On the Security of Iterated Hashing based on Forgery-resistant Compression Functions

Charles Bouillaguet[1], Orr Dunkelman[1], Pierre-Alain Fouque[1], and Antoine Joux[2,3]

[1] École normale supérieure, Paris
{charles.bouillaguet,orr.dunkelman,pierre-alain.fouque}@ens.fr
[2] DGA, Saint-Quentin-en-Yvelines
[3] Université de Versailles Saint-Quentin
Antoine.Joux@m4x.org

Abstract. In this paper we re-examine the security notions suggested for hash functions, with an emphasis on the delicate notion of second preimage resistance. We start by showing that, in the random oracle model, both Merkle-Damgård and Haifa achieve second preimage resistance beyond the birthday bound, and actually up to the level of known generic attacks, hence demonstrating the optimality of Haifa in this respect.
We then try to distill a more elementary requirement out of the compression function to get some insight on the properties it should have to guarantee the second preimage resistance of its iteration. We show that if the (keyed) compression function is a secure FIL-MAC then the Merkle-Damgård mode of iteration (or Haifa) still maintains the same level of second preimage resistance. We conclude by showing that this "new" assumption (or security notion) implies the recently introduced Preimage-Awareness while ensuring all other classical security notions for hash functions.
Key words: hash function, security proof, MAC, second preimage, random oracle

1 Introduction

Of all major cryptographic primitives, hash functions have been continuously avoiding the theoretical nirvana other cryptographic primitives enjoy. While ciphers, encryption schemes, message authentication codes and signature schemes have well understood theoretical foundations, acknowledged security definitions, and some can be idealized using primitives which are considered natural and fair, hash functions have remained as illusive as they were.

The task of defining the security of hash functions dates back to the collision resistance preservation of the Merkle-Damgård mode of iteration [14, 24] and the works of Naor and Yung on Universal One-Way Hash Functions [27] and of Bellare and Rogaway on Target Collision Resistance [5]. In recent years, some other notions of security were suggested such as the (always/everywhere) second preimage resistance and the (always/everywhere) preimage resistance [29], the enhanced Target Collision Resistance [17], the indifferentiability from random oracle (also called Preservation of Random Oracle) [23], and most recently the Preimage Awareness (PA) [16]. The situation is all the more complex that despite the existence of a plethora of security notions, there are still intuitive results that these notions do not capture on classical modes of iteration.

We take the notion of *second preimage resistance* as the archetypal example of a problematic security notion. It can be defined in multiple ways (with or without keys, with a chosen or imposed challenge), but this is not the main issue. As far as this notion is concerned, there is a gap between the best attacks and the best proofs, and there is a gap between theory and practice.

Gap Between Attacks and Proofs. The Merkle-Damgård mode of iteration and its improvement Haifa [6] are the most practical modes of iteration, as a quick examination of the SHA-3 candidates reveals. What (provable) level resistance to second preimage attacks do these modes of iteration exhibit? To the best of our knowledge, they only achieve provable second preimage resistance up to the birthday bound, because a second preimage is in particular a collision, and both modes of iteration reduce a collision to a collision on the compression function.

We are not aware of any proof technique that would allow us to prove any kind of result beyond the birthday bound for these two main modes of iteration. For example, the recently introduced notion of PA can only be achieved up to the birthday bound because it entails collision resistance. The same goes for indifferentiability: first of all, Merkle-Damgård is *not* indifferentiable from a random oracle due to the length extension attack. Second, both modes of iteration (and also the Enveloped Merkle-Damgård [2]) are only indifferentiable up to the birthday bound, because once a single collision has been found, many additional collisions can be cooked up for free (which is not possible with a random function).

This lack of security guarantees has to be compared with the best generic attacks on these constructions: on Merkle-Damgård, and assuming that the size of both the chaining value and the hash is n bits, the best attack is that of [15, 21] which finds a second preimage of a message of size 2^k in time $\mathcal{O}\left(2^{n-k}\right)$, while there is no known attack against HAIFA besides a trivial exhaustive search.

Gap Between Theory and Practice. There was for some time a big gaping hole between theory and practice regarding second preimage resistance. Constructions of *keyed* hash functions (UOWHFs) provably achieving the eSec (a.k.a. TCR) notion in the standard model based on the hardness of a general assumption have been known for quite a while [27, 18]. Later on, schemes were designed that promote the eSec property of a compression function to the whole construction [5, 30]. Amongst the latter, Shoup's construction has an n-bit internal state and achieves $\mathcal{O}\left(2^{n-k}\right)$ second preimage resistance. This bound has recently been shown to be tight thanks to the second preimage attack of [1]. However, these schemes hardly made it to the world of practical cryptography, as keyed hash functions are rarely used.

An important progress towards closing the gap has been achieved with the double-pipe hash of [22], which is both second preimage resistant in the random oracle model (*i.e.* there are no generic attacks) and practical, since about half of the SHA-3 candidates adopted it.

This still leaves the case of *narrow-pipe*[1] Merkle-Damgård or HAIFA open. It is a legitimate question from both a theoretical and practical point of view, since this case nearly represent the other half of the SHA-3 candidates, and yet little is known about its second preimage resistance.

Related Work. After the work of Bellare and Rogaway [3], hash functions have usually been treated as random objects. Subsequently, many security proofs have been done in the relativized model of the random oracle. On the negative side, there are schemes that are secure in the random oracle model, but there is no efficient implementation of hash functions that can be used to implement them, as it has been shown by Canetti *et al.* [11]. One good news is that the construction of such schemes is rather artificial. On the positive side, the ROM allows to show that there is no structural failure in the design of the scheme. However, in practice, the random oracle is replaced by a fixed and, more importantly, *public* function which cannot be a random oracle.

An important line of research tries to avoid the use of the random oracle since the main drawback of the ROM is that, besides providing only heuristic security, we do not really know *what is the precise security property of the random oracle that we need for the security of the actual scheme.* It has been shown [8] that in some public-key schemes, hash functions with strong properties such as perfectly one-way hash function [10] and verifiable pseudorandom functions [25] can be substituted to one of the random oracles while maintaining security.

In order to construct hash functions that mimic random oracle, the indifferentiability methodology [23] has been used, and many modes of iteration have been proved secure in this framework [13, 12]. Proofs in this model show that if the compression function is a random oracle, then the iteration

[1] We call "narrow-pipe" a construction where the chaining value has the same length as the digest

behaves as a random oracle. Consequently, there are no generic attacks against any property of the iteration. A common point between the random oracle model and the indifferentiability model is that the compression function is considered as a *black-box* to which one only has oracle access. Therefore, only generic attacks, independent of the specific primitive, are taken into account in both cases.

In any case, the Merkle-Damgård construction is not indifferentiable from a random oracle. Recently, Dodis *et al.* in [16] have tried to "salvage" this mode of iteration by relaxing the indifferentiability framework to preimage awareness on one hand and to the concept of public-use random oracle on the other hand. They have made a first step in bridging the gap between the absence of attacks on Merkle-Damgård and some theoretical explanation of this fact. However, the compression function is always idealized and we do not know what is the *required assumption*, if such an assumption exists, even in the black-box model.

Our Goal and our Results. We pursue the search for a somewhat better understanding of the security of narrow-pipe Merkle-Damgård and Haifa, in particular with regard to the delicate notion of second preimage resistance. We start with proofs of security in the random oracle model. Our first contribution is to show that if the compression function is treated as a random oracle, then the second preimage resistance is $\mathcal{O}\left(2^{n-k}\right)$ for 2^k-blocks messages in Merkle-Damgård, and $\mathcal{O}\left(2^n\right)$ for Haifa. We therefore demonstrate that the existing generic second preimage attacks against Merkle-Damgård are optimal and that there is no generic second preimage attack at all against Haifa, therefore closing the gap between attacks and proofs.

We are aware that the suggested proofs do not tell the designers of practical schemes how to design a compression function to avoid second preimage attacks. We therefore examine these proofs with the hope to identify an assumption on the compression function that is simultaneously weaker that being a random oracle, and still strong enough to allow the same results to be proved.

Our second contribution is to show that if we treat the compression function as a fixed-input length MAC (FIL-MAC), *i.e.*, if it is *unforgeable*, then a Merkle-Damgård or Haifa iteration of it offers the same security level as in the random oracle model. This is the first time (to our knowledge) that a more concrete (and realizable) security property is identified to be sufficient for security. We note that this assumption still maintains some to be desired as it assumes that the MAC is secure when the key is random and unknown to the adversary, despite the fact that in practice it is "revealed" during the hash computation to everyone.

While we were trying to identify a better assumption on the compression function, we considered the recent PA notion as a natural candidate. Our last contribution is to show the existing connection between MACs and the PA notion: the Merkle-Damgård or Haifa iteration of a FIL-MAC achieves optimal PA of the hash function.

Organization of the Paper. The organization of this paper is as follows: In Section 2 we define the notations we use through the rest of the paper. In Section 3 we offer proofs of security for the Merkle-Damgård and Haifa modes of iteration in the random oracle model. We then follow in Section 4 to introduce the unforgeability assumption and offer proofs for the same security level while using this weaker assumption. The discussion concerning the Preimage-Awareness model is given in Section 5. We conclude the paper in Section 6.

2 Preliminaries and Definitions

2.1 Iterated Constructions of Hash Functions

The Merkle-Damgård mode of iteration. The Merkle-Damgård mode of iteration was independently suggested in [14, 24]. The hash function $H^F : \{0,1\}^* \to \{0,1\}^n$ is built by iterating a compression function $F\colon \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$. The hash process works as follows:

– Pad and split a message M into r blocks $x_1, \ldots, x_r$ of m bits each.
– Set h_0 to the initialization value IV .
– For each message block x_i compute $h_i = f(h_{i-1}, x_i)$.
– Output $H^F(M) = h_r$.

The padding is done usually by appending a single '1' bit followed by as many '0' bit as needed to complete a m-bit block including the length of M in bits (the well-known Merkle-Damgård strengthening).

The Haifa mode of iteration. The HAsh Iterative FrAmework (Haifa), introduced in [6], is a Merkle-Damgård construction where a bit counter and salt are added to the input of the compression function[2]. We describe an instance of Haifa with a 64-bit counter (this matches the sizes used in currently deployed hash functions). The HAsh Iterative FrAmework $H^F : \{0,1\}^* \to \{0,1\}^n$ is built by iterating a compression function $F\colon \{0,1\}^n \times \{0,1\}^m \times \{0,1\}^{64} \to \{0,1\}^n$.

The hash process works as follows:

– Pad and split a message M into r blocks $x_1, \ldots, x_r$ of m bits each.
– Set h_0 to the initialization value IV.
– For each message block x_i compute $h_i = F(h_{i-1}, x_i, i)$.
– Output $H^F(M) = h_r$.

The padding is done by appending a single '1' bit followed by as many '0' bit as needed to complete an m-bit block after the message length and the digest size are appended.

2.2 Computational Model

The proofs presented in this paper always assume *oracle access* to the compression function. When this primitive is considered as a (public) random function, the number of queries sent to the primitive can be used as a meaningful complexity measure (because the adversary cannot obtain any kind of advantage by computation alone without querying the function). Therefore, in this setting, we will consider information-theoretic (*i.e.*, computationally unbounded) adversaries, and the number of queries to the oracle will be the main measure of complexity. In any case, it gives a lower-bound on the time complexity of the adversary. This setting is very similar to the analysis of block cipher-based constructions in the ideal cipher model of [7].

We often denote by q the number of queries sent to the compression function F by an adversary $\mathcal{A}$. For the sake of convenience, we enforce adversaries not to abort, as well as collision and (second) preimage adversaries to always return a message M, even if they fail the security game, and to evaluate $H^F(M)$ before terminating, by issuing the corresponding queries to the compression function. We also enforce adversaries not to ask the same query twice.

2.3 Security Properties

Unforgeability. The security of a Fixed-Input-Length MAC (FIL-MAC) is usually measured via the existential forgery game under *chosen message attack*. A forgery adversary $\mathcal{A}$ against the MAC F has oracle access to $F_k(\cdot)$ (for which the key k is chosen uniformly at random), and thus can learn the tag values for some adaptively chosen messages of its choice $m_1, m_2, \ldots$. It then returns a *forgery* (m, τ) made of a message along with a tag. The forger wins if $F_k(m) = \tau$ *and* the tag for m was *not* queried before. We refer to an adversary of this kind as a (t, q, ε)-forger, where t and q

[2] We shall disregard the salt throughout this paper (as it has no affect on our results. We will also treat the counter as a block counter, rather than a bit counter. For the purposes of this paper, the definition we use is equivalent.

are upper bounds on its running time and on its number of MAC-queries, and ε is a lower bound on its success probability. In the same vein, a function family F is a (t, q, ε)-secure FIL-MAC family, if the success probability of any forger against F_k running in time t and using a number of at most q queries is at most ε for a randomly sampled key k. It is straightforward to see that a family of random functions with n-bit output is a $(t, q, 2^{-n})$-secure FIL-MAC for any value of q. We use the term "unforgeable" to denote a secure MAC.

Collision Resistance. It is well known that collision resistance cannot be defined for an individual hash functions (because of trivial adversaries already containing a collision and which is "produced" as output). A *keyed* hash function family H is (t, q, ε)-collision resistant if the advantage of any attacker running in time t and asking at most q queries to the hash function finds a collision on F_k with probability at most ε, when F_k is sampled at random from the family F.

Second Preimage Resistance. Amongst the numerous notions of second preimage resistance, we will mostly consider the one defined by the following game: a second preimage adversary $\mathcal{A}$ has oracle access to a compression function F. It receives a randomly generated challenge M of length ℓ, and succeeds if it outputs a second message M' such that $M \neq M'$ and $H^F(M) = H^F(M')$, where H is an iteration mode for F (such as Merkle-Damgård or HAIFA). Such an adversary $(t, q, \ell, \varepsilon)$-breaks H^F if its running time is at most t, and if after at most q queries to F its success probability is lower-bounded by ε. A hash function H^F is $(t, q, \ell, \varepsilon)$-second preimage resistant (SPR) if the advantage for messages of length ℓ of any attacker asking at most q queries and running in time at most t is upper-bounded by ε.

Preimage-Awareness. The Preimage-Awareness notion introduced in [16] is an adaptation of the notion of Plaintext-Awareness [4] to keyless constructions. Informally, it means that an adversary has to evaluate a cryptographic construction to obtain a later-useful output, *i.e.*, if one knows the output, then one is "aware" of a preimage. Both notions are defined in a model where one has oracle access to an idealized primitive. The Preimage-Awareness notion is therefore a property of *mode of iterations*, and not of monolithic primitives. We consider the setting where PA adversary are made of two parts $(\mathcal{A}_1, \mathcal{A}_2)$ allowed to have some common internal state and having oracle access to an ideal primitive $\mathcal{P}$ (say, a compression function). The game played by the adversary is defined as follows:

(1) $\mathcal{A}_1$ queries $\mathcal{P}$, and then output a hash value z. (2) A polynomial *extractor* $\mathbf{Ex}$ is then run on z and on the transcript of the interaction between $\mathcal{A}_1$ and $\mathcal{P}$. (3) The extractor outputs either $M_{\mathbf{Ex}} = \perp$ or a message $M_{\mathbf{Ex}}$ such that $H^{\mathcal{P}}(M_{\mathbf{Ex}}) = z$. (4) $\mathcal{A}_2$ is then run on the output $M_{\mathbf{Ex}}$ of the extractor, and outputs a message M'.

The adversary wins if $H^{\mathcal{P}}(M') = z$ and $M' \neq M_{\mathbf{Ex}}$. An adversary $(q, t, \varepsilon, \mathbf{Ex})$-break the PA of a construction H if q and t are upper-bounds on the number of queries to the ideal primitive and on the running time of $\mathcal{A}$, for a given extractor $\mathbf{Ex}$, while ε is a lower-bound on the success probability of $\mathcal{A}$. A construction H is PA if there exists an extractor for which no polynomial adversary achieves non-negligible success probability.

3 Second Preimage Resistance in the Random Oracle Model

Let us consider a *narrow-pipe* Merkle-Damgård or HAIFA iteration of a public random function. What can we say about the security of this construction? We know it offers collision-resistance up to the birthday bound, since the ideal compression function offers optimal collision-resistance, and any collision on the iteration induces a collision on the compression function. We also know that it offers optimal preimage-resistance, since a preimage on the iteration induces a preimage on the ideal compression function (on the last block). To the best of our knowledge, the second-preimage-resistance of this scheme is known to lie somewhere between $2^{n/2}$ and 2^{n-k} queries in the case of Merkle-Damgård for messages of length 2^k (resp. 2^n queries for Haifa).

We now come to the first point of this paper, namely that it is possible to achieve beyond-the-birthday-bound second preimage resistance when the compression function is considered to be ideal.

Theorem 1 (Second Preimage Resistance of Merkle-Damgård). *Let F be a public random function, let MD^F be the Merkle-Damgård iteration of F, and $\mathcal{A}$ a second preimage adversary against MD^F which (t,q,ℓ,ε)-break the SPR of MD^F. Then:*

$$\varepsilon \leq \frac{q \cdot \ell}{2^n}.$$

Before the proof, we note that throughout the paper $\left|M\right|$ denotes the length of M in blocks (and not in bits as usually done), i.e., $\left|M\right| = \ell$ if M has l blocks.

Actually, the main idea common to all the proofs presented in this paper is almost directly adapted from the existing generic second preimage attacks: we lower-bound the complexity of one particular step common to all these attacks, namely when some kind of a possible prefix has to be "connected" to the target message.

Proof (of theorem 1). Consider an adversary $\mathcal{A}$ that (t,q,ℓ,ε)-breaks the SPR of MD^F. We denote by h_i, for $1 \leq i \leq \ell$, the chaining values obtained while hashing M, according to the description in section 2.1. If $\mathcal{A}$ succeeds in finding a second preimage, then in particular $\mathcal{A}$ has found a collision. It is well known that in the presence of the Merkle-Damgård *strengthening*, this implies a collision on the compression function F [14, 24]. In our case, there exists an index i_0 such that one of the colliding chaining value is h_{i_0}. This collision on F is therefore actually a second preimage of h_{i_0} for F. Note that because F is a random function, all the h_i's are random values.[3]

We now give an upper bound on the probability that $\mathcal{A}$ finds a second preimage of one out of ℓ random chaining values. We simulate the execution of $\mathcal{A}$, and bookmark the queries sent to the oracle for F. Every time $\mathcal{A}$ submits a new query to the oracle, it receives a uniformly-distributed random value. The probability that $\mathcal{A}$ wins thanks to this particular query is upper-bounded by the probability that this random value is one of the h_i's. This probability is exactly $\ell \cdot 2^{-n}$. Since $\mathcal{A}$ sends at most q queries, $\mathcal{A}$ wins with probability at most $q \cdot \ell \cdot 2^{-n}$. □

It must be noted that this proof is fairly general, because it reduces the problem of finding a second preimage for MD^F to the problem of finding a second preimage of one out of many random chaining values for F. It actually covers nearly all the existing iterated hash functions; for example, it could be adapted to the EMD [2] mode of iteration, to Shoup's UOWHF [30], to Rivest's dithered hash [28], to Haifa [6], etc.

The inventors of Haifa claim that it has optimal resistance against generic second preimage attacks. The bound given by theorem 1 is however not strong enough to back up their claim. A slightly more involved proof technique is required to prove that Haifa achieves optimal second preimage resistance. The next theorem captures the intuitive idea that the attacks of [15, 21, 1] do not work against Haifa.

Theorem 2 (Second Preimage Resistance of Haifa). *Let F be a public random function and Haifa^F be the Haifa-iteration of F, and $\mathcal{A}$ a second preimage adversary that (t,q,ℓ,ε)-break the SPR of Haifa^F. Then:*

$$\varepsilon \leq \frac{q}{2^{n-1}}.$$

[3] We note that this claim is not necessarily true when the message is long and there are collision between the various chaining values. However, as this has a non-negligible probability only when $\ell \geq O(2^{n/2})$, we allow ourselves to disregard such very long messages.

The proof is deferred to annex A. It uses essentially the same techniques as the proof of theorem 1, but takes into consideration the counter used in each application of F.

Partial Conclusion: What do we Learn From These Proofs? These two results rely crucially on the fact that the compression function is modeled as a public random function. For this reason, it could be argued that since no actual compression function will ever satisfy this hypothesis, then our results are vacuous.

Considering the underlying primitive to be ideal is a natural idea when reasoning about *modes of iteration* of hash functions. The 64 constructions from PGV were proved secure in the Ideal Cipher model [7], *i.e.* assuming that the underlying primitive is ideal. At the very least, the security results obtained in our model imply security against generic attacks, so they say something meaningful about the security of the mode of iteration itself. For example, we know that the existing generic second preimage attacks [1, 20, 21] are almost optimal: in order to find a second preimage on Merkle-Damgård in less than 2^{n-k} operations, an attacker has to take a look at what is happening inside the compression function.

4 Hashing With an Unforgeable Compression Function

We were able to state and prove theorems 1 and 2 because we were reasoning in the random oracle model, which is arguably too strong and unrealistic (for symmetric cryptography). It is however in this model that the generic attacks which have inspired these proofs have been conceived. After noting that in both the indifferentiability framework and in the definition of the PA notion the underlying primitive is also assumed to be ideal, we now get to the second point of this paper. Assuming that the compression function is random does not tell anyone how to design better compression functions. We therefore try to identify more precisely what are the good properties of the compression function that would allow us to replay the same proofs, while being less far-fetched.

It seems that the crucial point in the proofs presented earlier is that the adversary has to evaluate the compression function to know its output, and that this output cannot be predicted or biased. This is reminiscent of some well-known notions such as pseudorandomness, unpredictability, or unforgeability. Additionally, it is folklore that a random function can be replaced by a pseudorandom function, as long as it is still accessed through a black-box interface. In this section, we show that assuming the *unforgeability* of the compression function is sufficient to prove many security properties on the iteration. It must be noted that this property is strictly weaker than the function being a PRF.

In the sequel, we consider a mode of iteration H which is either Merkle-Damgård or Haifa. We still suppose that the compression function F is accessed through a black-box interface, and that it takes as input the (chaining value, message block) pair (and also the counter when relevant for Haifa). We first prove that if F is an unforgeable function family, then H^{F_k} is both collision-resistant and preimage resistant if k is an (unknown) key sampled uniformly at random.

Theorem 3. *Let $\mathcal{A}$ be a collision-adversary that (t, q, ε)-breaks the collision resistance of H^{F_k}. Then we may construct a MAC-forger $\mathcal{B}$ that (t, q, μ)-breaks F, with:*

$$\mu = \frac{2 \cdot \varepsilon}{q \cdot (q+1)}$$

Proof. We simulate the execution of $\mathcal{A}$ and bookmark the queries sent to the F_k: let us denote by (h_i, x_i) the i-th query sent to F_k and by $y_i = F_k(h_i \,||\, x_i)$ the answer of F_k.

Let us observe what happens when $\mathcal{A}$ wins. Then $\mathcal{A}$ has necessarily sent two queries (h_i, x_i) and (h_j, x_j) (with $1 \leq i < j \leq q$) that collide, *i.e.*, yield the same output $y_i = y_j$. We call the j-th query the *magic query*. If we knew the indices i and j *in advance*, it would be easy to forge F_k: we could interrupt $\mathcal{A}$ just after it issued the magic j-th query, and not relay it to F_k. Instead, we could output the forgery: $(h_j \,||\, x_j, y_i)$.

To construct $\mathcal{B}$, we just guess uniformly at random the values of i and j (by fixing $1 \leq i < j \leq q$). We now have to lower-bound the probability of success of the forgery. There are $q \cdot (q+1)/2$ pairs (i,j) such that $1 \leq i < j \leq q$. Choosing i and j uniformly at random will yield the good guess with probability at least $2/(q \cdot (q+1))$. □

Theorem 4. *Let $\mathcal{A}$ be a preimage-adversary that (t,q,ε)-breaks the preimage resistance of H^{F_k}. Then we may construct a MAC-forger $\mathcal{B}$ that $(t,q,\varepsilon \cdot q^{-1})$-breaks F.*

Proof. We simulate the execution of $\mathcal{A}$, exactly as in the proof of theorem 3. Let us denote by h the challenge hash. If $\mathcal{A}$ wins, then one of its queries yields h as its answer. If we knew in advance the index i of this particular query, then we could play the same trick, interrupting $\mathcal{A}$, not relaying this query to F_k, and then outputting the forgery $(h_i \,||\, x_i, h)$.

Now, to construct $\mathcal{B}$, we guess the index of the "magic query" by taking i uniformly at random with $1 \leq i \leq q$. Since our guess will be right with probability at least $1/q$, this gives a lower-bound on the success probability of $\mathcal{B}$. □

The case of second preimage is a bit more involved, especially in the case of Haifa, as it is intermediate between the collision case and the preimage case.

Theorem 5. *i) Let $\mathcal{A}$ be a second preimage-adversary that (t,q,ℓ,ε)-break the SPR of MD^{F_k}. Then we may construct a MAC-forger $\mathcal{B}$ that (t,q,μ)-break F, with:*

$$\mu = \frac{\varepsilon}{q \cdot \ell}$$

ii) Let $\mathcal{A}$ be a second preimage-adversary that (t,q,ℓ,ε)-break the SPR of Haifa^{F_k}*. Then we may construct a MAC-forger $\mathcal{B}$ that (q,t,μ)-break F, with:*

$$\mu = \frac{\varepsilon}{q}$$

Proof. We simulate the execution of $\mathcal{A}$, exactly as in the proofs of theorems 3 and 4. Let us denote by $m_1, \ldots, m_\ell$ the chaining values produced by the hashing process of the challenge M, as they play a particular role, and let h be the hash value of the challenge M.

i) We observe when $\mathcal{A}$ wins, then $\mathcal{A}$ has necessarily sent a query (h_j, x_j) (with $1 \leq j \leq q$) that yielded one of the chaining values m_i obtained while hashing the challenge M. Again, if we knew the indices i and j *in advance*, it would be easy to forge F_k: we could just play the same trick, stop $\mathcal{A}$ before relaying its j-th query and outputting the forgery: $(h_j \,||\, x_j, m_i)$.
To construct $\mathcal{B}$, we again guess uniformly at random the values of i and j (by fixing $1 \leq j \leq q$ and $1 \leq i \leq \ell$). We now have to lower-bound the success probability of the forgery. There are $q \cdot \ell$ pairs (i,j) such that $1 \leq i \leq q$, $1 \leq i \leq \ell$. Choosing i and j uniformly at random will yield the good guess with probability $1/(q \cdot \ell)$.

ii) In the case of Haifa, we consider the same division into two cases as in the proof of theorem 2, as there are essentially two possible strategies for $\mathcal{A}$ to win. $\mathcal{A}$ can:
   (a) either find a second preimage M' with $|M'| = |M|$, and "connect" somewhere in the middle of M.
   (b) or find a second preimage M' with $|M'| \neq |M|$, which essentially amount to invert F_k on the last block.

Because we do not know in advance which strategy $\mathcal{A}$ will use, we flip a coin to "guess" what strategy will $\mathcal{A}$ make us of. Then we guess the index i of the "magic query" uniformly at random such that $1 \leq i \leq q$. We simulate $\mathcal{A}$ as before, interrupt it after it sent its i-th query, and do not relay the query to F_k. Then:

– If we guessed that $\mathcal{A}$ will "connect" to M, we look at the counter value c_i of the i-th query. We then output the forgery $(h_i \,||\, x_i \,||\, c_i, m_{c_i})$.

– Or if we "guessed" that $\mathcal{A}$ would invert the last block, we output the forgery $(h_i \,||\, x_i \,||\, c_i, h)$.

In both cases, provided that the "guess" was right and that $\mathcal{A}$ wins, the probability that the choice of i was right is at least $1/q$, which in turn gives a lower bound on the success probability of the forger.

□

The results can be summarized in the following synthetic way:

Corollary 1. *Assume that F_k is an optimally secure MAC, with the key k being sampled uniformly at random. Then*

i) H^{F_k} is collision resistant up to $2^{(n-1)/2}$ queries.
ii) MD^{F_k} is second-preimage resistant up to 2^{n-k} queries.
iii) Haifa^{F_k} *is second-preimage resistant up to 2^{n-1} queries.*
iv) H^{F^k} is preimage resistant up to 2^n queries.

5 Preimage-Awareness Using an Unforgeable Primitive

While we were trying to find a suitable assumption on the compression function, the notion of Preimage-Awareness was a natural candidate. It is weaker than being a full-blown RO, and at the same time it allows to prove interesting and useful security results for Merkle-Damgård. Unfortunately, the notion of PA could not help us to find a replacement for the random oracle, as it can only be achieved up to the birthday bound (not to mention that it cannot be a property of monolithic compression functions).

However, the PA notion was reminiscent of some kind of unforgeability property. Consequently, we have investigated the relation between PA and unforgeability. The next theorem demonstrates that PA is implied by the unforgeability of the underlying primitive.

When we say that a construction is PA, it is always with respect to a certain extractor. We therefore define our extractor $\mathbf{Ex}^U$, which we call the "universal extractor". We consider the set of queries issued by $\mathcal{A}_1$ as a graph, where each vertex is labeled with a chaining value, each edge is label-led with a message block, and $x \xrightarrow{m} y$ means that $F_k\,(x \,||\, m) = y$. In this graph, there are two special nodes, the IV node, and the z node, where z is the hash value $\mathcal{A}_1$ committed to. If there is a path in this graph connecting the IV node to the z node, then the sequence of blocks labeling the edges of this path forms a message that hashes to z. In that case, $\mathbf{Ex}^U$ replies it. If there is no such path, $\mathbf{Ex}^U$ replies $\perp$. It is straightforward to check that $\mathbf{Ex}^U$ runs in polynomial time in the size of the transcript.

We conjecture that this is the best possible extractor in the case of Merkle-Damgård. In any case, the next theorem show that if the compression function is unforgeable, then its Merkle-Damgård or Haifa iteration is PA with respect to this reasonably good extractor.

Theorem 6. *Let $\mathcal{A}$ be a PA-adversary that $(t, q, \varepsilon, \boldsymbol{Ex}^U)$-breaks the Preimage-Awareness of H^{F_k} for the universal extractor. Then we may construct a MAC-forger $\mathcal{B}$ that (t, q, μ)-breaks F, with:*

$$\mu = \frac{\varepsilon}{q \cdot (q+1)}$$

Proof. We consider a two-part PA adversary $(\mathcal{A}_1, \mathcal{A}_2)$ against the preimage-awareness of H^{F_k}. Assume that $\mathcal{A}_1$ committed to the hash value z, and that $\mathcal{A}$ wins. There are again two possible scenarios:

1. The extractor $\mathbf{Ex}^U$ replies $\perp$. In that case, $\mathcal{A}$ wins by outputting a message that is a preimage of h.
2. The extractor $\mathbf{Ex}^U$ outputs a message $M_{\mathbf{Ex}}$. In that case, $\mathcal{A}$ wins by outputting a message M that collides with $M_{\mathbf{Ex}}$. $\mathcal{A}$ essentially finds a collision in this case.

As we did in the proof of Theorem 5, we flip a coin to "guess" what the strategy of $\mathcal{A}$ will be. The second case is in fact covered by Theorem 3, and in that case we can just ignore the extractor and look for a collision (note that we can abort early if it outputs $\perp$, as this means that our guess was wrong).

Let us now assume that we are in the case where the extractor replied $\perp$. If we were to run the extractor after $\mathcal{A}_2$, and providing it with all the additional queries $\mathcal{A}_2$ sent to F_k, then the extractor would answer with a message (and not $\perp$), because $\mathcal{A}_2$ output a message that it necessarily hashed (so $\mathbf{Ex}^U$ would find the path created by this message in the graph).

This allows us to make use of a hybrid argument: there is a particular query of $\mathcal{A}_2$ that makes the extractor change its answer from $\perp$ to some message. This is the magic query. Note that we cannot expect this query to yield the answer z (suppose for example that z was obtained by hashing a fixed message from a random IV, and that the magic query connects to this IV). So what can we say about the answer of F_k to the magic query?

We can interpret the answer of the extractor in graph-theoretic terms: the fact that the extractor answers $\perp$ means that the two special nodes IV and z (where z is the output of A_1) are in two distinct connected components, and *a contrario* the fact that it answers a message means that there exist a path between them. The magic query therefore connects a node that is reachable from the IV node to a node from which z is reachable. Let us rephrase this idea: the magic query yield a chaining value which is part of another query previously sent to F_k.

To construct $\mathcal{B}$, we then proceed as follow:

– We guess the strategy of $\mathcal{A}$ uniformly at random.
– If we guessed that the extractor will reply a message, we ignore it and do as if $\mathcal{A}$ were a collision adversary.
– If we guessed that the extractor will reply $\perp$, we pick uniformly at random two indices i and j such that $1 \leq i < j \leq q$. We then run $\mathcal{A}$, interrupt it when it sends the j-th query, and output the forgery: $(h_j \,||\, x_j, h_i)$.

What remains is to lower-bound the success probability of the forgery. In both case, if we guessed the strategy of $\mathcal{A}$ correctly, we choose a pair of indices amongst $q\cdot(q+1)/2$. Therefore, in each separate case the success probability is $\frac{2\varepsilon}{q\cdot(q+1)}$. Since the guess is only right with probability $1/2$, this gives the announced result. $\square$

Corollary 2. *Assume that F is an optimally unforgeable function family Then H^{F_k} is Preimage-Aware up to the birthday bound if the (unknown) key k is sampled uniformly at random.*

6 Concluding Thoughts

We first demonstrated that there are proofs of second preimage resistance beyond the birthday bound for both Merkle-Damgård and HAIFA when the compression function is modeled as a random oracle. We then demonstrated that the same results can be achieved when the compression function is a secure MAC to which one only has *oracle access* (in particular, the key is unknown). We proceeded to show that Preimage Awareness is implied by the unforgeability assumption, hence, suggesting a more viable design target for implementors.

In this somewhat non-technical concluding section, we discuss the possibility of a *reduction* of the second preimage resistance of H^F to some particular properties of F. Such a reduction would have the advantage of removing the need for F to be accessed through an oracle interface, and provide a proof that covers practical situations. So far, such a reduction has not been found for the ubiquitous Merkle-Damgård mode.

Impossibility of a Reduction to Randomness Assumptions. The randomness of F, or more precisely the fact that $\mathcal{A}$ *has* to evaluate it and cannot *predict* or *bias* its output, plays an essential role in all the proofs presented in this paper. In particular, the randomness of the output can be either achieved when the used function contains sufficient randomness (e.g., a random oracle), or that there is a hidden key which "masks" nonrandom behavior.

In the first case, it is clear that if an adversary succeeds in breaking the SPR of H^F with an advantage greater that what the proofs guarantee, then clearly F is not random. This reduces the second preimage resistance of H^F to some kind of "randomness" property of F, such as unpredictability [26] or into random oracle assumption [3].

Unfortunately, the second class of randomness notions (such as PRF-ness [26]) are usually defined though a distinguishing game which again supposes the black-box access to the primitive we are precisely trying to avoid. The main obstacle is that the description of H^F is public; the distinguishing or forgery games are meaningless when the description of H^F is public. The use of keys cannot circumvent this problem, as they are made public.

Existing Constructions and the Specific Problem of Merkle-Damgård and HAIFA. As mentioned in the introduction, Shoup's UOWHF [30] and the Randomized Hashing [17] achieve beyond the birthday bound security in the standard model. Both constructions however lose a factor ℓ in the reduction. The later reduces the SPR of the iteration the e-SPR notion on the compression function. The e-SPR notion is not very natural, as it expresses the fact that the second preimage resistance of H^F actually depends on properties of the *iteration* of F. In the case of Merkle-Damgård, it would also be possible to define a similar *ad hoc* security notion, with the same level of guarantee. However, the common inconvenience of these *ad hoc* notion is that they do not help at all compression function designers in producing compression functions that will provide provable second preimage resistance for Merkle-Damgård or HAIFA.

Suppose that we are given a second preimage adversary $\mathcal{A}$ against an iterated mode of iteration. We wish to use $\mathcal{A}$ to attack some property against a single iteration of the compression function. Intuitively, the interesting thing that $\mathcal{A}$ does is "connecting" to M. However, we have no control over *where* in M the connection happens. Shoup's UOWHF and the randomized hashing solve this problem by *manipulating the input* of the compression function: in Shoup's UOWHF, masks that are part of the key are XORed to the chaining value, while in the randomized hashing, the key is a mask that is XORed to all the message blocks. This enables them to place a single-block challenge *at a random position* in a bigger message M by choosing the key carefully. Intuitively, $\mathcal{A}$ cannot tell in which place in M is the actual challenge we are interested in. This means that whatever $\mathcal{A}$'s strategy be, the "connection" will hit the actual challenge with probability $1/\ell$. This explains, by the way, the loss of a factor ℓ in the security proofs of these two constructions. Such a loss appears to be unavoidable, especially as there are attacks which offer this time complexity (providing an upper bound for the security).

In the case of Merkle-Damgård and HAIFA we cannot easily manipulate the input of the compression function. This makes it very difficult to randomly embed a single block challenge into a bigger message. It therefore seems unlikely to the authors that the second preimage resistance of these schemes could be established thanks to a reduction to a similar property of the compression function. Additionally, the existence of an unavoidable loss of a factor ℓ in the existing security proofs for Merkle-Damgård opens a new gap between second preimage resistance in the standard and random oracle models. Closing this gap, by either exhibiting a narrow-pipe mode of iteration which enjoys full SPR in the standard model, or proving some kind of separation result, is an exciting subject of future work.

References

1. Andreeva, E., Bouillaguet, C., Fouque, P.A., Hoch, J.J., Kelsey, J., Shamir, A., Zimmer, S.: Second Preimage Attacks on Dithered Hash Functions. In Smart, N.P., ed.: EUROCRYPT. Volume 4965 of Lecture Notes in Computer Science, Springer (2008) 270–288
2. Bellare, M., Ristenpart, T.: Multi-Property-Preserving Hash Domain Extension and the EMD Transform. In Lai, X., Chen, K., eds.: ASIACRYPT. Volume 4284 of Lecture Notes in Computer Science, Springer (2006) 299–314
3. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: ACM Conference on Computer and Communications Security. (1993) 62–73
4. Bellare, M., Rogaway, P.: Optimal Asymmetric Encryption. In Santis, A.D., ed.: EUROCRYPT. Volume 950 of Lecture Notes in Computer Science, Springer (1994) 92–111
5. Bellare, M., Rogaway, P.: Collision-Resistant Hashing: Towards Making UOWHFs Practical. [19] 470–484
6. Biham, E., Dunkelman, O.: A Framework for Iterative Hash Functions — HAIFA. Cryptology ePrint Archive, Report 2007/278 (August 24–25 2006) `http://eprint.iacr.org/2007/278`.
7. Black, J., Rogaway, P., Shrimpton, T.: Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. In Yung, M., ed.: CRYPTO. Volume 2442 of Lecture Notes in Computer Science, Springer (2002) 320–335
8. Boldyreva, A., Fischlin, M.: Analysis of Random Oracle Instantiation Scenarios for OAEP and Other Practical Schemes. [31] 412–429
9. Brassard, G., ed.: CRYPTO '89, Santa Barbara, California, USA, August0-24, 1989, Proceedings. In Brassard, G., ed.: CRYPTO. Volume 435 of Lecture Notes in Computer Science, Springer (1990)
10. Canetti, R.: Towards Realizing Random Oracles: Hash Functions That Hide All Partial Information. [19] 455–469
11. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. ACM **51**(4) (2004) 557–594
12. Chang, D., Nandi, M.: Improved Indifferentiability Security Analysis of chopMD Hash Function. In Nyberg, K., ed.: FSE. Volume 5086 of Lecture Notes in Computer Science, Springer (2008) 429–443
13. Coron, J.S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård Revisited: How to Construct a Hash Function. In: CRYPTO'05. (2005) 430–448
14. Damgård, I.: A Design Principle for Hash Functions. [9] 416–427
15. Dean, R.D.: Formal Aspects of Mobile Code Security. PhD thesis, Princeton University (January 1999)
16. Dodis, Y., Ristenpart, T., Shrimpton, T.: Salvaging Merkle-Damgard for Practical Applications. In: EUROCRYPT '09, Springer-Verlag (2009) `http://people.csail.mit.edu/dodis/ps/md-good.ps`.
17. Halevi, S., Krawczyk, H.: Strengthening Digital Signatures Via Randomized Hashing. In Dwork, C., ed.: CRYPTO. Volume 4117 of Lecture Notes in Computer Science, Springer (2006) 41–59
18. Impagliazzo, R., Naor, M.: Efficient Cryptographic Schemes Provably as Secure as Subset Sum. J. Cryptology **9**(4) (1996) 199–216
19. Kaliski, B.S.J., ed.: Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings. In Kaliski, B.S.J., ed.: CRYPTO. Volume 1294 of Lecture Notes in Computer Science, Springer (1997)
20. Kelsey, J., Kohno, T.: Herding Hash Functions and the Nostradamus Attack. In Vaudenay, S., ed.: EUROCRYPT'06. Volume 4004 of Lecture Notes in Computer Science, Springer (2006) 183–200
21. Kelsey, J., Schneier, B.: Second Preimages on n-Bit Hash Functions for Much Less than 2^n Work. In Cramer, R., ed.: EUROCRYPT'05. Volume 3494 of Lecture Notes in Computer Science, Springer (2005) 474–490
22. Lucks, S.: A Failure-Friendly Design Principle for Hash Functions. In Roy, B.K., ed.: ASIACRYPT'05. Volume 3788 of Lecture Notes in Computer Science, Springer (2005) 474–494
23. Maurer, U.M., Renner, R., Holenstein, C.: Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In Naor, M., ed.: TCC. Volume 2951 of Lecture Notes in Computer Science, Springer (2004) 21–39

24. Merkle, R.C.: One Way Hash Functions and DES. [9] 428–446
25. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable Random Functions. In: FOCS. (1999) 120–130
26. Naor, M., Reingold, O.: From Unpredictability to Indistinguishability: A Simple Construction of Pseudo-Random Functions from MACs (Extended Abstract). In Krawczyk, H., ed.: CRYPTO. Volume 1462 of Lecture Notes in Computer Science, Springer (1998) 267–282
27. Naor, M., Yung, M.: Universal One-Way Hash Functions and their Cryptographic Applications. In: STOC, ACM (1989) 33–43
28. Rivest, R.L.: Abelian Square-Free Dithering for Iterated Hash Functions. Presented at ECrypt Hash Function Workshop, June 21, 2005, Cracow, and at the Cryptographic Hash workshop, November 1, 2005, Gaithersburg, Maryland (August 2005)
29. Rogaway, P., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In Roy, B.K., Meier, W., eds.: FSE. Volume 3017 of Lecture Notes in Computer Science, Springer (2004) 371–388
30. Shoup, V.: A Composition Theorem for Universal One-Way Hash Functions. In: EUROCRYPT'00. (2000) 445–452
31. Shoup, V., ed.: Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings. In Shoup, V., ed.: CRYPTO. Volume 3621 of Lecture Notes in Computer Science, Springer (2005)

A Proof of Theorem 2

We simulate the execution of the adversary $\mathcal{A}$, and bookmark the queries sent by A to F: it is a set S of tuples (x, m, c, y), with $y = F(x, m, c)$. We suppose that $\mathcal{A}$ evaluates $H^F(M)$, so $\mathcal{A}$ sends the corresponding queries to the oracles at some point. Let us denote these particular queries $(h_i, m_i, c_i, h_{i+1})_{1 \leq i \leq \ell}$. In particular, $H^F(M) = h_{\ell+1}$.

Suppose now that $\mathcal{A}$ wins. We first eliminate the special case when $\mathcal{A}$ finds a preimage of h_r for F (this essentially means that $\mathcal{A}$ has found a preimage without using the fact that M is known).

1. If $|M| \neq |M'|$, then the values of the counter entering the compression function in its last invocation are different. Therefore, $\mathcal{A}$ has found a second preimage on F. Each query has a probability 2^{-n} to give this preimage, because of the randomness of F.
2. Otherwise, $|M| = |M'|$. This means that $\mathcal{A}$ has found collision with M, similarly to what happens in the proof of theorem 1. We model this situation with the following event, that we call **E**. Intuitively, **E** is realized as soon as $\mathcal{A}$ submits a query to F the answer of which gives a second preimage of one of the h_i. Formally, **E** is realized if and only if there is in S a query (x, m, i_0, h_{i_0+1}) for a given value of i_0 (recall that h_{i_0} is the i_0-th chaining value obtained in the process of hashing M), and such that $(x, m) \neq (h_{i_0}, m_{i_0})$.

Claim. If $\mathcal{A}$ wins and $|M| = |M'|$, then **E** is realized.

Justification. Thanks to the result of Merkle-Damgård, we know that there is a collision on the compression function where one of the colliding hash value is one of the h_i. However, this is not sufficient to say that **E** is realized, because we would need to know that the values of the counter are actually the same. We now prove that it is indeed the case.

Lemma 1 (Collision-Resistance Preservation on HAIFA). *Let H^F be the* HAIFA *iteration of an arbitrary compression function F.*

If $H^F(M) = H^F(M')$ with $M \neq M'$ and $|M| = |M'|$, then there is a collision on F, with the same value of the counter (this means that **E** *is realized).*

Proof. let us note $M = x_1, \ldots, x_r$, $M' = x'_1, \ldots, x'_r$, $h_0 = h'_0 = IV$, $h_i = F(h_{i-1}, x_i, i)$ and $h'_i = F(h'_{i-1}, x'_i, i)$.

Since $h_r = h'_r$, either there is a collision on F (with counter value r), or $(x_r, h_{r-1}) = (x'_r, h'_{r-1})$. In the latter case, either there is a collision for F (with counter value $r - 1$) or $(x_{r-1}, h_{r-2}) =$

(x'_{r-1}, h'_{r-2}). This argument repeats. Since $|M| = |M'|$, then either there is a collision for F at some point (with the same counter value), or $x_i = x'_i$, for all i, $1 \leq i \leq r$. In the latter case, $M = M'$, which is impossible. This completes the proof of the lemma. □

To complete the proof of theorem 2, we now show an upper-bound on the probability that **E** is realized. When $\mathcal{A}$ submits its i-th query to the simulator (and note that the number i is part of the query), a random value is chosen by the simulator and returned to $\mathcal{A}$. The event **E** is realized if and only if this value is h_{i+1}, and this happens with probability 2^{-n}. This query may also allow $\mathcal{A}$ to invert h_ℓ with probability 2^{-n}. Each query allows $\mathcal{A}$ to win with probability $2^{-(n-1)}$, and there are q queries, which completes the proof. □