k})) \end{array}\right] + \epsilon\,.$$

In words, the above guarantee allows us to perform an analysis of the security of an idealized game G^{C_π} with Π as the ideal object, instead of C^π, incurring an additive cost of at most ϵ in the resulting adversarial advantage. Moreover, this analysis needs to only take into account online-only adversaries because $\mathcal{B}_1$ does not have access to Π, thereby possibly making the required analysis much simpler. This statement holds for all "offline-online" games. We refer to Definition 3 and Theorem 3.2 for the precise definition of offline-online-indifferentiability and the associated composition theorem, respectively.

Our offline-online-indifferentiability bounds for Merkle-Damgård. We prove offline-online-indifferentiability bounds for the popular keyed version of the Merkle-Damgård (MD) construction which iterates an (idealized) basic compressing function in order to get a variable-input-length hash function, operating on arbitrary longer input lengths. Precisely, the MD construction is defined relative to a compressing function $\pi\colon [N] \times [M] \to [N]$,[1] and is defined as

$$\mathsf{MD}_\pi(\mathsf{k}, (\alpha_1, \ldots, \alpha_\ell)) = \pi(\mathsf{MD}_\pi(\mathsf{k}, (\alpha_1, \ldots, \alpha_{\ell-1})), \alpha_\ell),$$

where $\mathsf{MD}_\pi(\mathsf{k}, \alpha) = \pi(\mathsf{k}, \alpha)$, and $\mathsf{k} \in [N]$ and $\alpha, \alpha_1, \ldots, \alpha_\ell \in [M]$.[2] The value k is referred to as a *key* (sometimes also called IV or salt) and the following ℓ values are referred to as *blocks*.

To present our result, we consider the ℓ-block version of the above construction with prefix-free message encoding, called *ℓ-block-MD*, where the number blocks that it can process is ℓ.[3] We prove the following offline-online-indifferentiability bound for this construction:

Theorem 1.1 (Informal and simplified; see Theorem 5.1)**.** *Let $\ell, T_1, T_2 \geq 1$. The ℓ-block-MD with prefix-free message encoding is (T_1, T_2, ϵ)-offline-online-indifferentiable for*

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

Above, we assume T_2 is the number of *message blocks* queried by the online phase of the adversary, as opposed to just the number of queries.

Note that the bound above is meaningful and especially interesting when $T_1 \in [N^{1/2}, N)$. In particular, our bound shows that the MD construction remains somewhat secure even if the offline phase can easily find collisions. We remark that for preimage-resistance and collision resistance, the bounds of [GT23] are quantitatively better: namely, ignoring constants, for pre-image resistance they prove a tight bound of $T_1^2/N^3 + T_1T_2/N^2 + T_2/N$ and for collision resistance they prove a bound of $T_1^2/N^{7/3} + T_1/N^{5/4} + T_1T_2/N^{3/2} + T_2^2/N$.

To complement the picture, we show that (essentially) each term in the above bound is necessary, making the upper bound almost tight. Specifically, we show that for every $\ell, T_1, T_2 \geq 1$, there is a

[1] We use the notation $[N]$ to denote the set $\{1, 2, \ldots, N\}$ for a natural number N.

[2] We do not mention it here for simplicity of presentation, but in the technical sections, we consider constructions where a prefix-free encoding is applied on the input before hashing it. This is used to prevent well-known length-extensions attacks.

[3] In the technical part, we analyze a slightly stronger primitive where one is allowed to query MD_π with inputs consisting of *up to* ℓ blocks (instead of exactly ℓ blocks) with prefix-free message encoding.

(T_1, T_2)-adversary for ℓ-block-MD whose winning probability in the indifferentiability game is

$$\epsilon \geq \max\left\{\frac{T_1 T_2}{N}, \frac{T_2^2}{N}\right\}.$$

The second term is merely an (online-only) birthday attack and the first term is the more interesting one, utilizing the offline phase. Obviously, the above attack is tight up to constants and an ℓ multiplicative factor in both terms. We discuss the (somewhat artificial) reason for the ℓ factor in Remark 6.3.

An optimally-secure construction. As mentioned, the second term in the indifferentiability guarantee of the ℓ-block-MD construction comes from a birthday attack which seems unavoidable for every construction. The more interesting and perhaps more significant term is the first one. Indeed, the bound above is especially interesting when $T_1 \gg N^{1/2}$ and in this case it is always the case that $T_1 T_2 \geq T_2^2$, thereby making the first term the dominant one. We ask whether the $T_1 T_2/N$ term is necessary for every construction. To this end, we consider a variant of the MD construction, called *keyed-MD* [GB08, FGK23], which is the same as the above MD, except that the key k is fed into each block in the chain (rather than only to the first one). More precisely, $\pi\colon [N]^2 \times [M] \to [N]$, and

$$\mathsf{MD}_\pi(\mathsf{k}, (\alpha_1, \ldots, \alpha_B)) = \pi(\mathsf{k}, \mathsf{MD}_\pi(\mathsf{k}, (\alpha_1, \ldots, \alpha_{B-1})), \alpha_B)$$

for $\mathsf{k} \in [N]$ and $\alpha_1, \ldots, \alpha_B \in [M]$.

This variant of MD has been recently studied in the context of (S, T)-attacks (with bounded size advice) in [FGK23], where it was shown to be significantly more robust than the first (more popular) variant of MD from above. We show that the ℓ-block version of keyed-MD is more secure than MD even in our offline-online-indifferentiability setting. In particular, we show essentially optimal security: the only ways to attack the indifferentiability of the construction are either by guessing the key in the offline phase, or by performing a birthday attack in the online phase. In other words, there is no "non-trivial" use of the offline phase.

Theorem 1.2 (Informal and simplified; see Theorem 5.3)**.** *Let $\ell, T_1, T_2 \geq 1$. The ℓ-block-keyed-MD is (T_1, T_2, ϵ)-offline-online-indifferentiable for*

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

The second term is the same as in our bound for the first MD construction, coming from a possible (online only) birthday attack. The main improvement in the bound for keyed-MD over the one for MD is that we have a T_1/N term instead of a $T_1 T_2/N$ term, thereby providing security for a much larger range of parameters. For example, for $T_1 = N^{0.7}$ and $T_2 = N^{0.3}$, the MD construction is broken with constant probability, whereas the keyed-MD construction still guarantees security, except with $\approx N^{-0.3}$ probability.

It is not hard to see that this bound is essentially tight. Specifically, we show that for every $\ell, T_1, T_2 \geq 1$, there is a (T_1, T_2)-adversary for ℓ-block-keyed-MD whose winning probability in the indifferentiability game is

$$\epsilon \geq \max\left\{\frac{T_1}{N}, \frac{T_2^2}{N}\right\}.$$

1.2 Related Work

The works of [MPS12, DGHM13, Mit14, DFMT20] all suggest various generalizations of the indifferentiability framework and sometimes present an associated composition theorem and bounds on existing constructions.

Mandal, Patarin, and Seurin [MPS12] considered a notion called *sequential indifferentiability*, wherein the adversary has access to several oracles but there is a pre-defined sequence of allowed calls to the oracles. Nevertheless, the query complexity of the adversary is measured as the total number of queries made to all oracles together. This is the main point of difference from our offline-online model in which the offline phase can support many more queries than the online phase (which is the regime one cares about in the preprocessing setting). If not for this difference, the offline-online model could be rephrased in the language of sequential indifferentiability (but then bounds would be much less meaningful).

Demay et al. [DGHM13] suggested a variant of indifferentiability called *resource-restricted* indifferentiability which lies somewhere in between plain indifferentiability and reset indifferentiability. However, we are not aware of any meaningful positive results, not in constructions nor in composition (i.e., constructions that achieve any form of resource-restricted indifferentiability or security games for which a resource-restricted construction allows composition).

Mittelbach [Mit14] (essentially) weakened the reset indifferentiability definition by changing the order of quantifiers between the simulator and the distinguisher, allowing the simulator to run and feed queries to the ideal object that are longer than the running time of the distinguisher. While this relaxation allowed [Mit14] to prove bounds for various constructions and formalize a corresponding composition theorem, it has caveats: (1) because the simulator's running time is unrealistically large, the resulting composition theorem is extremely weak and not so meaningful, (2) their simulator needs to feed extremely long inputs to the ideal primitive, meaning that the proof only works for hash functions that support unrealistically large input length, and (3) the proven bounds are very weak, providing security guarantees only for attackers that have overall sub-birthday complexity (which is typically a somewhat less interesting regime).

Dodis et al. [DFMT20] consider a strengthening of the indifferentiability framework, called *indifferentiability with auxiliary input*, where it is assumed that the adversary has limited amount of advice about the random oracle before the online phase begins (in which it can make a bounded number of queries). They suggest a concrete notion that does not lend itself to a useful composition theorem (in particular, the blow-up in the size of the advice is significant because they do not limit the state size of the simulator), but otherwise they manage to prove an indifferentiability result for constructions where every invocation of the random oracle is salted. Quantitatively, their bound is not known to be tight (and we conjecture it is not),[4] while our bounds are tight (see Section 6). Overall, while they manage to prove a non-trivial result, it applies only to a particular class of constructions (and not to classical ones like salted Merkle-Damgård) and it is not very useful in applications.

2 Technical Overview

In this section, we give an overview of the proof techniques we use to prove our concrete offline-online indifferentiability bounds for Merkle-Damgård (MD). Recall that the MD construction is

[4] We note that the bound they prove in the second part of their Theorem 3 is incorrect as stated. Specifically, the second term of the bound should have a multiplicative T factor. The source of the error is in their application of Theorem 5 from [CDGS18] where they miss the T term. While this is not directly related to our results, we wanted to make a note of this.

an iterative hashing mechanism that hashes messages of variable input length to a fixed output length, given a randomly sampled key or salt. The underlying fundamental primitive for MD is a compression function $h : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ which is modeled as a random oracle. In order to hash a message M with respect to a key $\mathsf{k} \in \{0,1\}^n$, M is first padded appropriately so that its length is a multiple of m and then broken into blocks $M_1, \ldots, M_B$, each m bits long. Then, y_0 is set to k and for $i = 1, 2, \ldots, B$, y_i is computed as $h(y_{i-1}, M_i)$. The hash output is y_B. In this overview, we will only consider two-block MD, where the message M has bit-length $2m$. While in the technical section we consider the more general ℓ-block case, the proof overview for the two-block case already highlights some of our main ideas.

In order to prove the offline-online indifferentiability of the two-block MD construction from a random oracle $\Pi : \{0,1\}^{2m} \to \{0,1\}^n$, we need to consider the following two worlds:

- **Real world:** In the offline phase, the adversary can issue queries to the random oracle h. Then, a key k is chosen uniformly at random, and in the online phase, the adversary can issue queries to $\mathsf{MD}^h_{\mathsf{k}}$, the two-block MD construction using h with salt k.
- **Ideal world:** In the offline phase, the adversary can issue queries to the simulator $\mathcal{S}_1$. Then, Π is chosen uniformly at random, and in the online phase, the adversary can issue queries to Π and $\mathcal{S}_2$.

For an appropriately-defined simulator, we need to prove that these two worlds are indistinguishable for any (T_1, T_2)-adversary, except with small probability. Recall that a (T_1, T_2)-adversary issues T_1 queries to its oracles in the offline phase and T_2 queries to its oracles in the online phase.

We define the simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ as follows: $\mathcal{S}_1$ simply answers the queries of the adversary uniformly at random (and consistently if a repeated query occurs), and storing the responses in a table. $\mathcal{S}_2$, knowing k, checks at the beginning whether $\mathcal{A}_1$ had made any queries prefixed with k, and simply aborts if that is the case. Otherwise, it answers the queries of the adversary uniformly at random, keeping them consistent with the answers of $\mathcal{S}_1$, except for the following: If there is a query on (x, y) such that there was a query on (k, z) earlier that it answered with x, then it answers this query with $\Pi(z, y)$. Observe that the number of queries issued by the simulator in the offline and online phases is same as the number of queries issued by the offline and online phases of $\mathcal{A}$, i.e., the query complexity is preserved.

We need to prove that the adversary's views in the two worlds mentioned above are indistinguishable except with small probability. For this, we consider a sequence of hybrid games, starting with the real world and culminating in the ideal world. Since each two "adjacent" hybrid games will be close, by a union bound we will be able to conclude that the first and last worlds (the real and ideal, respectively) are close. Next, we present the main hybrids in the proof, noting that the actual proof requires more intermediate steps that we skip for this overview (see Section 4 for the full proof).

1. **Hybrid 1**: This hybrid simulates the real world to the adversary.
2. **Hybrid 2**: In this hybrid, the main changes compared to the previous hybrid are as follows.
   - The random oracle query answers are lazily sampled.
   - We introduce the following bad event: if the there is a MD query on (x, y), such that (1) the value of the random oracle had not been already defined on (k, x), (2) the value z is sampled as the value of the random oracle at that point, and (3) the value of the random oracle at (z, y) is already defined. If the bad event happens, we resample the value of the random oracle on (z, y).

Hybrid 2 is identical to hybrid 1, except if the bad event happens. Notice that the bad event happens only if on an MD query (x, y), the value of the random oracle at (k, x) is sampled to be z such that the value of the random oracle on some point of the form $(z, *)$ was defined already. Since every MD query causes at most two new random oracle evaluations, for a given MD query, this happens with probability at most $(T_1 + 2T_2)/2^n$. By a union bound over all MD queries, one can distinguish this hybrid from the previous one with probability at most $(T_1T_2 + 2T_2^2)/2^n$.

3. **Hybrid 3**: In this hybrid, two types of bad events: **1)**the sampled key k is such that the offline phase of the adversary made a query prefixed with k **2)** the lazily sampled value for one of the two oracles is either an input salt or output of an offline query, or same as some other sampled value. If the first type of bad event happens, the hybrid aborts. Further, the answer of the MD queries are always freshly sampled. From the details of the hybrid, it is not hard to see that this hybrid is identical to hybrid 2 unless the bad event happens. Further, the probability that the bad event happens is at most $T_1/2^n + 2T_1T_2/2^n + 2T_2^2/2^n$ because k is sampled uniformly at random and there are T_1 offline queries and at most T_2 online queries that came earlier.

4. **Hybrid 4**: This hybrid simulates the ideal world to the adversary. With the details that we omit here, it is not hard to verify that the behavior of hybrids 3 and 4 are identical.

Putting it all together, we have that the adversary can distinguish the real and the ideal worlds with probability at most $T_1/2^n + 3T_1T_2/2^n + 2T_2^2/2^n \leq 4T_1T_2/2^n + 2T_2^2/2^n$, as needed.

Tightness. We argue that this bound is (almost) tight. In the regime $1 \leq T_2 < 2^{n/2}$, the term $T_1T_2/2^n$ always dominates $T_1/2^n$. We give the following attack that achieves advantage roughly $T_1T_2/2^n$:

- **Offline phase**: adversary makes queries on T_1 distinct salts.
- **Online phase**: given the salt k, the adversary makes $T_2 - 1$ distinct queries on that salt. If there is any query (say (k, x)) in the online phase such that the answer of the query (say z) is same as one of the salts the adversary queried in the offline phase (say (z, w)), the adversary queries (x, w) to the MD oracle to check if the answer is consistent with the answer of the query on (z, w). If it is consistent, the adversary outputs "real", and otherwise it says "ideal".

Indeed, by definition, in the real world the answer will always be consistent. In the ideal world, consistency will not be satisfied with overwhelming probability because the offline phase of the simulator is independent of Π. Note that the adversary succeeds in making such an online query with probability roughly $T_1T_2/2^n$. Finally, the tightness of the term $T_2^2/2^n$ can be shown by the following (standard) birthday attack:

- **Offline phase**: do nothing.
- **Online phase**: given the salt k, the adversary makes a query on (k, b) for arbitrary b – say the output is c. Now, it makes $T_2 - 3$ distinct queries on c until it finds $x \neq y$ such that (c, x) and (c, y) have the same answer (which happens with probability roughly $T_2^2/2^n$). Finally, the adversary makes MD queries on (b, x) and (b, y). If the answers are consistent, the adversary outputs "real", and otherwise "ideal".

Offline-online-indifferentiability for ℓ-block MD and keyed-MD. The proof for offline-online indifferentiability for ℓ-block MD follows a very similar flow. The proof for ℓ-block keyed-MD is

also similar to ℓ-block MD, with one crucial difference in the analysis. In the transition between hybrids 1 and 2 described above, the probability that the bad event happens is $T_1T_2/2^{2n}$ instead of $T_1T_2/2^n$. This is true since for the bad event to happen at all, the adversary needs to make an offline query on the correct salt which is sampled after the offline phase. Thus, redoing the analysis with the above bound, the adversary can distinguish the real and the ideal worlds (in the 2-block case) with probability roughly $T_1/2^n + T_1T_2/2^{2n} + 2T_2^2/2^n \leq 2T_1/2^n + 2T_2^2/2^n$, as needed.

3 The Definition of Offline-Online Indifferentiability

Preliminaries. Let $\mathbb{N} = \{0, 1, \dots\}$ and $[n] = \{1, \dots, n\}$ for $n \in \mathbb{N}$. If $x \in \{0,1\}^*$ is a string, then $|x|$ denotes its length in bits. If S is a set, then $|S|$ denotes its size and S^+ denotes the set of all lists that have one or more elements of S. $\mathsf{Fcs}(D, R)$ denotes the set of all f such that $f(\cdot)$ is a function from D to R. We regularly use pseudocode inspired by the code-based framework of [BR06]. If $\mathcal{A}$ is an algorithm, then $y \leftarrow \mathcal{A}^{O_1,\cdots}(x_1, \dots; r)$ denotes running $\mathcal{A}$ on inputs $x_1, \dots$ with coins r and access to the oracles $O_1, \dots$ to produce output y. When the coins are implicit we write $\leftarrow\!\!\$$ in place of $\leftarrow$ and omit r.

We let $x \leftarrow\!\!\$\, \mathcal{D}$ denote sampling x according to the distribution $\mathcal{D}$. If $\mathcal{D}$ is a set, we overload notation and let $\mathcal{D}$ also denote the uniform distribution over elements of $\mathcal{D}$. The probability that G outputs true is denoted $\Pr[\mathsf{G}]$.

A cryptographic construction describes a cryptographic algorithm e.g., a hash function, an encryption scheme, etc. A cryptographic construction might use another cryptographic object (which may be another cryptographic construction or a cryptographic primitive) as a building block in a black-box manner. In this case, we abstract it as an oracle-aided circuit that uses a primitive π. We denote the oracle-aided circuit as C and use C^π to denote the construction.

Constructions built from ideal primitives. We are interested in a special class of constructions that depend on an *ideal primitive*, such as a random oracle, random permutation, ideal cipher, etc. We model the ideal primitive π 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 the set $\mathsf{Fcs}(\mathsf{D}, \mathsf{R})$ of all functions mapping D to R.

Games with ideal primitives. A *game* G describes an environment with which an adversary $\mathcal{A}$ can interact, 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 $\mathsf{G}(\mathcal{A})$. 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 a pre-specified set of possible functions π, which we refer to as *compatible* with the game G. We denote by $\mathsf{G}^\pi(\mathcal{A}^\pi)$ the experiment where $\mathcal{A}$ plays the game G^π, and is given access to the same π as the game. We overload notation and also use $\mathsf{G}^\pi(\mathcal{A}^\pi)$ to denote 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 say that construction C is *compatible* with an ideal primitive $\mathcal{I}$, if the range of $\mathcal{I}$ is a subset of the compatible objects for C. For a construction compatible with $\mathcal{I}$, we denote by $\mathsf{G}^{C^\pi}(\mathcal{A}^{\pi, C^\pi})$ an oracle game G which is compatible with $\mathcal{I}$ where the game can make queries to C^π and the adversary can make queries to both C^π and π. Further, we use $\mathsf{G}^{C^\pi_{\mathsf{k}}}(\mathcal{A}^{\pi, C^\pi}(\tau, \mathsf{k}))$ to denote that in this random experiment, the adversary $\mathcal{A}$ when run has inputs τ and k.

Offline-online adversaries. As in [GT23], we define the notion of offline-online adversaries. An offline-online adversary $\mathcal{A}$ is split into two parts, the offline adversary $\mathcal{A}_1$ and the online adversary $\mathcal{A}_2$. In the *offline stage*, $\mathcal{A}_1$ is given access *only* to the ideal primitive oracle π. At the end of this

stage, $\mathcal{A}_1$ outputs the entire transcript of its queries τ. Then, in the *online stage*, adversary $\mathcal{A}_2$ is initialized with τ (and possibly other inputs) and given access to the oracle π and C^π. 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 to its oracles. We can assume without loss of generality that both $\mathcal{A}_1$ and $\mathcal{A}_2$ are deterministic because we can fix the best possible randomness that maximizes the advantage of $\mathcal{A}$.

As argued in [GT23], some games are more interesting than others in the context of offline-online adversaries. As in that work, we will focus on salted cryptographic primitives $\mathcal{I}$ which permit an additional input–called a *salt*–that is chosen in the online phase of an attack.

Defining indifferentiability against offline-online adversaries. We extend the notion of indifferentiability [MRH04] to the context of offline-online adversaries. Recall that in the original indifferentiability notion, saying that C that uses π is indifferentiable from an ideal object Π requires having a real world where the adversary has access to the construction C and an oracle for the ideal object π. In the ideal world, in contrast, the adversary has access to the ideal object oracle Π and a simulator which provides the same interface as π but simulates it using Π. There are several challenges in extending this meaningfully to the notion of an offline-online adversary. First, since the adversary is two-stage, the simulator needs to be two-stage. So the following questions arise: **1)** what state should the two simulator stages be allowed to share, and **2)** do both of the simulator stages have access to Π?

In order to answer these, we need to remember that the main goal of indifferentiability is to show that for any adversary $\mathcal{A}$ playing a game where it has access to both C and π, there is an adversary $\mathcal{B}$ that only has access to Π and has similar advantage. This allows for composition. Moreover, if $\mathcal{B}$ is online-only, then the latter game is easier to analyze.

In the offline-online model, the offline stage of $\mathcal{A}$ outputs the entire query transcript that is sent to the online phase of $\mathcal{A}$. So, we similarly allow the first stage of the simulator to send over the entire query transcript of the adversary $\mathcal{A}$ to the second stage. This means that the resulting $\mathcal{B}$ does the same thing. As for the second question, since we aim for an online-only $\mathcal{B}$, we allow *only* the second stage of the simulator to access Π. Below, we state the formal definition of offline-online indifferentiability.

Definition 3.1 ((T_1, T_2, T', ϵ)-offline-online-indifferentiability)**.** Let Samp be a sampler that takes as input a salt length λ, and outputs the salt according to some specific distribution. Let C be an oracle-aided circuit, let $\mathcal{I}$ and $\mathcal{I}'$ be ideal primitives, and let $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ be a simulator. We define the offline-online-indifferentiability advantage of an adversary $\mathcal{A}$ against C with respect to $\mathsf{Samp}, \mathcal{I}$ and $\mathcal{S}$ for $\lambda \in \mathbb{N}^+$ from $\mathcal{I}'$ as follows.

$$\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) := \left| \Pr_{\pi \leftarrow\$ \mathcal{I}} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^\lambda) \\ 1 \leftarrow\$ \mathcal{A}_2^{C_{\mathsf{k}}^{\pi}, \pi}(\tau, \mathsf{k}) \end{array} \right] - \Pr_{\Pi \leftarrow\$ \mathcal{I}'} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\mathcal{S}_1} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^\lambda) \\ 1 \leftarrow\$ \mathcal{A}_2^{\Pi, \mathcal{S}_2^{\Pi}(\tau,\mathsf{k})}(\tau, \mathsf{k}) \end{array} \right] \right|$$

We say that C is (T_1, T_2, T', ϵ)-offline-online-indifferentiable from $\mathcal{I}'$ with respect to $\mathcal{I}, \mathsf{Samp}$ if for any (T_1, T_2)-adversary $\mathcal{A}$ there exists a $(0, T')$-simulator $\mathcal{S}$ such that $\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) \leq \epsilon$ for all sufficiently large $\lambda \in \mathbb{N}$.

Comparison with other definitions. We briefly compare our definition of offline-online indifferenitiability to prior attempts at similar definitions.

- Our definition is a generalization of the indifferenitiability definition of Maurer, Renner and Holenstein [MRH04]. Specifically setting $T_1 = 0$ recovers the definition and the composition theorem of [MRH04].

- Riestenpart et al [RSS11] formalized the notion of reset indifferentiability, a strengthening of indifferentiability that enables an indifferentiability composition theorem covering multi-stage security notions. However, they show that practical hash constructions cannot be reset indifferentiable. Our indifferenitiability notion is *incomparable to* i.e., neither implied by nor implies reset indifferenitiability.

- Mandal, Patarin, and Seurin [MPS12] considered a notion called *sequential indifferentiability*, wherein the adversary has access to several oracles but there is a pre-defined sequence of allowed calls to the oracles. While we can cast our indifferenitiability notion in this framework, the query complexity of the adversary against sequential indifferentiability is measured as the total number of queries made to all oracles together. This means that if we use sequential indifferenitiability, all the bounds would be in terms of T_1+T_2. As such, one could not hope for bounds which remain meaningful in regimes when $T_1 >> 2^n/2 >> T_2$ for security properties like collision-resistance. In fact for any preprocessing adversary, the regime of $T_1 >> 2^{n/2}$ is highly realistic. Our work, unlike that of [MPS12], provides meaningful bounds in that regime. Our indifferenitiability notion is *incomparable to* the notion of sequential indifferentiability, because while our definition is more general for the specific setting of offline-online adversaries, sequential indifferenitiability can capture more types of adversaries which have more than two stages.

- Demay et al. [DGHM13] suggested a variant of indifferentiability called *resource-restricted* indifferentiability which lies somewhere in between plain indifferentiability and reset indifferentiability. However, this notion has only been used to prove impossibility results. Again, resource-restricted indifferenitiability is incomparable to our notion of offline-online indifferenitiability.

- Mittelbach [Mit14] weakened the reset indifferentiability definition by changing the order of quantifiers between the simulator and the distinguisher, allowing the simulator to run and feed queries to the ideal object that are longer than the running time of the distinguisher. Their definition is also neither implied by, nor implies our notion of offline-online indifferentiability.

  While the relaxation allowed [Mit14] to prove bounds for various constructions and formalize a corresponding composition theorem, it suffers from the following caveats that our work does not encounter.

  - Their simulator they give in their proofs has unrealistically large running time, and therefore the resulting composition theorem is extremely weak and not meaningful
  - Further, their simulator in the proof needs to feed extremely long inputs to the ideal primitive, meaning that the proof only works for hash functions that support unrealistically large input length
  - The bounds they prove are very weak, providing security guarantees only for attackers that have overall sub-birthday complexity (which is typically a somewhat less interesting regime). In contrast, we can prove guarantees against attackers that have offline running time $T_1 >> 2^{n/2}$.

- Dodis et al. [DFMT20] consider a strengthening of the indifferentiability framework, called *indifferentiability with auxiliary input*, where it is assumed that the adversary has limited amount of advice about the random oracle before the online phase begins (in which it can make a bounded number of queries). They suggest a concrete notion that does not lend itself

to a useful composition theorem (in particular, because they do not limit the state-size of the simulator in their notion, the blow-up in the size of the advice can significant).

Their indifferenitiability notion is not implied by, nor does it imply our definition of offline-online indifferenitiability. However, in contrast to their work, since we limit the online queries of the simulator and do not give the offline phase of the simulator access to the ideal object, we get a meaningful composition theorem. Moreover, the bound they prove for their notion is not known to be tight, while our bounds are tight. Overall, they manage to prove a non-trivial result for constructions where every invocation of the random oracle is salted. Our notion of offline-online indifferentiability does not impose such restriction on the construction and unlike their notion, can be used to prove results for common constructions such that the salted Merkle-Damgård construction.

3.1 The Composition Theorem

Theorem 3.2. *Let $T_1, T_2, T_C > 1$ and let $\epsilon < 1$. Let C be an oracle-aided construction, let $\mathcal{I}$ and $\mathcal{I}'$ be ideal primitives and G be an oracle game such that C is compatible with $\mathcal{I}$ and G is compatible with C and $\mathcal{I}'$. Let Samp be a sampler such that C is $(T_1, T_2 + T_C, T', \epsilon)$-offline-online-indifferentiable from $\mathcal{I}'$ with respect to $\mathcal{I}, \mathsf{Samp}$. Then for every (T_1, T_2)-adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ we can construct a $(0, T')$-adversary $\mathcal{B}$ where for $\lambda \in \mathbb{N}$*

$$\left| \Pr_{\pi \leftarrow\$ \mathcal{I}} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^\lambda) \\ 1 \leftarrow\$ \mathsf{G}^{C_{\mathsf{k}}^{\pi}}(\mathcal{A}_2^{C_{\mathsf{k}}^{\pi}, \pi}(\tau, \mathsf{k})) \end{array} \right] - \Pr_{\Pi \leftarrow\$ \mathcal{I}'} \left[\begin{array}{c} \tau \leftarrow \mathcal{B}_1 \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^\lambda) \\ 1 \leftarrow\$ \mathsf{G}^{\Pi}(\mathcal{B}_2^{\Pi}(\tau, \mathsf{k})) \end{array} \right] \right| \leq \epsilon \ .$$

Proof. We first define the following adversary $\mathcal{A}' = (\mathcal{A}'_1, \mathcal{A}'_2)$ against the offline-online indifferentiability of C. $\mathcal{A}'_1$ is same as $\mathcal{A}_1$. $\mathcal{A}'_2$ gets the query transcript and the salt as input: it runs game G, forwarding the game's oracle queries to its first oracle. When G runs the adversary $\mathcal{A}_2$, $\mathcal{A}'_2$ starts running $\mathcal{A}_2$ with its transcript and salt, answering $\mathcal{A}_2$'s queries to its own two oracles. It finally outputs whatever the game outputs.

Let $\mathcal{S}$ be the $(0, T')$ simulator for C^π with respect to $\mathcal{A}, \mathsf{Samp}$ guaranteed by the definition. We next define the adversary $\mathcal{B}$ as follows: the adversary $\mathcal{B}_1$ runs $\mathcal{A}_1$ answering its oracle queries by simulating $\mathcal{S}_1$ internally. The adversary $\mathcal{B}_1$ outputs τ, the output of $\mathcal{A}_1$. The adversary $\mathcal{B}_2$ which has oracle access to Π and gets τ, a as input runs $\mathcal{A}_2$ with these inputs. For every query $\mathcal{A}_2$ makes to C_{k}^{π}, $\mathcal{B}_2$ forwards the query to Π and forwards the answers. For every π query made by $\mathcal{A}_2$, $\mathcal{B}_2$ answers them by simulating $\mathcal{S}_2$ internally (it can do this because it has access to Π and the transcript of queries τ).

It is easy to see that the adversary $\mathcal{A}'_1$, when given access to π and $\mathcal{A}'_2$ given access to $C_{\mathsf{k}}^{\pi}, \pi$ perfectly simulates the experiment on the left-hand side of the expression in the theorem statement to $\mathcal{A}$. From the description of $\mathcal{B}$, it follows that the experiment where the adversary $\mathcal{A}'_1$, when given access to $\mathcal{S}_1$ and $\mathcal{A}'_2$ given access to $\Pi, \mathcal{S}_2^{\Pi}$ is identical to the experiment on the right-hand side of the expression in the theorem statement. Since $\mathcal{A}'$ is a $(T_1, T_2 + T_C)$ adversary and C is $(T_1, T_2 + T_C, T', \epsilon)$-indifferentiable from $\mathcal{I}'$ with respect to $\mathcal{I}, \mathsf{Samp}$, the claimed advantage bound follows. Also the claims about the number of queries made by $\mathcal{B}$ follows from its description.

■

4 Warm-up: Offline-Online Indifferentiability of 2-Block Merkle-Damgård

In this section, we give a warm-up formal proof of indifferentiability for 2-block Merkle-Damgård (MD). The 2-MD construction is defined relative to a compressing function $\pi\colon \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$, and is defined as

$$\text{2-MD}_\pi(\mathsf{k}, (\alpha_1, \alpha_2)) = \pi(\pi(\mathsf{k}, \alpha_1), \alpha_2),$$

where $\mathsf{k} \in \{0,1\}^n$ and $\alpha_1, \alpha_2 \in \{0,1\}^m$.

Theorem 4.1. *Let $C :=$ 2-MD be the two-block Merkle-Damgård construction. Let $m, n \in \mathbb{N}$. Let $\mathcal{I}$ be the uniform distribution on $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^m, \{0,1\}^n)$. Let Samp be a sampler that takes as input a salt length n, and outputs a string uniformly at random in $\{0,1\}^n$. Let $\mathcal{I}'$ be the uniform distribution on $\mathsf{Fcs}((\{0,1\}^m)^+, \{0,1\}^n)$. Then, there exists a $(0, T_2)$-simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for all (T_1, T_2)-adversaries $\mathcal{A}$,*

$$\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) \le \frac{T_1}{2^n} + \frac{3T_1T_2}{2^n} + \frac{4T_2^2}{2^n}\ .$$

Proof. Recall that

$$\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) := \left| \Pr_{\pi \leftarrow\$ \mathcal{I}} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^n) \\ 1 \leftarrow\$ \mathcal{A}_2^{\text{2-MDO}^{\pi}_{\mathsf{k}},\pi}(\tau, \mathsf{k}) \end{array} \right] - \Pr_{\Pi \leftarrow\$ \mathcal{I}'} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\mathcal{S}_1} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^n) \\ 1 \leftarrow\$ \mathcal{A}_2^{\Pi,\mathcal{S}_2^{\Pi}(\tau,\mathsf{k})}(\tau, \mathsf{k}) \end{array} \right] \right|.$$

We start by defining the simulator $\mathcal{S}$ below. The initialization procedure of $\mathcal{S}_1$ simply initializes a table T to $\perp$. $\mathcal{S}_1$ answers the queries by lazily sampling its answer. The initialization procedure of $\mathcal{S}_2$ checks if there was a query made on k in the offline phase – if so, it simply aborts. To answer a query, $\mathcal{S}_2$ checks if the query is prefixed by x_1 such that there was an earlier query prefixed by k that had answer x_1, then it uses Π, otherwise answers by lazily sampling. It follows from the description that the simulator makes no offline queries and at most T_2 online queries.

Simulator $\mathcal{S}_1$	Simulator $\mathcal{S}_2^{\Pi}(\mathsf{k}, \tau)$
Procedure Init $T \leftarrow \perp$	Procedure Init If $\exists x : T[\mathsf{k}][x] \neq \perp$: **Abort**

Procedure Query(x_1, x_2)
  If $T[x_1][x_2] = \perp$:
  $T[x_1][x_2] \leftarrow\$ \{0,1\}^n$
  Return $T[x_1][x_2]$
	Procedure Query(x_1, x_2)
	If $T[x_1][x_2] = \perp$:
	  If $\exists y : T[\mathsf{k}][y] = x_1$
	  $T[x_1][x_2] \leftarrow \Pi(y, x_2)$
	  Else $T[x_1][x_2] \leftarrow\$ \{0,1\}^n$
	Return $T[x_1][x_2]$

We now prove the advantage claim for $\mathcal{S}$ using a sequence of games. We first describe the game hops at a high level. We start with the real world. As a first step, we will lazily sample the random oracle. We then introduce the following bad event: if the there is a MD query on (x, y), such that the value of the random oracle had not been already defined on (k, x), the value z is sampled as the value of the random oracle at that point, and the value of the random oracle at (z, y) is already defined. If the bad event happens, we resample the value of the random oracle on (z, y). Then, we move to a game where we define two types of bad events: **1)**the sampled key k is such that the

offline phase of the adversary made a query prefixed with k **2)** the lazily sampled value for one of the two oracles is either an input salt or output of an offline query, or same as some other sampled value. If the first type of bad event happens, the hybrid aborts. Further, the answer of the MD queries are always freshly sampled. Finally, we move to the ideal world.

Concretely, we will define the games G_0-G_{10} below and prove the following claims.

- $\Pr_{\pi \leftarrow\$ \mathcal{I}}\left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^n) \\ 1 \leftarrow\$ \mathcal{A}_2^{\text{2-MDO}_{\mathsf{k}}^{\pi},\pi}(\tau, \mathsf{k}) \end{array}\right] = \Pr[\mathsf{G}_0]$.
- $\Pr[\mathsf{G}_1] = \Pr[\mathsf{G}_0]$.
- $\Pr[\mathsf{G}_2] = \Pr[\mathsf{G}_1]$.
- $\Pr[\mathsf{G}_3] = \Pr[\mathsf{G}_2] + \frac{T_1 T_2}{2^n} + \frac{T_2^2}{2^n}$.
- $\Pr[\mathsf{G}_4] = \Pr[\mathsf{G}_3]$
- $\Pr[\mathsf{G}_5] = \Pr[\mathsf{G}_4]$.
- $\Pr[\mathsf{G}_6] = \Pr[\mathsf{G}_5]$.
- $\Pr[\mathsf{G}_7] = \Pr[\mathsf{G}_6]$.
- $\Pr[\mathsf{G}_8] \leq \Pr[\mathsf{G}_7] + \frac{T_1}{2^n} + \frac{2T_1 T_2}{2^n} + \frac{T_2^2}{2^n}$.
- $\Pr[\mathsf{G}_9] = \Pr[\mathsf{G}_8]$.
- $\Pr[\mathsf{G}_{10}] = \Pr[\mathsf{G}_9]$.
- $\Pr[\mathsf{G}_{10}] = \Pr_{\Pi \leftarrow\$ \mathcal{I}'}\left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\mathcal{S}_1} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^n) \\ 1 \leftarrow\$ \mathcal{A}_2^{\Pi,\mathcal{S}_2^{\Pi}(\tau,\mathsf{k})}(\tau, \mathsf{k}) \end{array}\right]$.

Putting this together, the claimed advantage bound follows. We now prove the claims.

Define game G_0 as follows. In the game, the random oracle π is sampled, T, T' are initialized to $\perp$ everywhere. Then the adversary $\mathcal{A}_1$ is run with oracle access to π, it outputs a state τ. A salt k is sampled uniformly at random from $\{0,1\}^n$. The adversary $\mathcal{A}_2$ gets the salt as input, along with access to oracles $O_\pi, \text{2-MDO}_{\mathsf{k}}^{\pi}$. $\mathcal{A}_2$ returns a bit which is the output of the game. The oracle O_π simulates the random oracle π with additional bookkeeping. Similarly, the oracle $\text{2-MDO}_{\mathsf{k}}^{\pi}$ is the two-block MD construction with salt k and compression function π, with additional book-keeping in T'. It follows by inspection that

$$\Pr_{\pi \leftarrow\$ \mathcal{I}}\left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^n) \\ 1 \leftarrow\$ \mathcal{A}_2^{\text{2-MDO}_{\mathsf{k}}^{\pi},\pi}(\tau, \mathsf{k}) \end{array}\right] = \Pr[\mathsf{G}_0] \ .$$

Game G_0	Oracle 2-MDO$^{\pi}_{\mathsf{k}}(x_1, x_2)$

$\pi \leftarrow\!\!\$\ \mathcal{I}$
$T, T' \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\text{2-MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow \pi(x_1, x_2)$
  Return $T[x_1][x_2]$
	If $T'[x_1][x_2] \neq \bot$:
	  Return $T'[x_1][x_2]$
	If $T[\mathsf{k}][x_1] = \bot$:
	  $T[\mathsf{k}][x_1] \leftarrow \pi(\mathsf{k}, x_1)$
	$y \leftarrow T[\mathsf{k}][x_1]$
	If $T[y][x_2] = \bot$:
	  $T[y][x_2] \leftarrow \pi(y, x_2)$
	$T'[x_1][x_2] \leftarrow T[y][x_2]$
	Return $T'[x_1][x_2]$

We next move to game G_1, where the random oracle π is lazily sampled. This does not change the distribution of π, hence the output of the game. Therefore, we have that $\Pr[\mathsf{G}_1] = \Pr[\mathsf{G}_0]$.

Game G_1	Oracle 2-MDO$^{\pi}_{\mathsf{k}}(x_1, x_2)$

$T, T' \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\text{2-MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  Return $T[x_1][x_2]$
	If $T'[x_1][x_2] \neq \bot$:
	  Return $T'[x_1][x_2]$
	If $T[\mathsf{k}][x_1] = \bot$:
	  $T[\mathsf{k}][x_1] \leftarrow\!\!\$\ \{0,1\}^n$
	$y \leftarrow T[\mathsf{k}][x_1]$
	If $T[y][x_2] = \bot$:
	  $T[y][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	$T'[x_1][x_2] \leftarrow T[y][x_2]$
	Return $T'[x_1][x_2]$

We move to game G_2 where the only change compared to G_1 is during a 2-MDO query on x_1, x_2 if $T[\mathsf{k}][x_1] = \bot$, and $T[T[\mathsf{k}][x_1]][x_2] \neq \bot$, a flag bad is set to true. The introduction of this flag does not alter the behavior of the game, hence $\Pr[\mathsf{G}_2] = \Pr[\mathsf{G}_1]$.

Game G_2	Oracle 2-MDO$^{\pi}_{\mathsf{k}}(x_1, x_2)$

$T, T' \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\text{2-MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$
	If $T'[x_1][x_2] \neq \bot$:
	  Return $T'[x_1][x_2]$
	If $T[\mathsf{k}][x_1] = \bot$:
	  $T[\mathsf{k}][x_1] \leftarrow\!\!\$\ \{0,1\}^n$
	  If $T[T[\mathsf{k}][x_1]][x_2] \neq \bot$:
	  $\mathsf{bad} \leftarrow \mathsf{true}$
	$y \leftarrow T[\mathsf{k}][x_1]$
	If $T[y][x_2] = \bot$:
	  $T[y][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	$T'[x_1][x_2] \leftarrow T[y][x_2]$
	Return $T'[x_1][x_2]$

Game G_3	Oracle $\mathsf{2\text{-}MDO}^{\pi}_{\mathsf{k}}(x_1, x_2)$
$T, T' \leftarrow \perp$	If $T'[x_1][x_2] \neq \perp$:
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$	Return $T'[x_1][x_2]$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$	If $T[\mathsf{k}][x_1] = \perp$:
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\mathsf{2\text{-}MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$	$T[\mathsf{k}][x_1] \leftarrow\!\!\$\ \{0,1\}^n$
Return b	If $T[T[\mathsf{k}][x_1]][x_2] \neq \perp$:
	$\mathsf{bad} \leftarrow \mathsf{true}$
Oracle $O_\pi(x_1, x_2)$	$T[T[\mathsf{k}][x_1]][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
If $T[x_1][x_2] = \perp$:	$y \leftarrow T[\mathsf{k}][x_1]$
$T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$	If $T[y][x_2] = \perp$:
Return $T[x_1][x_2]$	$T[y][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	$T'[x_1][x_2] \leftarrow T[y][x_2]$
	Return $T'[x_1][x_2]$

In game G_3, $T[T[\mathsf{k}][x_1]][x_2]$ is resampled if bad is set. Since $\mathsf{G}_2, \mathsf{G}_3$ are identical-until-bad, using the Fundamental Lemma of Game Playing [BR06], we have that

$$\Pr[\mathsf{G}_3] \leq \Pr[\mathsf{G}_2] + \Pr[\mathsf{G}_2 \text{ sets } \mathsf{bad}] \ .$$

Observe that bad is set in G_2 only if the oracle 2-MDO is invoked on (x_1, x_2) such that $T[\mathsf{k}][x_1] = \perp$ and the value of $T[\mathsf{k}][x_1]$ sampled uniformly at random is such that $T[T[\mathsf{k}][x_1]][x_2] \neq \perp$. This can only happen if the sampled $T[\mathsf{k}][x_1]$ is equal to some z such that $T[z][x_2]$ was already defined. Since for every 2-MDO query, there are at most two new values of T that are defined, this happens with probability at most $(T_1 + 2T_2)/2^n$ for every 2-MDO query. Via a union bound over all 2-MD queries

$$\Pr[\mathsf{G}_2 \text{ sets } \mathsf{bad}] \leq \frac{T_1 T_2}{2^n} + \frac{2T_2^2}{2^n} \ .$$

Hence,

$$\Pr[\mathsf{G}_3] \leq \Pr[\mathsf{G}_2] + \frac{T_1 T_2}{2^n} + \frac{2T_2^2}{2^n} \ .$$

Game G_4	Oracle $\mathsf{2\text{-}MDO}^{\pi}_{\mathsf{k}}(x_1, x_2)$
$T, T' \leftarrow \perp$	If $T'[x_1][x_2] \neq \perp$:
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$	Return $T'[x_1][x_2]$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$	If $T[\mathsf{k}][x_1] = \perp$:
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\mathsf{2\text{-}MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$	$T[\mathsf{k}][x_1] \leftarrow\!\!\$\ \{0,1\}^n$
Return b	$T[T[\mathsf{k}][x_1]][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	$y \leftarrow T[\mathsf{k}][x_1]$
Oracle $O_\pi(x_1, x_2)$	If $T[y][x_2] = \perp$:
If $T[x_1][x_2] = \perp$:	$T[y][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
$T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$	$T'[x_1][x_2] \leftarrow T[y][x_2]$
Return $T[x_1][x_2]$	Return $T'[x_1][x_2]$

In game G_4, we remove the bad flag, and after the check $T[\mathsf{k}][x_1] = \perp$, we remove the check whether $T[T[\mathsf{k}][x_1]][x_2] \neq \perp$ and just always sample it. Note that this does not change behavior because in game G_3, it was resampled here if $T[T[\mathsf{k}][x_1]][x_2] \neq \perp$ and sampled below if the check failed. Hence, $\Pr[\mathsf{G}_4] = \Pr[\mathsf{G}_3]$.

Game G_5
$T, T' \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\text{2-MDO}_\mathsf{k}^\pi, O_\pi}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$
	Oracle $\text{2-MDO}_\mathsf{k}^\pi(x_1, x_2)$
	If $T'[x_1][x_2] \neq \bot$:
	  Return $T'[x_1][x_2]$
	If $T[\mathsf{k}][x_1] = \bot$:
	  $T[\mathsf{k}][x_1] \leftarrow\!\!\$\ \{0,1\}^n$
	  $T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	  $T[T[\mathsf{k}][x_1]][x_2] \leftarrow T'[x_1][x_2]$
	$y \leftarrow T[\mathsf{k}][x_1]$
	If $T[y][x_2] = \bot$:
	  $T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	  $T[y][x_2] \leftarrow T'[x_1][x_2]$
	Return $T'[x_1][x_2]$

In game G_5, instead of sampling $T[T[\mathsf{k}][x_1]][x_2]$ and later assigning it to $T'[x_1][x_2]$, we do it the other way round. This does not change behavior and hence, $\Pr[\mathsf{G}_5] = \Pr[\mathsf{G}_4]$.

Game G_6
$T, T' \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\text{2-MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi^1(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$
 	
Oracle $O_\pi^2(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  If $\exists y : T[\mathsf{k}][y] = x_1$:
  If $T'[y][x_2] = \bot$:
  $T'[y][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  $T[x_1][x_2] \leftarrow T'[y][x_2]$
  Else $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$
	Oracle $\text{2-MDO}_\mathsf{k}^\pi(x_1, x_2)$
	If $T'[x_1][x_2] \neq \bot$:
	  Return $T'[x_1][x_2]$
	If $T[\mathsf{k}][x_1] = \bot$:
	  $T[\mathsf{k}][x_1] \leftarrow\!\!\$\ \{0,1\}^n$
	  $T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	  $T[T[\mathsf{k}][x_1]][x_2] \leftarrow T'[x_1][x_2]$
	$y \leftarrow T[\mathsf{k}][x_1]$
	If $T[y][x_2] = \bot$:
	  $T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	  $T[y][x_2] \leftarrow T'[x_1][x_2]$
	Return $T'[x_1][x_2]$

Game G_7
$T, T' \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
If $\exists x : T[\mathsf{k}][x] \neq \bot$
  $\mathsf{bad} \leftarrow \mathsf{true}$
$V := \{z : \exists (x,y) : T[x][y] = z$ or
$\exists (x,y) : T[z][y] = x\}$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\text{2-MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b

Oracle $O_\pi^2(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  If $\exists y : T[\mathsf{k}][y] = x_1$:
    If $T'[y][x_2] = \bot$:
      $T'[y][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
      If $T'[y][x_2] \in V$
        $\mathsf{bad} \leftarrow \mathsf{true}$
      $V \leftarrow V \cup \{T'[y][x_2]\}$
    $T[x_1][x_2] \leftarrow T'[y][x_2]$
  Else
    $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
    If $T[x_1][x_2] \in V$
      $\mathsf{bad} \leftarrow \mathsf{true}$
    $V \leftarrow V \cup \{T[x_1][x_2]\}$
Return $T[x_1][x_2]$

Oracle $\text{2-MDO}_\mathsf{k}^\pi(x_1, x_2)$
If $T'[x_1][x_2] \neq \bot$:
  Return $T'[x_1][x_2]$
If $T[\mathsf{k}][x_1] = \bot$:
  $T[\mathsf{k}][x_1] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T[\mathsf{k}][x_1] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T[\mathsf{k}][x_1]\}$
  $T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T'[x_1][x_2] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T'[x_1][x_2]\}$
  $T[T[\mathsf{k}][x_1]][x_2] \leftarrow T'[x_1][x_2]$
$y \leftarrow T[\mathsf{k}][x_1]$
If $T[y][x_2] = \bot$:
  $T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T'[x_1][x_2] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T'[x_1][x_2]\}$
  $T[y][x_2] \leftarrow T'[x_1][x_2]$
Return $T'[x_1][x_2]$
Oracle $O_\pi^1(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$

In game G_6, we separate the O_π oracles given to $\mathcal{A}_1$ and $\mathcal{A}_2$. The O_π oracle for $\mathcal{A}_1$ remains the same. In the O_π oracle for $\mathcal{A}_2$, we introduce a new condition if $\exists y : T[\mathsf{k}][y] = x_1$, then $T[x_1][x_2]$ is assigned $T'[y][x_2]$ if $T[x_1][x_2]$ was undefined. Notice that this does not change the distribution and hence $\Pr[\mathsf{G}_6] = \Pr[\mathsf{G}_5]$.

In game G_7, we introduce several bad events. The bad event happens if the k sampled at the beginning of the online phase is such that $(\mathsf{k}, *)$ had been queried to π by $\mathcal{A}_1$. The bad event also happens if one of sampled values of T or T' collide with the input or output of one of the offline queries or any of the other values of T, T' sampled in the online phase. Introduction of the bad events does not change behavior, so $\Pr[\mathsf{G}_7] = \Pr[\mathsf{G}_6]$.

Game G_8

$T, T' \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
If $\exists x : T[\mathsf{k}][x] \neq \bot$
  $\mathsf{bad} \leftarrow \mathsf{true}$
  ABORT
$V := \{z : \exists(x,y) : T[x][y] = z \text{ or } \exists(x,y) : T[z][y] = x\}$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\mathsf{2\text{-}MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b

Oracle $O_\pi^1(x_1, x_2)$

If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$

Oracle $O_\pi^2(x_1, x_2)$

If $T[x_1][x_2] = \bot$:
  If $\exists y : T[\mathsf{k}][y] = x_1$:
    If $T'[y][x_2] = \bot$:
      $T'[y][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
      If $T'[y][x_2] \in V$
        $\mathsf{bad} \leftarrow \mathsf{true}$
      $V \leftarrow V \cup \{T'[y][x_2]\}$
    $T[x_1][x_2] \leftarrow T'[y][x_2]$
  Else
    $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
    If $T[x_1][x_2] \in V$
      $\mathsf{bad} \leftarrow \mathsf{true}$
    $V \leftarrow V \cup \{T[x_1][x_2]\}$
  Return $T[x_1][x_2]$

Oracle $\mathsf{2\text{-}MDO}_\mathsf{k}^\pi(x_1, x_2)$

If $T'[x_1][x_2] \neq \bot$:
  Return $T'[x_1][x_2]$
$T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
If $T[\mathsf{k}][x_1] = \bot$:
  $T[\mathsf{k}][x_1] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T[\mathsf{k}][x_1] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T[\mathsf{k}][x_1]\}$
  $T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T'[x_1][x_2] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T'[x_1][x_2]\}$
  $T[T[\mathsf{k}][x_1]][x_2] \leftarrow T'[x_1][x_2]$
$y \leftarrow T[\mathsf{k}][x_1]$
If $T[y][x_2] = \bot$:
  $T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T'[x_1][x_2] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T'[x_1][x_2]\}$
  $T[y][x_2] \leftarrow T'[x_1][x_2]$
Return $T'[x_1][x_2]$

In game G_8, if the k sampled is such that the $\mathcal{A}_1$ had made a query on $(\mathsf{k}, *)$, the game aborts. Further, in 2-MDO of G_7, $T'[x_1][x_2]$ is always sampled uniformly at random if it is not defined when bad is not set – this is because if the adversary makes queries $T[\mathsf{k}][x_1] = y$ and $T[y][x_2] = z$ in order then $T'[x_1][x_2]$ will have been set to z; on the other hand if those queries happened out of order, bad would have been set. We observe that behavior in G_8 is identical to G_7 if bad is never set. This is because if bad is never set G_8 would never abort. Further, if none of the values of T, T' sampled during the online phase are same as any of the input or answer of the offline queries or other values sampled during the online phase and there is no offline query on k, whenever $T'[x_1][x_2] = \bot$, it is always sampled uniformly at random in 2-MDO is G_7, same as G_8. Moreover, the probability that bad is set in G_7 is $T_1/2^n + T_2(2T_1 + T_2)/2^n$. Therefore,

$$\Pr[\mathsf{G}_8] \leq \Pr[\mathsf{G}_7] + T_1/2^n + T_2(2T_1 + T_2)/2^n \ .$$

In game G_9, we just clean up code and remove the bad events. This does not change behavior. Hence, $\Pr[\mathsf{G}_9] = \Pr[\mathsf{G}_8]$.

Game G_9
$T, T' \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
If $\exists x : T[\mathsf{k}][x] \neq \bot$
  ABORT
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\text{2-MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi^1(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$
	Oracle $O_\pi^2(x_1, x_2)$
	If $T[x_1][x_2] = \bot$:
	  If $\exists y : T[\mathsf{k}][y] = x_1$:
	  If $T'[y][x_2] = \bot$:
	  $T'[y][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	  $T[x_1][x_2] \leftarrow T'[y][x_2]$
	  Else $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	Return $T[x_1][x_2]$
	
	Oracle $\text{2-MDO}_\mathsf{k}^\pi(x_1, x_2)$
	If $T'[x_1][x_2] \neq \bot$:
	  Return $T'[x_1][x_2]$
	$T'[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	Return $T'[x_1][x_2]$

Finally in G_{10}, we replace the lazy sampling of T' using a random oracle Π. This does not change behavior and we have $\Pr[\mathsf{G}_{10}] = \Pr[\mathsf{G}_9]$.

Game G_{10}
$T, T' \leftarrow \bot$
$\Pi \leftarrow\!\!\$\ \mathsf{Fcs}(\{0,1\}^{2m}, \{0,1\}^n)$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
If $\exists x : T[\mathsf{k}][x] \neq \bot$:
  ABORT
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\text{2-MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b
 	
Oracle $\text{2-MDO}_\mathsf{k}^\pi(x_1, x_2)$
Return $\Pi(x_1, x_2)$
	Oracle $O_\pi^1(x_1, x_2)$
	If $T[x_1][x_2] = \bot$:
	  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	Return $T[x_1][x_2]$
	
	Oracle $O_\pi^2(x_1, x_2)$
	If $T[x_1][x_2] = \bot$:
	  If $\exists y$: $T[\mathsf{k}][y] = x_1$
	  $T[x_1][x_2] \leftarrow \Pi(y, x_2)$
	  Else $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	Return $T[x_1][x_2]$

From the definition of the simulator $\mathcal{S}$ and G_{10}, it follows from inspection that

$$\Pr[\mathsf{G}_{10}] = \Pr_{\Pi \leftarrow\!\!\$\ \mathcal{I}'}\left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\mathcal{S}_1} \\ \mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n) \\ 1 \leftarrow\!\!\$\ \mathcal{A}_2^{\Pi, \mathcal{S}_2^\Pi(\tau,\mathsf{k})}(\tau, \mathsf{k}) \end{array}\right].$$

This concludes the proof. ∎

5 Offline-Online Indifferentiability of Merkle-Damgård

In this section we prove our main offline-online indifferentiability bound for the classical MD construction. Specifically, we consider the ℓ-block variant of MD, denoted ℓ-MD, where the message is of length *at most* ℓ blocks, as follows. We also assume that the input is prefix free.[5]

$$\ell\text{-MD}_\pi(\mathsf{k}, (\alpha_1, \ldots, \alpha_{\ell'})) = \pi(\ell\text{-MD}_\pi(\mathsf{k}, (\alpha_1, \ldots, \alpha_{\ell'-1})), \alpha_{\ell'}),$$

where $\ell' \leq \ell$, $\pi\colon \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$, $\ell\text{-MD}_\pi(\mathsf{k}, \alpha) = \pi(\mathsf{k}, \alpha)$, and $\mathsf{k} \in \{0,1\}^n$ and $\alpha_1, \ldots, \alpha_2\ell \in \{0,1\}^m$.

[5]Namely, no input is a prefix of another input; this can be achieved by applying a prefix-free encoding to the input before hashing it. Plain MD is known to not satisfy the classical notion of indifferenitiability (and therefore will not satisfy our more general notion), and therefore we need to make this assumption.

In the context of offline-online indifferentiability of ℓ-block MD, we define are more fine-grained notion of offline-online adversaries. Namely, we say that a (T_1, T_2)-offline-online adversary is a $(T_1, (T_{2,1}, T_{2,2}))$ adversary if it makes T_1 offline queries to its oracle, and in the online phase the cumulative number of *message blocks* it makes to its first oracle is $T_{2,1}$ and the total number of queries it makes to its second oracle is $T_{2,2}$, and $T_2 = T_{2,1} + T_{2,2}$, Our result is stated next.

Theorem 5.1. *Let $C := \ell$-MD be the ℓ-block Merkle-Damgård construction with prefix-free message encoding. Let $m, n \in \mathbb{N}$. Let $\mathcal{I}$ be the uniform distribution on $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^m, \{0,1\}^n)$. Let Samp be a sampler that takes as input a salt length n, and outputs a string uniformly at random in $\{0,1\}^n$. Let $\mathcal{I}'$ be the uniform distribution on $\mathsf{Fcs}((\{0,1\}^m)^+, \{0,1\}^n)$. Then, there exists a $(0, T_2)$ simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for every $(T_1, (T_2))$ adversary $\mathcal{A}$*

$$\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) \leq \frac{T_1}{2^n} + \frac{2T_1T_{2,1} + T_1T_{2,2}}{2^n} + \frac{2T_{2,1}^2 + T_{2,2}^2 + 3T_{2,1}T_{2,2}}{2^n} .$$

Remark 5.2. *We give a direct proof for the above theorem, following the high level idea of the proof when $\ell = 2$ in Section 4. We note, however, that an alternative route would be to apply the composition theorem (Theorem 3.2) directly, but the resulting bound would be somewhat worse (because we would need to apply a union bound over all possible lengths up to ℓ).*

Proof. Assume (T_1, T_2)-adversary $\mathcal{A}$ is a $(T_1, T_{2,1}, T_{2,2})$ adversary as defined above. Recall that

$$\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) := \left| \Pr_{\pi \leftarrow\$ \mathcal{I}} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^n) \\ 1 \leftarrow\$ \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}^{\pi}_{\mathsf{k}},\pi}(\tau, \mathsf{k}) \end{array} \right] - \Pr_{\Pi \leftarrow\$ \mathcal{I}'} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\mathcal{S}} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^n) \\ 1 \leftarrow\$ \mathcal{A}_2^{\Pi,\mathcal{S}^{\Pi}}(\tau, \mathsf{k}) \end{array} \right] \right| .$$

The initialization procedure of $\mathcal{S}_1$ simply initializes a table T to $\perp$. $\mathcal{S}_1$ answers the queries by lazily sampling its answers. The initialization procedure of $\mathcal{S}_2$ checks if there was a query made on k in the offline phase – if so, it simply aborts. To answer a query, $\mathcal{S}_2$ checks if there was a sequence of queries $(\mathsf{k}, z_1), (y_1, z_2), (y_2, z_3), \ldots, (y_{t-1}, z_{t-1})$ such that $t \leq \ell, T[\mathsf{k}][z_1] = y_1$, $\forall j \in \{2, \ldots, t-1\}$: $T[y_{j-1}][z_j] = y_j$ and $y_{t-1} = x_1$, then it uses Π, otherwise answers by lazily sampling. It follows from the description that the simulator makes no offline queries and at most $T_{2,2} \leq T_2$ online queries.

Simulator $\mathcal{S}_1$	Simulator $\mathcal{S}_2^{\Pi}(\mathsf{k}, \tau)$
Procedure Init $T \leftarrow \perp$ Procedure Query(x_1, x_2) If $T[x_1][x_2] = \perp$: $T[x_1][x_2] \leftarrow\$ \{0,1\}^n$ Return $T[x_1][x_2]$	Procedure Init If $\exists x : T[\mathsf{k}][x] \neq \perp$: **Abort** Procedure Query(x_1, x_2) If $T[x_1][x_2] = \perp$: If $\exists y_1, y_2, \ldots, y_{t-1}, z_1, z_2, \ldots, z_{t-1}$: $T[\mathsf{k}][z_1] = y_1$ and $\forall j \in \{2, \ldots, t-1\} : T[y_{j-1}][z_j] = y_j$ and $t \leq \ell$ and $z_t = x_1$ $T[x_1][x_2] \leftarrow \Pi(z_1, z_2, \ldots, z_{t-1}, x_2)$ Else $T[x_1][x_2] \leftarrow\$ \{0,1\}^n$ Return $T[x_1][x_2]$

We now prove the advantage claim for $\mathcal{S}$ using a sequence of games. Namely, we will define the games $\mathsf{G}_0, \mathsf{G}_1, \mathsf{G}_2, \mathsf{G}_3, \mathsf{G}_4, \mathsf{G}_5, \mathsf{G}_6, \mathsf{G}_7, \mathsf{G}_8, \mathsf{G}_9$ below and prove the following claims.

- $\Pr_{\pi \leftarrow\$ \mathcal{I}} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^n) \\ 1 \leftarrow\$ \mathcal{A}_2^{2\text{-}\mathsf{MDO}^{\pi}_{\mathsf{k}},\pi}(\tau, \mathsf{k}) \end{array} \right] = \Pr[\mathsf{G}_0].$

- $\Pr[\mathsf{G}_1] = \Pr[\mathsf{G}_0]$.
- $\Pr[\mathsf{G}_2] = \Pr[\mathsf{G}_1]$.
- $\Pr[\mathsf{G}_3] = \Pr[\mathsf{G}_2] + \frac{T_1 T_{2,1}}{2^n} + \frac{T_{2,1}T_{2,2}+T_{2,1}^2}{2^n}$.
- $\Pr[\mathsf{G}_4] = \Pr[\mathsf{G}_3]$
- $\Pr[\mathsf{G}_5] = \Pr[\mathsf{G}_4]$.
- $\Pr[\mathsf{G}_6] = \Pr[\mathsf{G}_5]$.
- $\Pr[\mathsf{G}_7] \leq \Pr[\mathsf{G}_6] + T_1/2^n + 2(T_{2,1}+T_{2,2})T_1/2^n + (T_{2,1}+T_{2,2})^2/2^n$.
- $\Pr[\mathsf{G}_8] = \Pr[\mathsf{G}_7]$.
- $\Pr[\mathsf{G}_9] = \Pr[\mathsf{G}_8]$.
- $\Pr[\mathsf{G}_9] = \Pr_{\Pi \leftarrow\!\$\, \mathcal{I}'}\left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\mathcal{S}_1} \\ \mathsf{k} \leftarrow\!\$\, \mathsf{Samp}(1^n) \\ 1 \leftarrow\!\$\, \mathcal{A}_2^{\Pi,\mathcal{S}_2^{\Pi}(\tau,\mathsf{k})}(\tau,\mathsf{k}) \end{array}\right]$.

Putting this together, we have that

$$\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) \leq \frac{T_1}{2^n} + \frac{2T_1T_{2,1} + T_1T_{2,2}}{2^n} + \frac{2T_{2,1}^2 + T_{2,2}^2 + 3T_{2,1}T_{2,2}}{2^n} \ .$$

Now, using $T_2 = T_{2,1} + T_{2,2}$ we get that

$$\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) \leq \frac{T_1}{2^n} + \frac{2T_1T_2}{2^n} + \frac{2T_2^2}{2^n} \ .$$

This proves the theorem. We now prove all the claims.

Define game G_0 as follows. In the game, the random oracle π is sampled, T, T' are initialized to $\perp$ everywhere. Then the adversary $\mathcal{A}_1$ is run with oracle access to π, it outputs a state τ. A salt k is sampled uniformly at random from $\{0,1\}^n$. The adversary $\mathcal{A}_2$ gets the salt as input, along with access to oracles $O_\pi, \ell\text{-}\mathsf{MDO}^\pi_\mathsf{k}$. $\mathcal{A}_2$ returns a bit which is the output of the game. The oracle O_π simulates the random oracle π with additional bookkeeping. Similarly, the oracle $\ell\text{-}\mathsf{MDO}^\pi_\mathsf{k}$ is the ℓ-block MD construction with salt k and compression function π, with additional bookkeping in T'. It follows from inspection that

$$\Pr_{\pi \leftarrow\!\$\, \mathcal{I}}\left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\!\$\, \mathsf{Samp}(1^n) \\ 1 \leftarrow\!\$\, \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}^\pi_\mathsf{k},\pi}(\tau,\mathsf{k}) \end{array}\right] = \Pr[\mathsf{G}_0] \ .$$

Game G_0	Oracle $\ell\text{-}\mathsf{MDO}^\pi_\mathsf{k}(x_1, x_2, \ldots, x_t)$
$\pi \leftarrow\!\$\, \mathcal{I}$	If $T'_t[x_1]\ldots[x_t] \neq \perp$
$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \perp$	Return $T'_t[x_1]\ldots[x_t]$
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$	$y_0 \leftarrow \mathsf{k}$
$\mathsf{k} \leftarrow\!\$\, \mathsf{Samp}(1^n)$	For $i \in \{1, \ldots, t\}$:
$b \leftarrow\!\$\, \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}^\pi_\mathsf{k}, O_\pi}(\mathsf{k}, \tau)$	If $T[y_{i-1}][x_i] = \perp$
Return b	$T[y_{i-1}][x_i] \leftarrow \pi(y_{i-1,x_i})$
Oracle $O_\pi(x_1, x_2)$	$y_i \leftarrow T[y_{i-1}][x_i]$
If $T[x_1][x_2] = \perp$:	$T'_t[x_1][x_2]\ldots[x_t] \leftarrow y_t$
$T[x_1][x_2] \leftarrow \pi(x_1, x_2)$	Return $T'_t[x_1][x_2]\ldots[x_t]$
Return $T[x_1][x_2]$	

We next move to game G_1, where the random oracle π is lazily sampled. This does not change the distribution of π, hence the output of the game. Therefore, we have that $\Pr[\mathsf{G}_1] = \Pr[\mathsf{G}_0]$.

Game G_1	Oracle $\ell\text{-MDO}^{\pi}_{\mathsf{k}}(x_1, x_2, \ldots, x_t)$

$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$
	If $T'_t[x_1][x_2]\ldots[x_t] \neq \bot$:
	  Return $T'_t[x_1][x_2]\ldots[x_t]$
	$y_0 \leftarrow \mathsf{k}$
	For $i \in \{1, \ldots, t\}$:
	  If $T[y_{i-1}][x_i] = \bot$
	  $T[y_{i-1}][x_i] \leftarrow\!\!\$\ \{0,1\}^n$
	  $y_i \leftarrow T[y_{i-1}][x_i]$
	$T'_t[x_1][x_2]\ldots[x_t] \leftarrow y_t$
	Return $T'_t[x_1][x_2]\ldots[x_t]$

We move to game G_2 where the only change compared to G_1 is during an ℓ-MDO query on $x_1, x_2, \ldots, x_t$ if $T[y_{i-1}][x_i] = \bot$, and $T[T[y_{i-1}][x_i]][x_{i+1}] \neq \bot$, a flag bad is set to true. The introduction of this flag does not alter the behavior of the game, hence $\Pr[\mathsf{G}_2] = \Pr[\mathsf{G}_1]$.

Game G_2	Oracle $\ell\text{-MDO}^{\pi}_{\mathsf{k}}(x_1, x_2, \ldots, x_t)$

$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$
	If $T'_t[x_1][x_2]\ldots[x_t] \neq \bot$:
	  Return $T'_t[x_1][x_2]\ldots[x_t]$
	$y_0 \leftarrow \mathsf{k}$
	For $i \in \{1, \ldots, t\}$:
	  If $T[y_{i-1}][x_i] = \bot$
	  $T[y_{i-1}][x_i] \leftarrow\!\!\$\ \{0,1\}^n$
	  If $T[T[y_{i-1}][x_i]][x_{i+1}] \neq \bot$
	  $\mathsf{bad} \leftarrow \mathsf{true}$
	  $y_i \leftarrow T[y_{i-1}][x_i]$
	$T'_t[x_1][x_2]\ldots[x_t] \leftarrow y_t$
	Return $T'_t[x_1][x_2]\ldots[x_t]$

In game G_3, $T[T[y_{i-1}][x_i]][x_{i+1}]$ is resampled if bad is set. Since $\mathsf{G}_2, \mathsf{G}_3$ are identical-until-bad, using the Fundamental Lemma of Game Playing [BR06], we have that

$$\Pr[\mathsf{G}_3] \leq \Pr[\mathsf{G}_2] + \Pr[\mathsf{G}_2 \text{ sets } \mathsf{bad}] \ .$$

Game G_3	Oracle $\ell\text{-MDO}^{\pi}_{\mathsf{k}}(x_1, x_2, \ldots, x_t)$

$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  Return $T[x_1][x_2]$
	If $T'_t[x_1][x_2]\ldots[x_t] \neq \bot$:
	  Return $T'_t[x_1][x_2]\ldots[x_t]$
	$y_0 \leftarrow \mathsf{k}$
	For $i \in \{1, \ldots, t\}$:
	  If $T[y_{i-1}][x_i] = \bot$
	  $T[y_{i-1}][x_i] \leftarrow\!\!\$\ \{0,1\}^n$
	  If $T[T[y_{i-1}][x_i]][x_{i+1}] \neq \bot$
	  $\mathsf{bad} \leftarrow \mathsf{true}$
	  $T[T[y_{i-1}][x_i]][x_{i+1}] \leftarrow\!\!\$\ \{0,1\}^n$
	  $y_i \leftarrow T[y_{i-1}][x_i]$
	$T'_t[x_1][x_2]\ldots[x_t] \leftarrow y_t$
	Return $T'_t[x_1][x_2]\ldots[x_t]$

Observe that bad is set in G_2 only if the oracle ℓ-MDO is invoked on $(x_1, x_2, \ldots, x_t)$ such that $T[y_{i-1}][x_i] = \perp$ and the value of $T[y_{i-1}][x_i]$ sampled uniformly at random is such that $T[T[y_{i-1}][x_i]][x_{i+1}] \neq \perp$. This can only happen if the sampled $T[y_{i-1}][x_i]$ is equal to some z such that $T[z][x_{i+1}]$ was already defined. For every ℓ-MDO query with i message blocks, there are at most i new values of T that are defined. Since the total number of message blocks queried is $T_{2,1}$, via a union bound over all ℓ-MD queries and the all the iterations in a single query we have that

$$\Pr[\mathsf{G}_2 \text{ sets } \mathsf{bad}] \leq T_{2,1}\frac{T_1 + T_{2,1} + T_{2,2}}{2^n} = \frac{T_1 T_{2,1}}{2^n} + \frac{T_{2,1}T_{2,2} + T_{2,1}^2}{2^n} .$$

Hence,

$$\Pr[\mathsf{G}_3] \leq \Pr[\mathsf{G}_2] + \frac{T_1 T_{2,1}}{2^n} + \frac{T_{2,1}T_{2,2} + T_{2,1}^2}{2^n} .$$

In game G_4, we remove the bad flag, and after the check $T[y_{i-1}][x_i] = \perp$, we remove the check whether $T[T[y_{i-1}][x_i]][x_{i+1}] \neq \perp$. Note that this does not change behavior because in game G_3, it was resampled here if $T[T[y_{i-1}][x_i]][x_{i+1}] \neq \perp$ and sampled in the next iteration if the check failed – now it is just always sampled in the next iteration Hence, $\Pr[\mathsf{G}_4] = \Pr[\mathsf{G}_3]$.

Game G_4	Oracle ℓ-$\mathsf{MDO}^{\pi}_{\mathsf{k}}(x_1, x_2, \ldots, x_t)$
$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \perp$	If $T'_t[x_1][x_2]\ldots[x_t] \neq \perp$:
$\tau \leftarrow \mathcal{A}_1^{O_\pi}$	Return $T'_t[x_1][x_2]\ldots[x_t]$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$	$y_0 \leftarrow \mathsf{k}$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}^{\pi}_{\mathsf{k}}, O_\pi}(\mathsf{k}, \tau)$	For $i \in \{1, \ldots, t\}$:
Return b	If $T[y_{i-1}][x_i] = \perp$
	$T[y_{i-1}][x_i] \leftarrow\!\!\$\ \{0,1\}^n$
Oracle $O_\pi(x_1, x_2)$	$y_i \leftarrow T[y_{i-1}][x_i]$
If $T[x_1][x_2] = \perp$:	$T'_t[x_1][x_2]\ldots[x_t] \leftarrow y_t$
$T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$	Return $T'_t[x_1][x_2]\ldots[x_t]$
Return $T[x_1][x_2]$	

In game G_5, we separate the O_π oracles given to $\mathcal{A}_1$ and $\mathcal{A}_2$. The O_π oracle for $\mathcal{A}_1$ remains the same. In the O_π oracle for $\mathcal{A}_2$, we introduce a new condition $\exists y_1, y_2, \ldots, y_{t-1}, z_1, z_2, \ldots, z_{t-1} : T[\mathsf{k}][z_1] = y_1$ and $\forall j \in \{2, \ldots, t-1\} : T[y_{j-1}][z_j] = y_j$ and $t \leq \ell$ and $z_t = x_1$ if $T[x_1][x_2]$ was undefined. Observe that because we use prefix-free MD, at most one set of y_i's will satisfy the stated condition. Notice that this does not change the distribution and hence $\Pr[\mathsf{G}_5] = \Pr[\mathsf{G}_4]$.

Game G_5	Oracle ℓ-$\mathsf{MDO}^{\pi}_{\mathsf{k}}(x_1, x_2, \ldots, x_t)$
$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \perp$	If $T'_t[x_1][x_2]\ldots[x_t] \neq \perp$:
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$	Return $T'_t[x_1][x_2]\ldots[x_t]$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$	$y_0 \leftarrow \mathsf{k}$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}^{\pi}_{\mathsf{k}}, O_\pi^2}(\mathsf{k}, \tau)$	For $i \in \{1, \ldots, t\}$:
Return b	If $T[y_{i-1}][x_i] = \perp$
	$T[y_{i-1}][x_i] \leftarrow\!\!\$\ \{0,1\}^n$
Oracle $O_\pi^1(x_1, x_2)$	$y_i \leftarrow T[y_{i-1}][x_i]$
If $T[x_1][x_2] = \perp$:	$T'_t[x_1][x_2]\ldots[x_t] \leftarrow y_t$
$T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$	Return $T'_t[x_1][x_2]\ldots[x_t]$
Return $T[x_1][x_2]$	

In game G_6, we introduce several bad events. The bad event happens if the k sampled at the beginning of the online phase is such that $(\mathsf{k}, *)$ had been queried to π by $\mathcal{A}_1$. The bad event also

happens if one of sampled values of T or T' collide with the input or output of one of the offline queries or any of the other values of T, T' sampled in the online phase. Introduction of the bad events does not change behavior, so $\Pr[\mathsf{G}_6] = \Pr[\mathsf{G}_5]$.

Game G_6
$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \perp$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
If $\exists x : T[\mathsf{k}][x] \neq \perp$
  $\mathsf{bad} \leftarrow \mathsf{true}$
$V := \{z : \exists (x,y) : T[x][y] = z$ or
$\exists (x,y) : T[z][y] = x\}$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b

Oracle $O_\pi^1(x_1, x_2)$
If $T[x_1][x_2] = \perp$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$

Oracle $O_\pi^2(x_1, x_2)$
If $T[x_1][x_2] = \perp$:
  If $\exists y_1, y_2, \ldots, y_{t-1}, z_1, z_2, \ldots, z_{t-1} : T[\mathsf{k}][z_1] = y_1$ and $\forall j \in \{2, \ldots, t-1\} : T[y_{j-1}][z_j] = y_j$ and $t \leq \ell$ and $z_t = x_1$
    If $T'_t[z_1][z_2]\ldots[z_{t-1}][x_2] = \perp$
      $T'_t[z_1][z_2]\ldots[z_{t-1}][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
      If $T'_t[z_1][z_2]\ldots[z_{t-1}][x_2] \in V$
        $\mathsf{bad} \leftarrow \mathsf{true}$
      $V \leftarrow V \cup \{T'_t[z_1][z_2]\ldots[z_{t-1}][x_2]\}$
    $T[x_1][x_2] \leftarrow T'_t[z_1][z_2]\ldots[z_{t-1}][x_2]$
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T[x_1][x_2] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T[x_1][x_2]\}$
Return $T[x_1][x_2]$

Oracle $\ell\text{-}\mathsf{MDO}_\mathsf{k}^\pi(x_1, x_2, \ldots, x_t)$
If $T'_t[x_1][x_2]\ldots[x_t] \neq \perp$:
  Return $T'_t[x_1][x_2]\ldots[x_t]$
$y_0 \leftarrow \mathsf{k}$
For $i \in \{1, \ldots, t\}$:
  If $T[y_{i-1}][x_i] = \perp$
    $T[y_{i-1}][x_i] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T[y_{i-1}][x_i] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T[y_{i-1}][x_i]\}$
  $y_i \leftarrow T[y_{i-1}][x_i]$
$T'_t[x_1][x_2]\ldots[x_t] \leftarrow y_t$
Return $T'_t[x_1][x_2]\ldots[x_t]$

In game G_7, if the k sampled is such that the $\mathcal{A}_1$ had made a query on $(\mathsf{k}, *)$, the game aborts. Further, in ℓ-MDO, $T'[x_1][x_2]\ldots[x_t]$ is always sampled uniformly at random if it is not defined. We argue that the behavior in G_7 is identical to G_6 if bad is never set. In G_6, if the ℓ-MDO oracle is queried on inputs such that no input is a prefix of another, then the bad event happens only if a freshly sampled T value collides with a T' or T value sampled earlier. As long as no two intermediate or resultant evaluations of ℓ-MDO queries and the answer of O queries have the same answer, bad is never set in game G_6 and therefore, the transition to game G_7 does not change behavior. The guarantee that two intermediate or resultant evaluations of ℓ-MDO queries and the answer of O queries have the same answer holds assuming we use prefix-free MD. Moreover, the probability that bad is set in G_6 is $T_1/2^n + (T_{2,1} + T_{2,2})(T_1 + T_{2,1} + T_{2,2})/2^n$. Therefore,

$$\Pr[\mathsf{G}_7] \leq \Pr[\mathsf{G}_6] + T_1/2^n + 2(T_{2,1} + T_{2,2})T_1/2^n + (T_{2,1} + T_{2,2})^2/2^n \ .$$

Game G_7

$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
If $\exists x : T[\mathsf{k}][x] \neq \bot$
  $\mathsf{bad} \leftarrow \mathsf{true}$
  ABORT
$V := \{z : \exists (x,y) : T[x][y] = z \text{ or } \exists (x,y) : T[z][y] = x\}$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b

Oracle $O_\pi^1(x_1, x_2)$

If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$

Oracle $O_\pi^2(x_1, x_2)$

If $T[x_1][x_2] = \bot$:
  If $\exists y_1, y_2, \ldots, y_{t-1}, z_1, z_2, \ldots, z_{t-1} : T[\mathsf{k}][z_1] = y_1$ and $\forall j \in \{2, \ldots, t-1\} : T[y_{j-1}][z_j] = y_j$ and $t \leq \ell$ and $z_t = x_1$
    If $T'_t[z_1][z_2]\ldots[z_{t-1}][x_2] = \bot$
      $T'_t[z_1][z_2]\ldots[z_{t-1}][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
      If $T'_t[z_1][z_2]\ldots[z_{t-1}][x_2] \in V$
        $\mathsf{bad} \leftarrow \mathsf{true}$
      $V \leftarrow V \cup \{T'_t[z_1][z_2]\ldots[z_{t-1}][x_2]\}$
    $T[x_1][x_2] \leftarrow T'_t[z_1][z_2]\ldots[z_{t-1}][x_2]$
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T[x_1][x_2] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T[x_1][x_2]\}$
Return $T[x_1][x_2]$

Oracle $\ell\text{-}\mathsf{MDO}_\mathsf{k}^\pi(x_1, x_2, \ldots, x_t)$

If $T'_t[x_1][x_2]\ldots[x_t] \neq \bot$:
  Return $T'_t[x_1][x_2]\ldots[x_t]$
$T'_t[x_1][x_2]\ldots[x_t] \leftarrow\!\!\$\ \{0,1\}^n$
$y_0 \leftarrow \mathsf{k}$
For $i \in \{1, \ldots, t\}$:
  If $T[y_{i-1}][x_i] = \bot$
    $T[y_{i-1}][x_i] \leftarrow\!\!\$\ \{0,1\}^n$
  If $T[y_{i-1}][x_i] \in V$
    $\mathsf{bad} \leftarrow \mathsf{true}$
  $V \leftarrow V \cup \{T[y_{i-1}][x_i]\}$
  $y_i \leftarrow T[y_{i-1}][x_i]$
$T'_t[x_1][x_2]\ldots[x_t] \leftarrow y_t$
$T[y_{t-1}][x_t] \leftarrow T'_t[x_1][x_2]\ldots[x_t]$
Return $T'_t[x_1][x_2]\ldots[x_t]$

In game G_8, we just clean up code and remove the bad events. This does not change behavior. Hence, $\Pr[\mathsf{G}_8] = \Pr[\mathsf{G}_7]$.

Game G_8

$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
If $\exists x : T[\mathsf{k}][x] \neq \bot$
  $\mathsf{bad} \leftarrow \mathsf{true}$
  ABORT
$V := \{z : \exists (x,y) : T[x][y] = z \text{ or } \exists (x,y) : T[z][y] = x\}$
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b

Oracle $O_\pi^1(x_1, x_2)$

If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$

Oracle $O_\pi^2(x_1, x_2)$

If $T[x_1][x_2] = \bot$:
  If $\exists y_1, y_2, \ldots, y_{t-1}, z_1, z_2, \ldots, z_{t-1} : T[\mathsf{k}][z_1] = y_1$ and $\forall j \in \{2, \ldots, t-1\} : T[y_{j-1}][z_j] = y_j$ and $t \leq \ell$ and $z_t = x_1$
    If $T'_t[z_1][z_2]\ldots[z_{t-1}][x_2] = \bot$
      $T'_t[z_1][z_2]\ldots[z_{t-1}][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
    $T[x_1][x_2] \leftarrow T'_t[z_1][z_2]\ldots[z_{t-1}][x_2]$
  Else $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$

Oracle $\ell\text{-}\mathsf{MDO}_\mathsf{k}^\pi(x_1, x_2, \ldots, x_t)$

If $T'_t[x_1][x_2]\ldots[x_t] \neq \bot$:
  Return $T'_t[x_1][x_2]\ldots[x_t]$
$T'_t[x_1][x_2]\ldots[x_t] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T'_t[x_1][x_2]\ldots[x_t]$

Finally, in G_9, we replace the lazy sampling of T' using a random oracle Π. This does not change behavior and we have $\Pr[\mathsf{G}_9] = \Pr[\mathsf{G}_8]$.

Game G_9
$T, T'_1, T'_2, \ldots, T'_\ell \leftarrow \bot$
$\tau \leftarrow \mathcal{A}_1^{O_\pi^1}$
$\mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n)$
If $\exists y : T[\mathsf{k}][y] \neq \bot$
  $\mathsf{bad} \leftarrow \mathsf{true}$
  ABORT
$b \leftarrow\!\!\$\ \mathcal{A}_2^{\ell\text{-}\mathsf{MDO}_\mathsf{k}^\pi, O_\pi^2}(\mathsf{k}, \tau)$
Return b
 	
Oracle $O_\pi^1(x_1, x_2)$
If $T[x_1][x_2] = \bot$:
  $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
Return $T[x_1][x_2]$
	Oracle $O_\pi^2(x_1, x_2)$
	If $T[x_1][x_2] = \bot$:
	  If $y_1, \ldots, y_{t-1}, z_1, \ldots, z_{t-1} : t \leq \ell$ and $T[\mathsf{k}][z_1] = y_1$ and $T[y_1][z_2] = y_2, \ldots, T[y_{t-2}][z_{t-1}] = y_{t-1}$ and $y_{t-1} = x_1$
	  $T[x_1][x_2] \leftarrow \Pi(z_1, \ldots, z_{t-1}, x_2)$
	  Else $T[x_1][x_2] \leftarrow\!\!\$\ \{0,1\}^n$
	Return $T[x_1][x_2]$
	
	Oracle $\ell\text{-}\mathsf{MDO}_\mathsf{k}^\pi(x_1, x_2, \ldots, x_t)$
	Return $\Pi(x_1, x_2, \ldots, x_t)$

From the definition of the simulator $\mathcal{S}$ and G_9, it follows from inspection that

$$\Pr[\mathsf{G}_9] = \Pr_{\Pi \leftarrow\!\!\$\ \mathcal{I}'}\left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\mathcal{S}_1} \\ \mathsf{k} \leftarrow\!\!\$\ \mathsf{Samp}(1^n) \\ 1 \leftarrow\!\!\$\ \mathcal{A}_2^{\Pi, \mathcal{S}_2^\Pi(\tau,\mathsf{k})}(\tau, \mathsf{k}) \end{array}\right].$$

This concludes the proof. ∎

5.1 Offline-Online Indifferentiability of Keyed Merkle-Damgård

The keyed-MD construction is defined relative to a compressing function $\pi\colon \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$, and is defined as

$$\mathsf{keyed\text{-}MD}_\pi(\mathsf{k}, (\alpha_1, \ldots, \alpha_\ell)) = \pi(\mathsf{k}, \mathsf{keyed\text{-}MD}_\pi(\mathsf{k}, (\alpha_1, \ldots, \alpha_{\ell-1})), \alpha_\ell),$$

where $\mathsf{keyed\text{-}MD}_\pi(\mathsf{k}, \alpha) = \pi(\mathsf{k}, 0^n, \alpha)$, and $\mathsf{k} \in \{0,1\}^n$ and $\alpha, \alpha_1, \ldots, \alpha_\ell \in \{0,1\}^m$.

Theorem 5.3. *Let $C :=$ keyed-MD be the ℓ-block keyed Merkle-Damgård construction. Let $m, n \in \mathbb{N}$. Let $\mathcal{I}$ be the uniform distribution on $\mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^n \times \{0,1\}^m, \{0,1\}^n)$. Let Samp be a sampler that takes as input a salt length n, and outputs a string uniformly at random in $\{0,1\}^n$. Let $\mathcal{I}'$ be the uniform distribution on $\mathsf{Fcs}((\{0,1\}^m)^+, \{0,1\}^n)$. Then, there exists a $(0, T_2)$-simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for every (T_1, T_2)-adversary $\mathcal{A}$*

$$\mathsf{Adv}^{\mathcal{I}'}_{C,\mathsf{Samp},\mathcal{I},\mathcal{S}}(\mathcal{A}) \leq \frac{T_1}{2^n} + \frac{2T_1T_2}{2^{2n}} + \frac{2T_2^2}{2^n}.$$

Proof. [Sketch] The proof of this theorem is almost identical to that of the proof of Theorem 5.1, with nearly identical hybrids, where the only difference being the analysis in the transition between G_2 and G_3. In the probability calculation that $\Pr[\mathsf{G}_2$ sets $\mathsf{bad}]$, we have that the expected number of offline queries on $(\mathsf{k}, *)$ is $T_1/2^n$, so the term $T_1T_2/2^n$ in the bound is replaced by $T_1T_2/2^{2n}$ (which is better). ∎

6 Tightness of Offline-Online Indifferentiability Bounds

In this section, we demonstrate the tightness of the terms in our bounds proved in Theorems 4.1, 5.1 and 5.3.

Remark 6.1 (Tightness of Theorem 4.1.)**.** *In Theorem 4.1, for $T_2 \geq 1$, the term $T_1T_2/2^n$ always dominates $T_1/2^n$. We give an attack in Theorem 6.2 that achieves the advantage roughly $T_1T_2/2^n - T/2^n$ for any $(0,T)$ simulator. Now, for the simulator to be meaningful, $T = O(T_2)$ since otherwise the resulting online-only adversary $\mathcal{B}$ in the composition theorem will make online queries much larger than T_2, thereby weakening the guarantee. Therefore, the attack achieves advantage roughly $T_1T_2/2^n$ against any meaning, meaning that term in our bound is tight. Further, for $T_2 \geq T_1$, a birthday attack for finding collisions achieves advantage $T_2^2/2^n$, matching the bound.*

Theorem 6.2. *Let C be the two-block MD construction. Let $\mathcal{I} = \mathsf{Fcs}(\{0,1\}^n \times \{0,1\}^n \times \{0,1\}^m, \{0,1\}^n)$ and $\mathcal{I}' = \mathsf{Fcs}((\{0,1\}^m)^+, \{0,1\}^n)$. Let Samp be the sampler that samples a uniformly random bit string in $\{0,1\}^n$. There is a (T_1,T_2)-adversary $\mathcal{A}$ such that for any $(0,T)$ simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ we have that*

$$\left| \Pr_{\pi \leftarrow\$ \mathcal{I}} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\pi} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^\lambda) \\ 1 \leftarrow\$ \mathcal{A}_2^{C_{\mathsf{k}}^{\pi},\pi}(\tau, \mathsf{k}) \end{array} \right] - \Pr_{\Pi \leftarrow\$ \mathcal{I}'} \left[\begin{array}{c} \tau \leftarrow \mathcal{A}_1^{\mathcal{S}_1} \\ \mathsf{k} \leftarrow\$ \mathsf{Samp}(1^\lambda) \\ 1 \leftarrow\$ \mathcal{A}_2^{\Pi,\mathcal{S}_2^{\Pi}(\tau,\mathsf{k})}(\tau, \mathsf{k}) \end{array} \right] \right| \geq \frac{T_1T_2}{2^n} - \frac{T}{2^n} \,.$$

Proof. The adversary in $\mathcal{A}_1$ makes T_1 queries on distinct salts to π. The adversary $\mathcal{A}_2$, on input k, makes distinct queries of the form $\pi(\mathsf{k}, *)$. If the output of a query (k, b) is z such that (z, y) was queried by $\mathcal{A}_1$ with answer w (for some y), the adversary queries $C_{\mathsf{k}}^{\pi}(b, y)$ and returns 1 if the answer is w and 0 otherwise.

First off it is easy to see that when interacting with $\pi, C_{\mathsf{k}}^{\pi}$, the probability of the adversary $\mathcal{A}_2$ outputting 1 is at least $T_1T_2/2^n$. This is because for every (z, y) that was queried by $\mathcal{A}_1$, the $\mathcal{A}_2$ succeeds in making a query (k, b) which has answer z with probability $T_2/2^n$. Since all the queries by $\mathcal{A}_1$ are on distinct salts, the probability that $\mathcal{A}_2$ makes a query (k, b) with answer z such that (z, y) was queried by $\mathcal{A}_1$ is $T_1T_2/2^n$. Finally if the answer of the query on (z, y) was w, by definition of two-block MD, $C_{\mathsf{k}}^{\pi}(b, y)$ will return w, and $\mathcal{A}_2$ returns 1.

Now, consider the case when $\mathcal{A}_1$ interacts with $\mathcal{S}_1$ and $\mathcal{A}_2$ interacts with $\mathcal{S}_2^{\Pi}$. If say $\mathcal{A}_2$ makes a query (k, b) which is answered with z such that (z, y) was queried by $\mathcal{A}_1$ with answer w: the probability that $H(b, y) = w$ is at most $T/2^n$ because $\mathcal{S}_1$ which answered with w, is independent of Π and $\mathcal{S}_2$ can make at most T queries. Therefore, the probability that the adversary outputs 1 is at most $T/2^n$. Hence, the claimed advantage bound follows. ■

Remark 6.3 (Tightness of Theorem 5.1.)**.** *In Theorem 5.1, the term $T_1T_2/2^n$ always dominates $T_1/2^n$. We give an attack in Theorem 6.2 that achieves advantage roughly $T_1T_2/2^n$ for meaningful simulators. This shows that the term $2T_1T_2/2^n$ is tight upto constant factors. Further, a birthday attack for finding collisions achieves advantage $T_2^2/2^n$, matching the last term bound when T_2 is defined as the total number of message blocks queried to the oracles by $\mathcal{A}_2$. Therefore, this bound is tight.*

Remark 6.4 (Tightness of Theorem 5.3.)**.** *In Theorem 5.3, the term $T_1/2^n$ would dominate $T_1T_2/2^{2n}$ because for $T_2 \geq 2^{n/2}$ the bound is trivial. An attack with T_1 offline queries and one online query would simply make distinct queries on T_1 different salts in the offline phase. If it makes a query on the salt k that is sampled, the single online query to the construction would be*

enough to distinguish the real and ideal worlds because the simulator in the ideal world has no access to Π *in the offline phase. This attack has advantage* $T_1/2^n$ *and can be applied to any salted construction. Further, a birthday attack for finding collisions achieves advantage* $T_2^2/2^n$*, matching the last term bound when* T_2 *is defined as the total number of message blocks queried to the oracles by* $\mathcal{A}_2$*. Therefore, this bound is tight.*

References

[ABD+13] Elena Andreeva, Andrey Bogdanov, Yevgeniy Dodis, Bart Mennink, and John P. Steinberger. On the indifferentiability of key-alternating ciphers. In *CRYPTO*, pages 531–550, 2013. 2

[ACDW20] Akshima, David Cash, Andrew Drucker, and Hoeteck Wee. Time-space tradeoffs and short collisions in merkle-damgård hash functions. In *CRYPTO*, pages 157–186, 2020. 2

[AGL22] Akshima, Siyao Guo, and Qipeng Liu. Time-space lower bounds for finding collisions in merkle-damgård hash functions. In *CRYPTO*, pages 192–221, 2022. 2

[BDPA08] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. On the indifferentiability of the sponge construction. In *EUROCRYPT*, pages 181–197, 2008. 2

[BPW04] Michael Backes, Birgit Pfitzmann, and Michael Waidner. A general composition theorem for secure reactive systems. In *TCC*, pages 336–354, 2004. 2

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS*, pages 62–73. ACM, 1993. 2

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 409–426. Springer, 2006. 10, 17, 24

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145. IEEE Computer Society, 2001. 2

[Can20] Ran Canetti. Universally composable security. *J. ACM*, 67(5):28:1–28:94, 2020. 2

[CDG18] Sandro Coretti, Yevgeniy Dodis, and Siyao Guo. Non-uniform bounds in the random-permutation, ideal-cipher, and generic-group models. In *CRYPTO*, pages 693–721, 2018. 2

[CDGS18] Sandro Coretti, Yevgeniy Dodis, Siyao Guo, and John P. Steinberger. Random oracles and non-uniformity. In *EUROCRYPT*, pages 227–258, 2018. 2, 7

[CDMP05] Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-damgård revisited: How to construct a hash function. In *CRYPTO*, pages 430–448, 2005. 2

[CPS08] Jean-Sébastien Coron, Jacques Patarin, and Yannick Seurin. The random oracle model and the ideal cipher model are equivalent. In *CRYPTO*, pages 1–20, 2008. 2

[DFMT20] Yevgeniy Dodis, Pooya Farshim, Sogol Mazaheri, and Stefano Tessaro. Towards defeating backdoored random oracles: Indifferentiability with bounded adaptivity. In *TCC*, pages 241–273, 2020. 3, 7, 12

[DGHM13] Grégory Demay, Peter Gazi, Martin Hirt, and Ueli Maurer. Resource-restricted indifferentiability. In *EUROCRYPT*, pages 664–683, 2013. 3, 7, 12

[DGK17] Yevgeniy Dodis, Siyao Guo, and Jonathan Katz. Fixing cracks in the concrete: Random oracles with auxiliary input, revisited. In *EUROCRYPT*, pages 473–495, 2017. 2, 3

[DSSL16] Yevgeniy Dodis, Martijn Stam, John P. Steinberger, and Tianren Liu. Indifferentiability of confusion-diffusion networks. In *EUROCRYPT*, pages 679–704, 2016. 2

[FGK22] Cody Freitag, Ashrujit Ghoshal, and Ilan Komargodski. Time-space tradeoffs for sponge hashing: Attacks and limitations for short collisions. In *CRYPTO*, pages 131–160, 2022. 2

[FGK23] Cody Freitag, Ashrujit Ghoshal, and Ilan Komargodski. Optimal security for keyed hash functions: Avoiding time-space tradeoffs for finding collisions. In *EUROCRYPT*, pages 440–469, 2023. 2, 6

[FN99] Amos Fiat and Moni Naor. Rigorous time/space trade-offs for inverting functions. *SIAM J. Comput.*, 29(3):790–803, 1999. 2

[GB08] Shafi Goldwasser and Mihir Bellare. Lecture notes on cryptography, 2008. 6

[GK23] Ashrujit Ghoshal and Ilan Komargodski. On time-space tradeoffs for bounded-length collisions in merkle-damgård hashing. *Comput. Complex.*, 32(2):9, 2023. 2

[GLLZ21] Siyao Guo, Qian Li, Qipeng Liu, and Jiapeng Zhang. Unifying presampling via concentration bounds. In *TCC*, pages 177–208, 2021. 2

[GT23] Ashrujit Ghoshal and Stefano Tessaro. The query-complexity of preprocessing attacks. In *CRYPTO*, pages 482–513, 2023. 3, 4, 5, 10, 11

[Hel80] Martin E. Hellman. A cryptanalytic time-memory trade-off. *IEEE Trans. Inf. Theory*, 26(4):401–406, 1980. 2

[HKT11] Thomas Holenstein, Robin Künzler, and Stefano Tessaro. The equivalence of the random oracle model and the ideal cipher model, revisited. In *STOC*, pages 89–98, 2011. 2

[LAMP12] Atul Luykx, Elena Andreeva, Bart Mennink, and Bart Preneel. Impossibility results for indifferentiability with resets. *IACR Cryptol. ePrint Arch.*, page 644, 2012. 3

[Mit14] Arno Mittelbach. Salvaging indifferentiability in a multi-stage setting. In *EUROCRYPT*, pages 603–621, 2014. 3, 7, 12

[MPS12] Avradip Mandal, Jacques Patarin, and Yannick Seurin. On the public indifferentiability and correlation intractability of the 6-round feistel construction. In *Theory of Cryptography: 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings 9*, pages 285–302. Springer, 2012. 3, 7, 12

[MRH04] Ueli M. Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In *TCC*, pages 21–39, 2004. 2, 3, 4, 11

[RSS11] Thomas Ristenpart, Hovav Shacham, and Thomas Shrimpton. Careful with composition: Limitations of the indifferentiability framework. In *EUROCRYPT*, pages 487–506, 2011. 2, 3, 12

[Unr07] Dominique Unruh. Random oracles and auxiliary input. In *CRYPTO*, pages 205–223, 2007. 2