Received: 3 December 2020 | Revised: 19 February 2021 | Accepted: 18 March 2021

IET Information Security

DOI: 10.1049/ise2.12025

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH PAPER

Indifferentiable hash functions in the standard model

Juha Partala

Center for Machine Vision and Signal Analysis, University of Oulu, Finland

Correspondence

Juha Partala, Center for Machine Vision and Signal Analysis, University of Oulu, Finland.
Email: firstname.lastname@oulu.fi

Funding information

Academy of Finland, Grant/Award Number: 318927; Infotech Oulu, Grant/Award Number: N/A

Abstract

Indifferentiability of iterated hash functions is seen as evidence that there are no structural flaws in the iteration structure of the algorithm. However, it is often overlooked that such considerations only hold in the random oracle model and do not give any guarantee in the standard model. In this article, we show the following separation result: there is a hash function that is indifferentiable from a random oracle, but is totally insecure in the standard model. In particular, we show that it does not satisfy collision or multicollision-resistance, second preimage-resistance or preimage-resistance for any family of compression functions. Therefore, at least in theory, hash function indifferentiability does not guarantee the structural integrity of the hash algorithm in the standard model. Results in the random oracle model are not affected.

1 | INTRODUCTION

The Random Oracle Model (ROM), introduced by Bellare and Rogaway [1], is the underlying security model of many efficient cryptographic schemes used in practice. It has turned out to be hard to simultaneously provide a security proof in the standard model and to maintain the efficiency of these constructions. However, it was first observed by Canetti, Goldreich and Halevi [2, 3] that there are schemes that are secure in the ROM but fail to be secure for any implementation of the random oracle (RO) using a fixed function, a function ensemble or even a restricted function ensemble. They constructed an encryption scheme and a digital signature scheme that are secure in the ROM, but insecure for any implementation of the RO using a function ensemble. The ROM is a strong model and it has since been observed that there are even tasks that are secure in the ROM, but not in the standard model [4].

While it is evident that the results of Canetti, Goldreich and Halevi apply to schemes other than encryption and digital signatures (even for length-restricted inputs [5]), it does not apply to all schemes. It is still largely unknown for which schemes the ROM is a feasible model. Separation results have been shown, for example, for the Fiat-Shamir construction [6], hybrid encryption [7], full-domain hash signatures [8, 9] and RSA-OAEP [10]. Assuming the existence of indistinguishability obfuscation, Green et al. have shown a separation result that applies to most of the natural simulation or game based security definitions [11]. Similar results can be found in [12]. Nevertheless, the ROM is still heavily applied and underlies, for example, several of the post-quantum primitives of the NIST PQC competition.

The techniques discussed above mostly concentrate on the original ROM. The indifferentiability framework, introduced by Maurer, Renner and Holenstein [13], enables proofs in the ROM to be transformed to weaker security models. In particular, hash function indifferentiability, introduced by Coron et al. [14], enables security proofs originally in the ROM to be transformed to a model involving only an ideal compression function, a finite random oracle. In this model, the RO is replaced with an iterative hash function algorithm $H = H^{\mathcal{G}}$ applying an ideal compression function oracle $\mathcal{G}$ and the hash function provably behaves like a RO. Modeling the iteration structure into the security definition enables us to argue about the lack of structural flaws in H itself. Many hash function proposals, such as SHA-3 (Keccak) [15], have been shown to be indifferentiable from a RO (see e.g. [16–23]). It is a prevalent opinion in the scientific literature that hash function indifferentiability captures the behavioral properties of a RO for single-stage games [24]. However, one has to be careful when applying the composition result of [13], since stronger notions are required for multi-stage games [25, 26].

Since, there is a separation result between a RO and a hash function, an analogous question can be posed regarding a finite RO and an iterative hash algorithm: if the hash function is indifferentiable from the RO (in the ideal compression function model), is it structurally sound in the standard model

when the ideal compression function is exchanged with a family of secure compression functions? Even though there are results that seem to indicate otherwise [27, 28], the question has not been thoroughly addressed thus far. It was shown by Bellare and Ristenpart that there is a secure compression function such that an indifferentiable hash function fails to satisfy collision-resistance in the standard model when the ideal compression function is replaced by the function. However, families of compression functions or other security properties such as preimage-resistance were not considered. Fleischmann, Gorski and Lucks have shown that it is possible to show a hash function both secure and insecure in the indifferentiability framework depending on the level of modeling [28].

In this paper, we show that there is a hash function $\widehat{H}$ that is indifferentiable from a RO but is totally insecure for *any* implementation of the compression function family. Our reduction can be described as 'artificial' or 'contrived' in the line of [2, 3]. In this approach, an insecure scheme $\widehat{H}$ is deliberately constructed based on a concrete scheme H and the knowledge of the code implementing the RO. Proponents of the ROM often argue that 'artificial' constructs do not undermine the security of concrete schemes in practice. Therefore, we leave open the question how this separation result affects practical iterated hash functions. However, our result means that in theory indifferentiability does not guarantee any properties that are typically required from a secure hash function in the standard model. We would like to emphasize that hash function indifferentiability is nevertheless a useful property to consider. It enables proofs in the full ROM to be transformed into a model requiring only a finite RO. Our results do not contradict the applicability of hash function indifferentiability in the ROM. However, we think that one should be careful when exchanging the ideal compression function with an actual function family and to consider stronger notions of security that incorporate standard model security such as multiproperty-preservation presented in [27] in order not to be mislead into believing that a result that holds in a restricted model such as ROM holds also in the standard model.

The paper is organized as follows: Section 2 includes the preliminaries for the rest of the paper. In Section 3, we briefly describe the general idea of our approach. In Section 4, we consider a crucial difference between a finite random oracle and a family of compression functions. The main separation result is shown in Section 5. Finally, Section 6 contains the conclusions.

2 | PRELIMINARIES

We shall follow standard notations and terminology. The employed algorithms are probabilistic and polynomial-time. Security of primitives is considered in the asymptotic setting, where the advantage of an adversary is considered as a function of the security parameter λ. The length of a binary string s is denote by $|s|$. If $|s|$ is polynomial in λ, we denote $|s| = \mathsf{poly}(\lambda)$ meaning that there is a polynomial $p : \mathbb{N} \to \mathbb{N}$ such that $|s| = p(\lambda)$. If s_1, s_2 are binary strings, their concatenation is denoted by $s_1 \| s_2$. A function $\epsilon : \mathbb{N} \to \mathbb{R}$ is *negligible* if for each eventually positive polynomial $p : \mathbb{N} \to \mathbb{R}$ there exists $n_0 \in \mathbb{N}$ such that for each integer n:

$$|\epsilon(n)| < \frac{1}{p(n)},$$

whenever $n \geq n_0$. If a function $f(\lambda)$ is negligible, we denote $f(\lambda) = \mathsf{negl}(\lambda)$. Note that the sum of two negligible functions is negligible and $\mathsf{poly}(\lambda) \cdot \mathsf{negl}(\lambda) = \mathsf{negl}(\lambda)$.

The original definition of indifferentiability was given using interacting systems [13] that are connected to each other using interfaces. These interfaces can be modeled as probability ensembles that can be either private (not seen by the adversary) or public. For clarity, Maurer et al. [13] restrict themselves to systems with a single private interface $R^{priv} = \{R^{priv}_\lambda\}_{\lambda \in \mathbb{N}}$ and a single public interface $R^{pub} = \{R^{pub}_\lambda\}_{\lambda \in \mathbb{N}}$. We shall do the same. An adversary, such as a distinguisher D, can be given oracle access to these interfaces. Following the standard notation, an algorithm D with access to an oracle $\mathcal{O}$ is denoted by $\mathsf{D}^{\mathcal{O}}$.

Definition 1 (Weak-indifferentiability) $R = (R^{priv}, R^{pub})$ is weak-indifferentiable from $T = (T^{priv}, T^{pub})$ if for every distinguisher D there is a simulator S_D and a negligible function ϵ such that

$$\left|\Pr\left[\mathsf{D}^{R^{priv}_\lambda, R^{pub}_\lambda}(1^\lambda) = 1\right] - \Pr\left[\mathsf{D}^{T^{priv}_\lambda, \mathsf{S}_\mathsf{D}^{T^{pub}_\lambda}}(1^\lambda) = 1\right]\right| \leq \epsilon(\lambda),$$

where λ is the security parameter.

The connections are depicted in Figure 1.

Coron et al. apply the indifferentiability framework for iterative hash functions [14]. Their idea is to show that a particular iterative structure is sound whenever the underlying compression function is ideal. The definition of indifferentiability applied by Coron et al. is stronger than the original one.

Definition 2 (Strong indifferentiability with ideal primitives) C with oracle access to an ideal primitive $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda \in \mathbb{N}}$ is strong-indifferentiable from an ideal primitive $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ if there is a simulator S such that for every distinguisher D there is a negligible function ϵ such that

$$\left|\Pr\left[\mathsf{D}^{C^{\mathcal{G}_\lambda}, \mathcal{G}_\lambda}(1^\lambda) = 1\right] - \Pr\left[\mathsf{D}^{\mathcal{F}_\lambda, \mathsf{S}^{\mathcal{F}_\lambda}}(1^\lambda) = 1\right]\right| \leq \epsilon(\lambda),$$

where λ is the security parameter.

According to Coron et al. there has to be a universal simulator S that fools any distinguisher. In such a case, we are considering the indistinguishability of $(C^{\mathcal{G}}, \mathcal{G})$ from $(\mathcal{F}, \mathsf{S}^{\mathcal{F}})$. For the weak-indifferentiability, we only need indistinguishability of $C^{\mathcal{G}}$ from $\mathcal{F}$ and the existence of a simulator $\mathsf{S} := \mathsf{S}_\mathsf{D}$

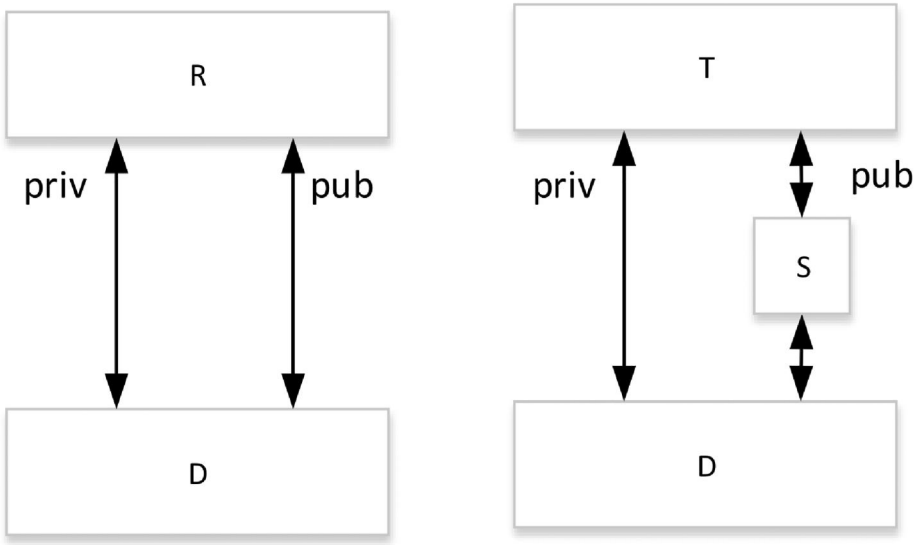

FIGURE 1 Indifferentiability. The distinguisher D cannot determine whether it is communicating with R or T (with the simulator S on the public interface of T) with non-negligible probability

for any distinguisher D. These two versions of indifferentiability have been called *weak* and *strong* in [24] and we adopt the same convention. For completeness, we give here the definition of weak-indifferentiability with ideal primitives.

Definition 3 (Weak indifferentiability with ideal primitives) C with oracle access to an ideal primitive $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda\in\mathbb{N}}$ is weak-indifferentiable from an ideal primitive $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ if for every distinguisher D there is a simulator S_D and a negligible function ϵ such that

$$|\Pr\left[\mathsf{D}^{C^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda)=1\right] - \Pr\left[\mathsf{D}^{\mathcal{F}_\lambda,\mathsf{S}_\mathsf{D}^{\mathcal{F}_\lambda}}(1^\lambda)=1\right]| \le \epsilon(\lambda),$$

where λ is the security parameter.

Considering hash functions, C will represent the hash function algorithm which turns an ideal compression function $\mathcal{G}_\lambda$ into a full hash function. When showing that the resulting hash function algorithm is indifferentiable from the random oracle, $\mathcal{F}_\lambda$ will be the RO.

The composition theorem [13] holds also for both weak and strong indifferentiability. Note, however, that the results of Coron et al. [14] regarding the differentiability of certain iterative hash functions from the RO are shown according to the definition of weak-indifferentiability.

3 | THE OUTLINE OF OUR APPROACH

In the ideal compression function model the adversary is given only oracle access to the compression function. In the standard model, the adversary has the complete specification for computing the function family that implements the ideal function. As noted in [2], the knowledge of this specification is a powerful tool. Similar to the work in [2], it enables us to devise a scheme that is secure in the ROM, but fails in the standard model for any family of compression functions. The goal is to devise a hash function $\widehat{H}$ that is indifferentiable from a random oracle but contains a trivial flaw for any compression function family. One could argue that our result follows from Proposition 4.3 of [3] adapting it to hash function, but this is not the case in a rigorous sense. Here, instead of working on the full ROM, we are working on a weaker model of finite random oracles. Such a case is explored in Section 5 of [3], but some cases are left as open problems (see Section 5.1.1 of [3]). In particular, the proof technique requires that the input to the function f_k implementing the random oracle can be sufficiently long compared to the key k. However, in the case of compression functions, it is easy to either increase the length of k or to limit the input size accordingly. Our work does not incur such a limitation provided that the lengths of the inputs and outputs are at least super-logarithmic in the length of k.

It was shown by Maurer, Renner and Holenstein that, in theory, a random oracle cannot be implemented by a finite random oracle [13]. They also formulated a simpler proof of the result of Canetti et al. for digital signatures. The proof involves the full random oracle and does not directly apply in the finite ROM. However, we adopt a very similar argumentation here adapting it to the case of indifferentiable hash functions and modify the argument to work for the finite random oracle. The outline of our approach is given as follows:

1. Show that there is an algorithm M with binary output and oracle access to either a compression function or a finite random oracle such that for any compression function f, there is an easily computable set $S \subset \{0,1\}^*$ such that $\mathsf{M}^f(s) = 1$ for every $s \in S$, but in the ideal model this happens with only negligible probability.

2. Define a new hash function $\widehat{H}$: Take an indifferentiable hash function H and for any string s output

$$\widehat{H}(s) = \begin{cases} H(s), & \text{if } \mathsf{M}(s) = 0, \\ v, & \text{if } \mathsf{M}(s) = 1, \end{cases}$$

where v is an output that leads to the complete insecurity of the hash function.

3. Show that $\widehat{H}$ is indifferentiable whenever H is.
4. Observe that, in the standard model, given any digest h in the range of $\widehat{H}$, it is easy to find multiple strings $s \in S$ such that $\widehat{H}(s) = h$ leading to complete insecurity of $\widehat{H}$.

4 | FINITE RANDOM ORACLES AND COMPRESSION FUNCTIONS

We start our investigation by describing the algorithm M. This algorithm can be seen as a way of distinguishing between the two worlds: the finite ROM and the standard model. Let

$$F = \{f^\lambda : \{0,1\}^\lambda \times \{0,1\}^{m(\lambda)} \to \{0,1\}^{n(\lambda)}\}_{\lambda \in \mathbb{N}}$$

be an efficiently computable family of compression functions, where $\{0,1\}^\lambda$ denotes the key space of F, $m, n : \mathbb{N} \to \mathbb{N}$ are strictly increasing functions and $m(\lambda) > n(\lambda)$ for every $\lambda \in \mathbb{N}$. For any key $k \in \{0,1\}^\lambda$, we denote by f_k^λ the function $f_k^\lambda(x) = f^\lambda(k, x)$. Let $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda \in \mathbb{N}}$ denote the collection of fixed length random oracles with the same domain and range as the functions of F.

We shall construct an algorithm M that will get an oracle access to a function which will either be a compression function f_k^λ or the finite random oracle $\mathcal{G}_\lambda$. M will output either 0 or 1 and satisfies the following proposition.

Proposition 1 *There is an algorithm* M *and* $\lambda' \in \mathbb{N}$ *such that for every compression function family* F, *every* $\lambda \geq \lambda'$ *and every* $k \in \{0,1\}^\lambda$ *there is a set* $S \subset \{0,1\}^*$ *for which* $|s| = \mathsf{poly}(\lambda)$ *for every* $s \in S$,

$$\mathsf{M}^{f_k^\lambda}(1^\lambda, s) = 1 \quad \text{for every } s \in S$$

and

$$\Pr\left[\exists s, |s| = \mathsf{poly}(\lambda) : \mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, s) = 1\right] \leq 2^{-n(\lambda)}.$$

Proof. Let the input s to M be interpreted as consisting of elements $v\|w\|\pi_k$, where $v, w \in \{0,1\}^{n(\lambda)}$ will be ignored at this stage but which will be used for showing the insecurity of the resulting hash function later and π_k is a description of an algorithm that computes f_k^λ. Furthermore, we assume that the description incorporates an upper bound t on the time complexity of computing π_k represented in unary.

Let $\mathcal{O}$ denote the oracle (which is either f_k^λ or $\mathcal{G}_\lambda$). On input $s = v\|w\|\pi_k$, our algorithm M will run as follows. If $|s| \leq 2n(\lambda)$, output 0. Otherwise, we set a number of queries

$$q = 1 + \left\lceil \frac{2|\pi_k|}{n(\lambda)} \right\rceil$$

The value is chosen so that we will achieve negligible probability in the ideal model. We simulate π_k for a set of outputs $r_1 = \pi_k(1)$, $r_2 = \pi_k(2)$, …, $r_q = \pi_k(q)$ each up to the maximum number of steps given by t. If π_k does not finish in t steps for some input s, then we can consider the input to be malformed and output 0. We now query the oracle $\mathcal{O}$ with the same inputs for a set of answers $r_1' = \mathcal{O}(1), r_2' = \mathcal{O}(2), \ldots, r_q' = \mathcal{O}(q)$. If $r_i = r_i'$ for every $i \in \{1, 2, \ldots, q\}$, the output is 1. Else, the output is 0.

Since, F is efficiently computable, there is an algorithm A that computes f_k^λ in time bounded from above by $t = \mathsf{poly}(\lambda)$. Let π_k^{A} denote the description of A together with the time bound t represented in unary. Since, A is a finite algorithm, there is $C \in \mathbb{N}$ such that $|\pi_k^{\mathsf{A}}| \leq C + \mathsf{poly}(\lambda)$.

Suppose first that $\mathcal{O} = f_k^\lambda$ and let us show that $\mathsf{M}^{f_k^\lambda}$ outputs 1 for the input $(1^\lambda, v\|w\|\pi_k^{\mathsf{A}})$ for every $v, w \in \{0,1\}^{n(\lambda)}$. Now, $\mathsf{M}^{f_k^\lambda}$ simulates A for at most $t = \mathsf{poly}(\lambda)$ steps. By the definition of π_k^{A} and the time bound t, this is enough to finish A on any input and $\mathsf{M}^{f_k^\lambda}$ can compute it in polynomial time with respect to its input. We have

$$\begin{aligned} r_1 &= \mathsf{A}(1) = f_k^\lambda(1), \\ r_2 &= \mathsf{A}(2) = f_k^\lambda(2), \\ &\vdots \\ r_q &= \mathsf{A}(q) = f_k^\lambda(q). \end{aligned}$$

Since, M was given oracle access to f_k^λ, we have $r_i' = r_i$ for every $i \in \{1, 2, \ldots, q\}$ and $\mathsf{M}^{f_k^\lambda}(1^\lambda, v\|w\|\pi_k^{\mathsf{A}}) = 1$.

Let now $\mathcal{O} = \mathcal{G}_\lambda$. We still need to show that, in this case, there does not exist an input s of polynomial length in λ such that $\mathsf{M}^{\mathcal{G}_\lambda}$ outputs 1 (with non-negligible probability). By the description of M, we only need to consider well-formed descriptions of algorithms that finish according to the upper bound given in unary, because otherwise $\mathsf{M}^{\mathcal{G}_\lambda}$ outputs 0. Note also that the first $2n(\lambda)$ bits of the input $s = v\|w\|\pi$ are not used by $\mathsf{M}^{\mathcal{G}_\lambda}$. Therefore, for every well-formed π, $|\pi| = \mathsf{poly}(\lambda)$, there are fixed, non-random outputs r_i for $i \in \{1, 2, \ldots, q\}$ computed by the program π. Let R'_i denote the random variable corresponding to the oracle answer r_i' for $i \in \{1, 2, \ldots, q\}$. For any π and $v, w \in \{0,1\}^{n(\lambda)}$, we have

$$\Pr\left[\mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, v\|w\|\pi) = 1\right] \leq \Pr\left[\bigcap_{i=1}^{q} R'_i = r_i\right].$$

Since, $|s| = \mathsf{poly}(\lambda)$ and subsequently $|\pi| = \mathsf{poly}(\lambda)$, for sufficiently large λ,

$$q = 1 + \left\lceil \frac{2|\pi|}{n(\lambda)} \right\rceil < 2^{m(\lambda)}$$

and the finite random oracle is queried with a different input for every query. In addition, each answer of the oracle is truly random and independent of the other queries. Therefore, the probability that for input $v\|w\|\pi$, $\mathsf{M}^{\mathcal{G}_\lambda}$ outputs one is

$$\begin{aligned}\Pr\left[\mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, v\|w\|\pi) = 1\right] &\leq \Pr\left[\bigcap_{i=1}^{q} R_i' = r_i\right] \\ &= \prod_{i=1}^{q} \Pr\left[R_i' = r_i\right] \\ &= 2^{-qn(\lambda)}.\end{aligned}$$

Note that there are at most $2^{|\pi|}$ different algorithms π (where a different upper bound t is considered to differentiate algorithms). This fact will be used below together with the union bound to get an upper bound for the probability:

$$\begin{aligned}&\Pr\left[\exists s, |s| = \mathsf{poly}(\lambda) : \mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, s) = 1\right] \\ =&\Pr\left[\bigcup_{\substack{s\in\{0,1\}^* \\ |s|=\mathsf{poly}(\lambda)}} \mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, s) = 1\right] \\ =&\Pr\left[\bigcup_{\substack{\pi\in\{0,1\}^+ \\ |\pi|=\mathsf{poly}(\lambda)}} \mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, v\|w\|\pi) = 1\right] \\ \leq& \sum_{\substack{\pi\in\{0,1\}^+ \\ |\pi|=\mathsf{poly}(\lambda)}} \Pr\left[\mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, v\|w\|\pi) = 1\right] \\ \leq& \sum_{i=1}^{\infty} \sum_{\pi\in\{0,1\}^i} \Pr\left[\mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, v\|w\|\pi) = 1\right] \\ \leq& \sum_{i=1}^{\infty} 2^i \cdot 2^{-qn(\lambda)} \leq \sum_{i=1}^{\infty} 2^i \cdot 2^{-\left(1+\frac{2i}{n(\lambda)}\right)n(\lambda)} \\ =& \sum_{i=1}^{\infty} 2^{-n(\lambda)-i} = 2^{-n(\lambda)} \cdot \sum_{i=1}^{\infty} 2^{-i} \\ =&2^{-n(\lambda)}.\end{aligned}$$

□ Note that in the standard model, the index k for the function f_k^λ is made public to anyone. This means that inputs of the type $v\|w\|\pi_k^{\mathsf{A}}$ are of polynomial length in λ and can be easily found.

5 | CONSTRUCTING A WEAK HASH FUNCTION

In this section, we show that it is possible that the iterative structure of a hash function algorithm contains a flaw in the standard model (given a specification of the compression function) even though it is shown to be secure in the ideal model (with only oracle access to the finite random oracle). In particular, given a hash function H that is indifferentiable from a random oracle, we transform H into another hash function $\widehat{H}$ that behaves correctly in the ideal compression function model, but completely breaks for any compression function family in the standard model.

We construct $\widehat{H}$ using the algorithm M. Let H be any hash function that is indifferentiable from a random oracle. Our construction works as follows.

Algorithm 1

```
1:  procedure Ĥ^O(s)
2:    b ← M^O(1^λ, s)
3:    if b = 1 then
4:      Parse s = v‖w‖π, where v, w ∈ {0,1}^n(λ)
5:      h ← v
6:    else
7:      h ← H^O(s)
8:    end if
9:    output h
10:  end procedure
```

In the ideal model, we can show that $\widehat{H}$ is secure.

Proposition 2 *Whenever H is strong-indifferentiable from a random oracle, so is $\widehat{H}$.*

Proof: Since, H is indifferentiable from the random oracle, there is a simulator S such that for every distinguisher D,

$$|\Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1\right] - \Pr\left[\mathsf{D}^{\mathcal{F}_\lambda,\mathsf{S}_\lambda^{\mathcal{F}}}(1^\lambda) = 1\right]| = \mathsf{negl}(\lambda).$$

Now, $\widehat{H}$ differs from H only when M outputs 1 for the input s. For a single distinguishing experiment using D, let the set of all queries of D to the first oracle (either $H^{\mathcal{G}_\lambda}$ or $\mathcal{F}_\lambda$) be Q. Note that for each $s \in Q$, we have $|s| = \mathsf{poly}(\lambda)$, since D is a polynomial time algorithm.

Let $D_{\mathcal{G}_\lambda}$ denote the event that there is $s \in Q$ such that $\mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, s) = 1$. Now,

$$\begin{aligned}&\Pr\left[\mathsf{D}^{\widehat{H}^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1\right] \\ &= \Pr\left[\mathsf{D}^{\widehat{H}^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 | D_{\mathcal{G}_\lambda}\right] \cdot \Pr[D_{\mathcal{G}_\lambda}] \\ &\quad + \Pr\left[\mathsf{D}^{\widehat{H}^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 | \overline{D_{\mathcal{G}_\lambda}}\right] \cdot \Pr\left[\overline{D_{\mathcal{G}_\lambda}}\right].\end{aligned}$$

By Proposition 1, we have

$$\Pr[D_{\mathcal{G}_\lambda}] \leq \Pr\left[\exists s, |s| = poly(\lambda) : \mathsf{M}^{\mathcal{G}_\lambda}(1^\lambda, s) = 1\right] \leq 2^{-n(\lambda)}$$

which is negligible. Therefore,

$$\begin{aligned}
&\Pr\left[\mathsf{D}^{\widehat{H}^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1\right] \\
&= \Pr\left[\mathsf{D}^{\widehat{H}^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 \,|\, \overline{D_{\mathcal{G}_\lambda}}\right] \cdot \Pr\left[\overline{D_{\mathcal{G}_\lambda}}\right] + \mathsf{negl}(\lambda) \\
&= \Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 \,|\, \overline{D_{\mathcal{G}_\lambda}}\right] \cdot \underbrace{\Pr\left[\overline{D_{\mathcal{G}_\lambda}}\right]}_{1-\mathsf{negl}(\lambda)} + \mathsf{negl}(\lambda) \\
&= \Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 \,|\, \overline{D_{\mathcal{G}_\lambda}}\right] + \mathsf{negl}(\lambda).
\end{aligned}$$

On the other hand, for the original hash function H, we also have

$$\begin{aligned}
&\Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1\right] \\
&= \Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 | \overline{D_{\mathcal{G}_\lambda}}\right] \cdot \underbrace{\Pr\left[\overline{D_{\mathcal{G}_\lambda}}\right]}_{1-\mathsf{negl}(\lambda)} \\
&\quad + \Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 | D_{\mathcal{G}_\lambda}\right] \cdot \underbrace{\Pr[D_{\mathcal{G}_\lambda}]}_{\mathsf{negl}(\lambda)} \\
&= \Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 | \overline{D_{\mathcal{G}_\lambda}}\right] + \mathsf{negl}(\lambda).
\end{aligned}$$

But this means that

$$\begin{aligned}
&\left|\Pr\left[\mathsf{D}^{\widehat{H}^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1\right] - \Pr\left[\mathsf{D}^{\mathcal{F}_\lambda,\mathsf{S}_\lambda^{\mathcal{F}}}(1^\lambda) = 1\right]\right| \\
&= \left|\Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1 \,|\, \overline{D_{\mathcal{G}_\lambda}}\right] + \mathsf{negl}(\lambda)\right. \\
&\qquad \left. - \Pr\left[\mathsf{D}^{\mathcal{F}_\lambda,\mathsf{S}_\lambda^{\mathcal{F}}}(1^\lambda) = 1\right]\right| \\
&= \left|\Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1\right] - \Pr\left[\mathsf{D}^{\mathcal{F}_\lambda,\mathsf{S}_\lambda^{\mathcal{F}}}(1^\lambda) = 1\right]\right. \\
&\qquad \left. + \mathsf{negl}(\lambda)\right| \\
&\leq \left|\Pr\left[\mathsf{D}^{H^{\mathcal{G}_\lambda},\mathcal{G}_\lambda}(1^\lambda) = 1\right] - \Pr\left[\mathsf{D}^{\mathcal{F}_\lambda,\mathsf{S}_\lambda^{\mathcal{F}}}(1^\lambda) = 1\right]\right| \\
&\qquad + |\,\mathsf{negl}(\lambda)\,|
\end{aligned}$$

which is negligible by the strong-indifferentiability of H. Therefore, $\widehat{H}$ is also strong-indifferentiable. □

Analogous argumentation works for the case of weak-indifferentiability. The only change needed in the proof is to allow the simulator S to be different for each distinguisher D. Therefore, we can also record the following.

Proposition 3 *Whenever H is weak-indifferentiable from a random oracle, so is $\widehat{H}$.*

In the standard model, the situation is different. Suppose that H is run with a compression function family F and suppose that the publicly chosen compression function is f_k^λ. Then for any v in the range of $\widehat{\mathsf{H}}^{f_k^\lambda}$, it is easy to find inputs $v\|w\|\pi_k^{\mathsf{A}}$ for which

$$\widehat{\mathsf{H}}^{f_k^\lambda}(v\|w\|\pi_k^{\mathsf{A}}) = v$$

leading to complete insecurity of $\widehat{H}^{f_k^\lambda}$.

Proposition 4 *There is a hash function $\widehat{H}$* that is *strong-indifferentiable from a random oracle, but is not collision or multicollision-resistant, second-preimage-resistant or preimage-resistant in the standard model for any family of compression functions.*

Proof: Let H be any hash function that is strong-indifferentiable from a random oracle. By Proposition 2, the hash algorithm $\widehat{H}$ based on H is also strong-indifferentiable from a random oracle.

Let

$$F = \{f^\lambda : \{0,1\}^\lambda \times \{0,1\}^{m(\lambda)} \to \{0,1\}^{n(\lambda)}\}_{\lambda \in \mathbb{N}}$$

be any efficiently computable compression function family, where $\{0,1\}^\lambda$ denotes the key space of F, $m, n : \mathbb{N} \to \mathbb{N}$ are strictly increasing functions and $m(\lambda) > n(\lambda)$ for every $\lambda \in \mathbb{N}$. Suppose now that a random compression function f_k^λ is chosen from the compression function family F, where $k \leftarrow U(\{0,1\}^\lambda)$. By Proposition 1, there is $\lambda' \in \mathbb{N}$ such that whenever $\lambda \geq \lambda'$ there is a set $S \subset \{0,1\}^*$ such that $|s| = \mathsf{poly}(\lambda)$ and $\mathsf{M}^{f_k^\lambda}(1^\lambda, s) = 1$ for every $s \in S$. These strings are of the type $v\|w\|\pi_k^{\mathsf{A}}$, where $v, w \in \{0,1\}^{n(\lambda)}$ are arbitrary and π_k^{A} is a description of an algorithm A that computes f_k^λ. Furthermore, since k is public, π_k^{A} can be easily computed. Consider now the following security properties:

- **Collision and multi-collision resistance.** Let $v, w, w' \in \{0,1\}^{n(\lambda)}$ be any strings such that $w \neq w'$. We have

$$\widehat{H}^{f_k^\lambda}(v\|w\|\pi_k^{\mathsf{A}}) = \widehat{H}^{f_k^\lambda}(v\|w'\|\pi_k^{\mathsf{A}}) = v$$

and the pair $(v\|w\|\pi_k^{\mathsf{A}}, v\|w'\|\pi_k^{\mathsf{A}})$ is a collision for $\widehat{H}$. In fact, $v\|w\|\pi_k^{\mathsf{A}}$ for any $w \in \{0,1\}^{n(\lambda)}$ maps to v and there are $2^{n(\lambda)} - 1$ easily computable collisions.

- **Second preimage-resistance.** Let $s \in \{0,1\}^*$ and suppose that $\widehat{H}^{f_k^\lambda}(s) = h$. Now, for any $w \in \{0,1\}^{n(\lambda)}$,

$$\widehat{H}^{f_k^{\lambda}}(h\|w\|\pi_k^{\mathsf{A}}) = h = \widehat{H}^{f_k^{\lambda}}(s)$$

and $\widehat{H}$ is vulnerable against second preimage attacks.

• **Preimage-resistance.** Let $h \in \{0,1\}^{n(\lambda)}$. It is easy to see that

$$\widehat{H}^{f_k^{\lambda}}(h\|w\|\pi_k^{\mathsf{A}}) = h$$

for any $w \in \{0,1\}^{n(\lambda)}$ and $\widehat{H}$ is vulnerable against preimage attacks.

□

6 | CONCLUSION

We show that there is a strong-indifferentiable hash function that is totally insecure for any family of compression functions in the standard model. Therefore, in theory, indifferentiability does not guarantee structural integrity of the hash function algorithm in the standard model. Results in the random oracle model are not affected.

ACKNOWLEDGEMENTS

Thanks to anonymous reviewers for valuable comments, as well as pointing out a way towards a stronger result for Proposition 4.

This research was conducted in a strategic research project called Personalization, Privacy and Quality Control for MaaS with Blockchain-TrustedMaaS under the focus institute Infotech Oulu, Faculty of Information Technology and Electrical Engineering (ITEE), and the University of Oulu, Finland. Furthermore, the research is supported by Academy of Finland 6Genesis Flagship [grant number: 318,927]. Declarations of interest: none.

ORCID

Juha Partala https://orcid.org/0000-0001-8181-5604

REFERENCES

1. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM Conference on computer and Communications security, pp. 62–73.CCS '93. ACM, New York (1993). http://doi.acm.org/10.1145/168588.168596
2. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited (preliminary version). In: Proceedings of the Thirtieth Annual ACM Symposium on theory of computing, pp. 209–218.STOC '98.ACM, New York (1998). http://doi.acm.org/10.1145/276698.276741
3. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. Acm. 51(4), 557–594 (2004). http://doi.acm.org/10.1145/1008731.1008734
4. Nielsen, J.: Separating random oracle proofs from complexity theoretic proofs: the non-committing encryption case. In: Yung, M. (ed.) Advances in Cryptology – CRYPTO 2002. Vol. 2442 of Lecture Notes in Computer Science, pp. 111–126. Santa Barbara, Springer Berlin Heidelberg (2002). http://dx.doi.org/10.1007/3-540-45708-9_8
5. Canetti, R., Goldreich, O., Halevi, S.: On the random-oracle methodology as applied to length-restricted signature schemes In: Naor, M. (ed.) Theory of Cryptography. Vol. 2951 of Lecture Notes in Computer Science, pp. 40–57. Cambridge, Springer Berlin Heidelberg (2004). http://dx.doi.org/10.1007/978-3-540-24638-1_3
6. Goldwasser, S., Kalai, Y.T.: On the (in)security of the Fiat-Shamir paradigm. In: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science., Cambridge, pp. 102–113 (2003)
7. Bellare, M., Boldyreva, A., Palacio, A.: An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In: Cachin, C., Camenisch, J. (eds.) Advances in Cryptology – EUROCRYPT 2004. Vol. 3027 of Lecture Notes in Computer Science, pp. 171–188. Springer Berlin Heidelberg (2004). http://dx.doi.org/10.1007/978-3-540-24676-3_11
8. Dodis, Y., Oliveira, R., Pietrzak, K.: On the generic insecurity of the full domain hash. In: Shoup, V. (ed.) Advances in Cryptology – CRYPTO 2005. Vol. 3621 of Lecture Notes in Computer Science pp. 449–466. Springer Berlin Heidelberg (2005). http://dx.doi.org/10.1007/11535218_27
9. Dodis, Y., Haitner, I., Tentes, A.: On the instantiability of hash-and-sign RSA signatures. In: Cramer, R. (ed.) Theory of Cryptography, pp. 112–132. Springer Berlin Heidelberg, Berlin (2012)
10. Kiltz, E., Pietrzak, K.: On the security of padding-based encryption schemes – or – why we cannot prove OAEP secure in the standard model. In: Joux, A. (ed.) Dvances in Cryptology – EUROCRYPT 2009. Vol. 5479 of Lecture Notes in Computer Science. Cologne, pp. 389–406. Springer Berlin Heidelberg (2009). http://dx.doi.org/10.1007/978-3-642-01001-9_23
11. Green, M.D., et al.: A unified approach to idealized model separations via indistinguishability obfuscation. In: Zikas, V., De.Prisco, R. (eds.) Security and Cryptography for Networks, pp. 587–603. Springer International Publishing, Cham (2016)
12. Brzuska, C., Farshim, P., Mittelbach, A.: Random-oracle uninstantiability from indistinguishability obfuscation. In: Dodis, Y., Nielsen, J.B. (eds.) Theory of Cryptography., pp. 428–455. Springer Berlin Heidelberg, Berlin (2015)
13. Maurer, U., Renner, R., Holenstein, C.: 'Indifferentiability, Impossibility results on reductions, and applications to the random oracle methodology'. In: Naor, M. (ed.) Theory of Cryptography. Vol. 2951 of Lecture Notes in Computer Science, Cambridge, pp. 21–39. Springer Berlin Heidelberg (2004). http://dx.doi.org/10.1007/978-3-540-24638-1_2
14. Coron, J.S., et al.: Merkle-Damgård revisited: how to construct a hash function. In: Shoup, V. (ed.) Advances in Cryptology – CRYPTO 2005. Vol. 3621 of Lecture Notes in Computer Science, pp. 430–448. Santa Barbara, Springer Berlin Heidelberg (2005). http://dx.doi.org/10.1007/11535218_26
15. Bertoni, G., et al.: The Keccak SHA-3 submission, Technical report (SHA-3 competition, round 3) (2011). https://keccak.team/files/Keccak-submission-3.pdf
16. Bertoni, G., et al.: On the indifferentiability of the sponge construction. In: Smart, N. (ed.) Advances in Cryptology – EUROCRYPT 2008. Vol. 4965 of Lecture Notes in Computer Science, pp. 181–197. Istanbul, Springer Berlin Heidelberg (2008). http://dx.doi.org/10.1007/978-3-540-78967-3_11
17. Hirose, S., Park, J., Yun, A.: A simple variant of the Merkle-Damgård scheme with a permutation. In: Kurosawa, K. (ed.) Advances in Cryptology – ASIACRYPT 2007. Vol. 4833 of Lecture Notes in Computer Science, Kuching, pp. 113–129.Springer Berlin Heidelberg (2007). http://dx.doi.org/10.1007/978-3-540-76900-2_7
18. Chang, D., Nandi, M.: Improved indifferentiability security analysis of chopMD hash function. In: Nyberg, K. (ed.) Fast Software encryption. Vol. 5086 of Lecture Notes in Computer Science, Lausanne, pp. 429–443. Springer Berlin Heidelberg (2008). http://dx.doi.org/10.1007/978-3-540-71039-4_27
19. Dodis, Y., et al.: Indifferentiability of permutation-based compression functions and tree-based modes of operation, with applications to MD6. In: Dunkelman, O. (ed.) Fast Software encryption. Vol. 5665 of Lecture Notes in Computer Science, Leuven, pp. 104–121.

Springer Berlin Heidelberg (2009). http://dx.doi.org/10.1007/978-3-642-03317-9_7
20. Dodis, Y., Ristenpart, T., Shrimpton, T.: Salvaging Merkle-Damgård for practical applications. In: Joux, A. (ed.) Advances in Cryptology – EUROCRYPT 2009. Vol. 5479 of Lecture Notes in Computer Science, pp. 371–388. Cologne, Springer Berlin Heidelberg (2009). http://dx.doi.org/10.1007/978-3-642-01001-9_22
21. Andreeva, E., Mennink, B., Preneel, B.: On the indifferentiability of the Grøstl hash function. In: Garay, J., De.Prisco, R. (eds.) Security and Cryptography for Networks. Vol. 6280 of Lecture Notes in Computer Science, pp. 88–105. Amalfi, Springer Berlin Heidelberg (2010). http://dx.doi.org/10.1007/978-3-642-15317-4_7
22. Bhattacharyya, R., Mandal, A.: On the indifferentiability of Fugue and Luffa. In: Lopez, J., Tsudik, G. (eds.) Applied Cryptography and Network security. Vol. 6715 of Lecture Notes in Computer Science, Nerja, pp. 479–497. Springer Berlin Heidelberg (2011). http://dx.doi.org/10.1007/978-3-642-21554-4_28
23. Dodis, Y., et al.: To hash or not to hash again? (in)differentiability results for H^2 and HMAC. In: Canetti, R. (eds.) Advances in Cryptology – CRYPTO 2012, pp. 348–366. Springer Berlin Heidelberg, Berlin (2012)
24. Ristenpart, T., Shacham, H., Shrimpton, T.: Careful with composition: Limitations of the indifferentiability framework. In: Paterson, K. (ed.) Advances in Cryptology – EUROCRYPT 2011. Vol. 6632 of Lecture Notes in Computer Science, Tallinn, pp. 487–506. Springer Berlin Heidelberg (2011). http://dx.doi.org/10.1007/978-3-642-20465-4_27
25. Bellare, M., Hoang, V., Keelveedhi, S.: Instantiating random oracles via UCEs. In: Canetti, R., Garay, J. (eds.) Advances in Cryptology – CRYPTO 2013. Vol. 8043 of Lecture Notes in Computer Science, Santa Barbara, pp. 398–415. Springer Berlin Heidelberg (2013). http://dx.doi.org/10.1007/978-3-642-40084-1_23
26. Mittelbach, A.: Salvaging indifferentiability in a multi-stage setting. In: Nguyen, P.Q., Oswald, E. (eds.) Advances in Cryptology – EUROCRYPT 2014. Vol. 8441 of Lecture Notes in Computer Science, Copenhagen, pp. 603–621.Springer Berlin Heidelberg (2014). http://dx.doi.org/10.1007/978-3-642-55220-5_33
27. Bellare, M., Ristenpart, T.: Multi-property-preserving hash domain extension and the EMD transform. In: Lai, X., Chen, K. (eds.) Advances in Cryptology – ASIACRYPT 2006, pp. 299–314. Springer Berlin Heidelberg, Berlin (2006)
28. Fleischmann, E., Gorski, M., Lucks, S.: Some observations on indifferentiability. In: Steinfeld, R., Hawkes, P. (eds.) Information security and privacy. Vol. 6168 of Lecture Notes in Computer Science, pp. 117–134. Sydney, Springer Berlin Heidelberg (2010). http://dx.doi.org/10.1007/978-3-642-14081-5_8

How to cite this article: Partala J. Indifferentiable hash functions in the standard model. *IET Inf. Secur.* 2021;15:309–316. https://doi.org/10.1049/ise2.12025