Internal Distinguishers in Indifferentiable Hashing: The Shabal Case*

Emmanuel Bresson[1], Anne Canteaut[2], Thomas Fuhr[1], Thomas Icart[3], María Naya-Plasencia[4], Pascal Paillier[5], Jean-René Reinhard[1], Marion Videau[1]

[(1)]ANSSI (France), [(2)]INRIA Paris-Rocquencourt (France), [(3)]Sagem Securité (France) and University of Luxembourg (Luxembourg), [(4)]FHNW Windisch (Switzerland), [(5)]CryptoExperts and Gemalto (France).

Abstract

We show the first indifferentiability proof of a hash construction $\mathcal{C}^{\mathcal{F}}$ which does not make the assumption that the inner primitive $\mathcal{F}$ is ideal, but allows the existence (up to certain bounds that we explicit) of statistical distinguishers on $\mathcal{F}$. Our hash construction is a general domain extender that generalizes both Chop-MD and Shabal and we prove that this general mode of operation is indifferentiable from a random oracle by providing tight security bounds when the inner primitive $\mathcal{F}$ is either an ideal compression function or a keyed permutation. Our proof provides the tightest possible security bounds on Chop-MD and even improves the original indifferentiability proof of Shabal. We then extend our results to the case where $\mathcal{F}$ is not assumed ideal anymore, but presents some (possibly strong) form of statistical bias in its input-output behavior. Our results allow us to derive new indifferentiability bounds for Shabal and show that the series of recently found (order-1, differential or rotational) distinguishers on its internal keyed permutation leave fully intact its indifferentiability properties.

1 Introduction

Shabal is one of the fastest unbroken candidate to the NIST hash competition. It is based on a new domain extender, which is in some sense intermediate between the classical Merkle-Damgård construction and the sponge construction, and which is provably secure. The underlying design idea was to adapt the provably secure mode of operation of the sponge construction in order to use a permutation over a smaller set, which can be faster. Shabal's mode of operation is proven to be secure in the ideal cipher model in [7, Chapter 5] in the following sense: it is indifferentiable from a random oracle up to a number of queries higher than the birthday bound. Moreover, similar results on the provable (second) preimage-resistance are given in [7].

Recently, some properties of the keyed permutation $\mathcal{P}$ used in Shabal have been observed by Aumasson *et al.*, by Knudsen *et al.* and by Van Assche [2, 4, 11, 1]. These works point out the existence of *related-key distinguishers* for $\mathcal{P}$, but all of them conclude that the given observations do not seem extensible to the full hash function, and have therefore no visible impact on the security of Shabal.

This paper provides evidence that Shabal fully remains indifferentiable even though $\mathcal{P}$ is not ideal. To do so, we actually address the following more general question. Assume that we are given a hash function $\mathcal{C}^{\mathcal{F}}$ which is made of a mode of operation $\mathcal{C}$ making calls to an internal primitive $\mathcal{F}$. Assume further that $\mathcal{C}$ is proved to be indifferentiable from a random oracle, which in turn means that $\mathcal{C}^{\mathcal{F}}$ behaves ideally assuming that $\mathcal{F}$ behaves ideally. When specifying an instantiation of the hash construction, one has to select a functional embodiment for $\mathcal{F}$ and provide a full-fledged description of the primitive $\mathcal{F}$. Assume now that the specified primitive does not behave ideally (or at least not in the sense required by the indifferentiability proof for $\mathcal{C}$). It seems at first sight that the indifferentiability result on $\mathcal{C}$ does not constitute a security

*This work was partially supported by the French Agence Nationale de la Recherche through the SAPHIR2 project under Contract ANR-08-VERS-014.

argument anymore since the basic requirement of the proof ($\mathcal{F}$ behaves ideally) is obviously not obeyed. The question we ask is whether $\mathcal{C}^{\mathcal{F}}$ can still be considered as a good hash function.

From a higher perspective, the question relates to a more general paradigm: can we prove that a construction $\mathcal{C}^{\mathcal{F}}$ behaves ideally even though $\mathcal{F}$ does not? We may ask ourselves to which extent $\mathcal{C}^{\mathcal{F}}$ differs from an ideal hash function when $\mathcal{F}$ differs from an ideal primitive. Obviously, it is desirable that a construction $\mathcal{C}$ remains close to ideal even when $\mathcal{F}$ is far from ideal. One may think of this notion as a form of robustness: even if a weakness is discovered on the full-fledge primitive $\mathcal{F}$ some time in the future, the hash function $\mathcal{C}^{\mathcal{F}}$ would remain almost equally ideal. Thus our motivation is driven by practical considerations; constructions that are robust in this sense answer the quest for more reliable hash proposals.

In this paper, we clarify the impact of such distinguishers on the security of Shabal: a new security proof for Shabal's mode of operation is provided where the keyed permutation is not assumed to be an ideal cipher anymore, but complies with some (standard model) distinguishing property. This new result underlines that the round keyed permutation of Shabal does not *need* to be ideal to achieve the SHA-3 security requirements. Most interestingly, the distinguishers for $\mathcal{P}$ put forward in [2, 4, 11, 1] are proven not to weaken the security of Shabal.

2 A general domain extender

In this paper, we focus on the general domain extender depicted on Figure 1. Our construction hashes arbitrary input messages $\mathcal{M} \in \{0,1\}^*$. At each round, an ℓ_m-bit message block is processed and the n-bit chaining value (also called internal state) is updated. Once the entire padded message $\mathsf{pad}(\mathcal{M})$ has been processed, the ℓ_h-bit hash value corresponds to a part of the final chaining value. We also consider a variant of this mode where n_f additional blank rounds are performed, once the whole padded message has been processed. Blank rounds just repeat n_f times the last message insertion round using the last message block.

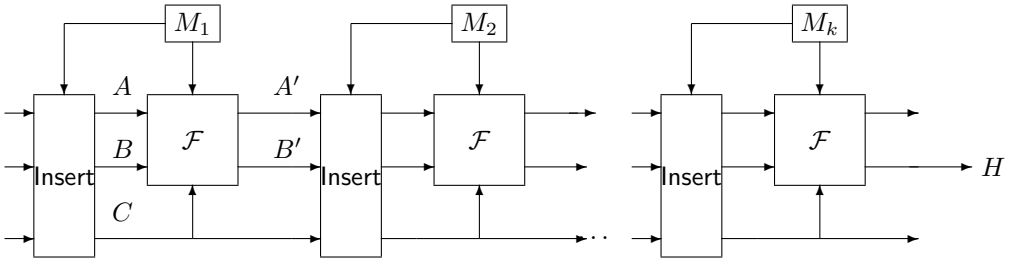


Figure 1: A general mode of operation.

The function used to update the chaining value at each round is composed of two operations. First, the message-block insertion $\mathsf{Insert}[M]$ performs an invertible transformation on the internal state, depending on the current message block $M \in \{0,1\}^{\ell_m}$. Second, the compression function or keyed permutation $\mathcal{F}$ takes as input the chaining value and the message block and outputs a new chaining value. However, applying $\mathcal{F}$ may not modify the entire internal state but only a fraction of it. In this case, the part of the internal state that remains unchanged can be seen as an extra parameter for $\mathcal{F}$. We split the n-bit internal state into three parts referred to as $A \in \{0,1\}^{\ell_a}$, $B \in \{0,1\}^{\ell_h}$ and $C \in \{0,1\}^{\ell_c}$ where $\ell_a + \ell_h + \ell_c = n$, and define the round function as

$$\begin{array}{rcl} \mathcal{F} : \{0,1\}^{\ell_m} \times \{0,1\}^n & \rightarrow & \{0,1\}^{\ell_a} \times \{0,1\}^{\ell_h} \\ (M, (A, B, C)) & \mapsto & (A', B'). \end{array}$$

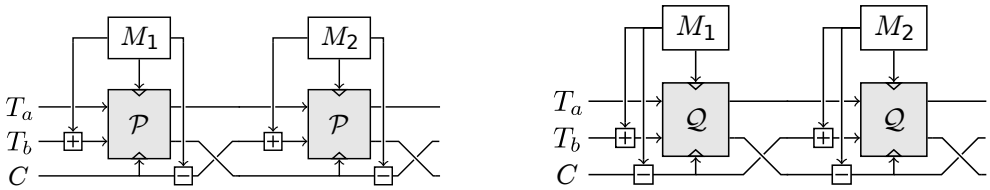
The i-th message round performs the transformation

$$\mathrm{Round}[M_i](x) = \mathcal{F}(M_i, \mathsf{Insert}[M_i](x))$$

where $x = (A, B, C)$ is the internal state at the beginning of the i-th round. The initial state is set to some initialization value $x_0 = (A_0, B_0, C_0)$. Once the entire padded message has been processed, and when there are no blank rounds ($n_f = 0$), the ℓ_h-bit hash value corresponds to the B-part of the final internal state. Otherwise n_f additional blank rounds are performed before extracting and returning the B-part of the internal state.

Our general domain extender includes a variant of the mode of operation of Shabal depicted in the original submission [7] and illustrated on Figure 2(a). In the original description of Shabal, the message block is subtracted from C after the call to the keyed permutation $\mathcal{P}$. We replace this with an equivalent mode where the subtraction is relocated before the call to $\mathcal{P}$. Simultaneously, we replace $\mathcal{P}$ with the new permutation $\mathcal{Q} : (M, T_a, T_b, C) \to \mathcal{P}(M, T_a, T_b, C \boxplus M)$. The new mode is shown on Figure 2(b). The parts of the chaining value referred to as T_a and T_b were denoted A and B in the submission document. This paper rather refers to B as the part which defines the output of the hash function, and to A as the truncated part. Obviously, T_b contains B and a fraction of A, and T_a contains the remainder of A. We then get an embodiment of our general mode of operation by taking $\mathcal{F} = \mathcal{Q}$ and $\mathsf{Insert}[M](A, B, C) = (A, C \boxplus M, B \boxminus M)$. Note that the original message counter of Shabal is ignored in our description.

Interestingly, our general domain extender also captures Chop-MD [9], which is obtained by taking a compression function for $\mathcal{F}$, by letting message insertion be the identity function $\mathsf{Insert}[M](A, B, C) = (A, B, C)$ and by setting $\ell_c = 0$ (*i.e.*, the compression function modifies the whole chaining value).



(a) Original representation.

(b) Equivalent representation.

Figure 2: Shabal's domain extender (message rounds).

3 Indifferentiability proof in the ideal cipher model

This section shows that our general mode of operation is indifferentiable from a random oracle. We provide the tightest possible security bounds, thus improving over prior results on Shabal and Chop-MD.

3.1 The indifferentiability framework

The concept of indifferentiability [12] specifies a security game played between an oracle system S and a distinguisher $\mathcal{D}$. It was used in [10] to evaluate the security of several variants of the Merkle-Damgård construction, including chop-MD. S may contain several components, typically a hash construction $\mathcal{C}^{\mathcal{F}}$ which makes calls to an inner primitive $\mathcal{F}$. The construction $\mathcal{C}$ is said to be indifferentiable (up to some security bound) if the system $S = (\mathcal{C}^{\mathcal{F}}, \mathcal{F})$ can be replaced by a second oracle system $S' = (\mathcal{H}, \mathcal{S}^{\mathcal{H}})$ with identical interface in such a way that $\mathcal{D}$ cannot tell the difference (see Figure 3). Here $\mathcal{H}$ is a random oracle and $\mathcal{S}$ is a simulator which must behave like $\mathcal{F}$. When the primitive $\mathcal{F}$ is a keyed permutation like in Shabal, $\mathcal{S}$ has to simulate both $\mathcal{F}$ and $\mathcal{F}^{-1}$.

In its interaction with the system S, the distinguisher makes calls to either $\mathcal{C}^{\mathcal{F}}$ or $\mathcal{F}$. Throughout the paper, N will denote the *total* number of calls received by $\mathcal{F}$ when $\mathcal{D}$ interacts with S – regardless of their origin which may be either $\mathcal{C}^{\mathcal{F}}$ or $\mathcal{D}$. We define the advantage of $\mathcal{D}$ as

$$\mathsf{Adv}(\mathcal{D}) = \left|\Pr\left[\mathcal{D}^S = 1 \mid S = (\mathcal{C}^{\mathcal{F}}, \mathcal{F})\right] - \Pr\left[\mathcal{D}^S = 1 \mid S = (\mathcal{H}, \mathcal{S}^{\mathcal{H}})\right]\right| ,$$

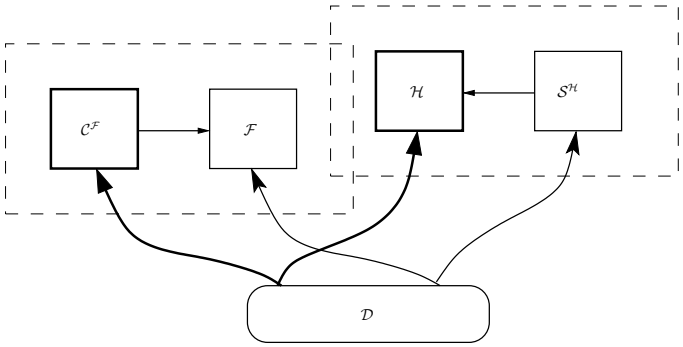


Figure 3: The hash construction $\mathcal{C}^{\mathcal{F}}$ has oracle access to $\mathcal{F}$. The simulator $\mathcal{S}^{\mathcal{H}}$ has oracle access to the random oracle $\mathcal{H}$. The distinguisher interacts either with $(\mathcal{C}^{\mathcal{F}}, \mathcal{F})$ or with $(\mathcal{H}, \mathcal{S}^{\mathcal{H}})$ and has to tell them apart.

where probabilities are taken over the random coins of all parties. Obviously $\mathsf{Adv}(\mathcal{D})$ is a function of N. The indifferentiability proof therefore consists in constructing an appropriate simulator for $\mathcal{F}$ (or for $(\mathcal{F}, \mathcal{F}^{-1})$ in the case of a keyed permutation) and in upper-bounding the advantage of any distinguisher interacting with it.

3.2 Basic design principles for the simulator

Let $\mathcal{X} = \{0,1\}^n$ denote the set of all possible internal states. Recall that the i-th message round consists in executing $\text{Round}[M_i](x) = \mathcal{F}(M_i, \mathsf{Insert}[M_i](x))$ on the current state $x \in \mathcal{X}$. The simulator of $\mathcal{F}$ (resp. of $(\mathcal{F}, \mathcal{F}^{-1})$) is obtained by dynamically constructing a graph $\mathcal{G} = (X, E) \subseteq \mathcal{X} \times \mathcal{X}^2$ where X is the set of nodes and E the set of edges. X is the set formed by all the states $\mathsf{Insert}^{-1}[M](A, B, C)$ where (M, A, B, C) is a query to $\mathcal{F}$ received by the simulator and by all associated responses completed with the corresponding C-part to yield a proper internal state $\in \mathcal{X}$. When $\mathcal{F}$ is a keyed permutation, X also contains the input queries (M, A', B', C) to the inverse function in which case $\mathsf{Insert}^{-1}[M](A, B, C)$ is also appended to X where (A, B) is the corresponding response. An edge between x and y in the graph is labelled by an ℓ_m-bit string M and it is denoted by $x \xrightarrow{M} y$. It expresses that x and y are associated by a definition of $\text{Round}[M]$ during the game. A *path* from the initial state x_0 to a node $x \in X$ in the graph is a non-empty list of ℓ_m-bit message blocks $\mu = \langle M_1, \ldots, M_k \rangle$ satisfying $M_{k-r} = M_k$ for all $0 \leq r \leq n_f$ and $\langle M_1, \ldots, M_{k-n_f} \rangle = \mathsf{pad}(\mathcal{M})$ for some $\mathcal{M} \in \{0,1\}^*$, such that there exist k edges in the graph of the form $x_{i-1} \xrightarrow{M_i} x_i$, $1 \leq i \leq k$, and $x_k = x$.

3.3 A tight indifferentiability proof when $\mathcal{F}$ is ideal

We consider the general mode of operation of Figure 1 and put forward the simulator $\mathcal{S}$ depicted on Figure 4. We describe $\mathcal{S}$ for a keyed permutation $(\mathcal{F}, \mathcal{F}^{-1})$. The simulation of $\mathcal{F}^{-1}$ can simply be ignored if $\mathcal{F}$ is a compression function.

Using our simulator, we obtain tight upper bounds on the distinguisher's advantage in the indifferentiability game. Our proof makes use of game-hopping to progressively construct the simulator $\mathcal{S}$. It can be shown that the probability gap between the original game where the adversary interacts with $S = (\mathcal{C}^{\mathcal{F}}, \mathcal{F})$ and the final game where she interacts with $S' = (\mathcal{H}, \mathcal{S}^{\mathcal{H}})$ equals the probability that the simulator aborts during the game, plus an additional term $p_3(N)$ that captures the success probability of length-extension attacks:

Theorem 1. *Consider the general mode of operation with $n_f \geq 0$ blank rounds and the simulator $\mathcal{S}$ defined*

Initialization of $\mathcal{S}$

No input, no output

1. set $X = \{x_0\}$ and $E = \emptyset$

Simulation of $\mathcal{F}$

Input: (M, A, B, C)

Output: (A', B')

1. set $x = \mathsf{Insert}^{-1}[M](A, B, C)$
2. if there exists an edge $x \xrightarrow{M} y \in E$
   (a) return (A', B') where $y = (A', B', C)$
3. if x has a path μ in graph $\mathcal{G}$
   (a) compute $\mathcal{M} = \mathsf{unpad}(\mu \parallel M)$
   (b) call $\mathcal{H}$ to get $h = \mathcal{H}(\mathcal{M})$
   (c) set $B' = h$
   (d) randomly select $A' \leftarrow \{0,1\}^{\ell_a}$ and set $y = (A', B', C)$.
4. else
   (a) randomly select $B' \leftarrow \{0,1\}^{\ell_h}$
   (b) randomly select $A' \leftarrow \{0,1\}^{\ell_a}$ and set $y = (A', B', C)$.
5. if $y \in X$ (event Abort_1), then abort
6. add nodes x and y to X and edge $x \xrightarrow{M} y$ to E
7. return (A', B')

Simulation of $\mathcal{F}^{-1}$

Input: (M, A', B', C)

Output: (A, B)

1. set $y = (A', B', C)$
2. if there exists an edge $x \xrightarrow{M} y \in E$
   (a) parse $\mathsf{Insert}[M](x)$ as $\mathsf{Insert}[M](x) = (A, B, C)$ for some A, B
   (b) return (A, B)
3. randomly select $A \leftarrow \{0,1\}^{\ell_a}, B \leftarrow \{0,1\}^{\ell_h}$ and set $x = \mathsf{Insert}^{-1}[M](A, B, C)$
4. if $x \in X$ (event Abort_2), then abort
5. add nodes x and y to X and edge $x \xrightarrow{M} y$ to E
6. return (A, B)

Figure 4: Simulator $\mathcal{S}$ for the general mode of operation in the ideal case. $\mathcal{S}$ simulates $(\mathcal{F}, \mathcal{F}^{-1})$ when $\mathcal{F}$ is a random keyed permutation, or just $\mathcal{F}$ when $\mathcal{F}$ is a random function.

as per Figure 4. Let

$$p_C = \begin{cases} \max_{m,c,c'} \Pr_{(a,b) \leftarrow \{0,1\}^{\ell_a + \ell_h}} \left[\mathsf{Insert}_C[m](a,b,c) = c'\right] & \text{if } n_f \geq 1\,, \\ 1 & \text{if } n_f = 0\,, \end{cases}$$

where the maximum is taken over $m \in \{0,1\}^{\ell_m}$, $c, c' \in \{0,1\}^{\ell_c}$ *and* $\mathsf{Insert}_C[m](x)$ *is the* C*-part of* $\mathsf{Insert}[m](x)$. *If* $\mathcal{F}$ *is a function, for any distinguisher* $\mathcal{D}$ *interacting with* $\mathcal{S}$ *and making at most* N *calls to* $\mathcal{F}$, *it holds that*

$$\mathsf{Adv}(\mathcal{D}) \leq \Pr\left[\mathsf{Abort}_1\right] + p_3(N)$$

where

$$p_3(N) = [1 + (1+e)p_C]\, 2^{-(\ell_a+\ell_h)}N^2 + \ell_h 2^{-(\ell_a-1)}p_C N$$

and $e = \exp(1) \approx 2.71828$. *Moreover when* $n_f = 0$ *and* $\ell_h > \ell_a$, *one has the tighter bound*

$$p_3(N) = \min\left(\ell_h 2^{-(\ell_a-1)}N + (1+e)2^{-(\ell_a+\ell_h)}N^2, 2^{-\ell_a}N\left\lceil\frac{\ell_h+\ell_a-\log_2(\sqrt{2\pi})}{\ell_h-\ell_a-\log_2 e}\right\rceil\right) .$$

If $\mathcal{F}$ *is a keyed permutation, for any distinguisher* $\mathcal{D}$ *totalling at most* N *calls to* $\mathcal{F}$ *and* $\mathcal{F}^{-1}$,

$$\mathsf{Adv}(\mathcal{D}) \leq \Pr\left[\mathsf{Abort}_1 \vee \mathsf{Abort}_2\right] + p_3(N) .$$

We comment that bounding $p_3(N)$ amounts to estimate the maximal success probability of length-extension attacks against the hash construction. More precisely, p_3 measures the probability that the adversary correctly guesses an internal state x reached when computing a hash value without making the corresponding calls to $\mathcal{F}$. When no blank rounds are performed, both the B- and C-parts of the final internal state in a hash computation may be known to the adversary: the B-part is known since it corresponds to the output hash value, and the C-part can be freely set to a prescribed value by choosing an appropriate last message block. Full control over the C-part may be avoided by adding at least one blank round parametrized by the last message block: it can be shown that if the insertion function is well-chosen, then the C-part of the final internal state in a hash computation is uniformly distributed.

As discussed in the original proof of Shabal [7], the abortion events $\mathsf{Abort}_i, i = 1, 2$ correspond to having the adversary generate internal collisions to detect inconsistencies between the hash construction and a random oracle. It can be shown that both $\Pr[\mathsf{Abort}_1]$ in the function case and $\Pr[\mathsf{Abort}_1 \vee \mathsf{Abort}_2]$ in the keyed permutation case are upper-bounded by $2^{-(\ell_a+\ell_h)}N^2$. Putting everything together, we get these new indifferentiability bounds:

Theorem 2. *Let us consider the general mode of operation depicted on Figure 1 with* $n_f \geq 0$ *blank rounds and the simulator* $\mathcal{S}$ *defined on Figure 4. Then for any distinguisher* $\mathcal{D}$ *interacting with* $\mathcal{S}$ *and making at most* N *calls to the compression function* $\mathcal{F}$ *(resp. to the keyed permutation* $(\mathcal{F}, \mathcal{F}^{-1})$*), one has*

$$\mathsf{Adv}(\mathcal{D}) \leq [(1+e)p_C + 2]\, 2^{-(\ell_a+\ell_h)}N^2 + \ell_h p_C 2^{-(\ell_a-1)}N . \quad (1)$$

Moreover, when $n_f = 0$ *and* $\ell_h > \ell_a$, *it also holds that*

$$\mathsf{Adv}(\mathcal{D}) \quad \leq \quad 2^{-(\ell_a+\ell_h)}N^2 + 2^{-\ell_a}N\left\lceil\frac{\ell_h+\ell_a-\log_2(\sqrt{2\pi})}{\ell_h-\ell_a-\log_2 e}\right\rceil \quad (2)$$

and $\mathsf{Adv}(\mathcal{D})$ *is then upper-bounded by the tightest of bounds* (1) *and* (2).

It is worth noticing that when no blank rounds are applied, the above security bounds do not involve the size ℓ_c of the parameter C at all. We comment that this fraction of the internal state improves the security of the construction in two contexts: first, the indifferentiability bound in the variant with blank rounds can be improved, and second it increases resistance against (second)-preimage attacks. Theorem 2 tells us that using a part of the internal state as a parameter can improve the level of indifferentiability in the case of blank rounds, under the condition that the message insertion function Insert satisfies $p_C = 2^{-\ell_c}$. This is obviously not the case for Chop-MD whose insertion function satisfies $p_C = 1$, whereas $p_C = 2^{-\ell_c}$ holds for the insertion function used in Shabal.

When applied to Chop-MD *i.e.*, with $\ell_c = 0$ and $n_f = 0$, Theorem 2 leads to

$$\mathsf{Adv}(\mathcal{D}) \leq (2+e)2^{-n}N^2 + \ell_h 2^{-(n-\ell_h-1)}N .$$

with the improved bound (2) when $\ell_h > n/2$. Our result must be compared to the best previously known indifferentiability bound for Chop-MD due to Chang and Nandi [9] which states that

$$\mathsf{Adv}(\mathcal{D}) \leq 2 \cdot 2^{-n}N^2 + (3\ell_h + 1)2^{-(n-\ell_h)}N + 2^{-(\ell_h-1)}N .$$

Clearly, this bound was relevant only for $\ell_h \geq n/2$ *i.e.*, when at most half of the bits of the chaining value are chopped. Our result then leads to a generalized bound for any possible choice of $\ell_h \in [0, n]$, and it also improves the Chang-Nandi bound in the case $\ell_h \geq n/2$.

4 Capturing distinguishers into indifferentiability proofs

We now show how our general mode of operation can be proven indifferentiable from a random oracle even if the underlying primitive $\mathcal{F}$ does not behave ideally. We focus on the compression function case in the sequel and discuss the keyed permutation case later. In order to capture distinguishers for $\mathcal{F}$ into the indifferentiability framework, we will no longer model $\mathcal{F}$ as a random function uniformly drawn from the space FUNC of all possible functions. Instead $\mathcal{F}$ will be randomly selected from a smaller subspace $\text{FUNC}' \subset \text{FUNC}$ made of all those statistically biased functions which comply with the given distinguishing property. We then extend proofs in the indifferentiability framework by relating the random selection of a function in the subspace FUNC' to the new adversarial capabilities brought by the distinguisher, which we make use in simulators. As an example of particular interest, we provide a new indifferentiability proof for our general domain extender, thereby giving a precise and quantitative security bound expressed as a function of the statistical biases introduced by the distinguisher given on $\mathcal{F}$.

A simple approach would be to adapt the game-hopping proof in the ideal case by adding a game at the beginning of the sequence of games that would capture the fact that the inner primitive is no longer ideal. We then would have to upper-bound the probability under which the adversary distinguishes the first two games of this new sequence, which is equivalent to distinguishing the biased primitive from the ideal one, while being limited to N queries. However doing so leads to vacuous bounds in the case of known practical distinguishers such as the following ones:

- Knudsen, Matusiewicz and Thomsen [11] have shown that for some inputs X of the inner keyed permutation of Shabal, it is possible to find a parameter K_X (that depends on X) such that $\mathcal{F}_{K_X}(X)$ and X have the same B-part.
- The compression function in Hamsi-256 admits differentials of probability one. For instance, it is shown in [8] that for any given K, the outputs corresponding to two values of X which only differ on their bits at positions 71 and 199 have the same bits on positions 228 and 230.
- The fact that the compression function has a low degree (or more generally that the inner keyed permutation consists of several rounds of a low-degree permutation) may lead to the existence of so-called zero-sums distinguishers [5]. A zero-sum of size 2^κ then corresponds to a set of 2^κ inputs $\{X_1, \ldots, X_{2^\kappa}\}$ with $\bigoplus_i X_i = 0$ and $\bigoplus_i F(K, X_i) = 0$. Then, the knowledge of the outputs of $(2^\kappa - 1)$ values in the zero-sum completely determines the output of the remaining value. Such zero-sums exist for the compression function of Hamsi-256 [3] and for the inner permutation of Keccak with 18 rounds [6].

However, intuition tells that these distinguishers have little or no impact on the security of the entire construction since they involve only a very small set of specific inputs and outputs. Clearly however, their existence cannot be captured by the same simulator as in the ideal case: queries which correspond to related inputs enable the adversary to distinguish between $\mathcal{F}$ and the simulator. Therefore, we need to adapt the simulator to the biased case.

We rely on statistical distance to measure how distributions differ. We recall that the statistical distance between two distributions D_1 and D_2 defined over a common set $\mathcal{Z}$ is

$$\|D_1 - D_2\| = \frac{1}{2} \sum_{z \in \mathcal{Z}} \left| \Pr_{Z \leftarrow D_1}[Z = z] - \Pr_{Z \leftarrow D_2}[Z = z] \right| .$$

4.1 An algorithmic representation of biased functions

Besides the existence of marginal input sets which lead to a large statistical distance, it is clear that biases in the distribution of the A'-part and of the B'-part of the outputs of $\mathcal{F}$ have a very different impact on the security of the construction. Since B' is by definition the part of the output of $\mathcal{F}$ which defines the output hash value, only a small bias with respect to the uniform distribution can be safely accepted, whereas the bias on the distribution of the A'-part of outputs may be less severely limited.

Recall that the compression function $\mathcal{F}$ used in the general mode of operation has the form $\mathcal{F}(M, A, B, C) = (A', B')$.

Definition 1 (Admissible output)**.** *Let $\mathcal{L}$ be a list of input/ouput pairs of $\mathcal{F}$ and let* $\mathrm{FUNC}'[\mathcal{L}]$ *denote the set of all functions $\mathcal{F}$ in* FUNC' *such that $\mathcal{F}(u) = v$ for all $(u, v) \in \mathcal{L}$. For a given input (M, A, B, C), we will say that an output B' is* admissible *if there exists some $\mathcal{F} \in \mathrm{FUNC}'[\mathcal{L}]$ such that $\mathcal{F}_{M,C}(A, B) = (A', B')$ for some $A' \in \{0,1\}^{\ell_a}$. The set of all admissible B' is denoted by* $\mathrm{Adm}(M, A, B, C, \mathcal{L})$.

We will assume that an efficient distinguisher for $\mathcal{F}$ is given under the form of efficient subroutines comprising

- a sampling algorithm `AdmSamp` which takes as input any (M, A, B, C) and any history $\mathcal{L}$ and samples all possible outputs B' such that $B' \in \mathrm{Adm}(M, A, B, C, \mathcal{L})$ over a uniformly random choice of $\mathcal{F} \leftarrow \mathrm{FUNC}'[\mathcal{L}]$.
- a sampling algorithm `ForSamp` which takes as input any $\mathcal{L}$ and any (M, A, B, C, B'), and samples all possible outputs A' over a uniformly random choice of $\mathcal{F} \leftarrow \mathrm{FUNC}'[M, A, B, C, B', \mathcal{L}]$ in the subset of all functions $\mathcal{F}$ in $\mathrm{FUNC}'[\mathcal{L}]$ such that $\mathcal{F}(M, A, B, C) = (A', B')$ for some A', if $B' \in \mathrm{Adm}(M, A, B, C, \mathcal{L})$. When $B' \notin \mathrm{Adm}(M, A, B, C, \mathcal{L})$, we impose `ForSamp` outputs a uniformly random value for A'.

We will assume that given a set FUNC', one can construct such testing and sampling algorithms and that they can be implemented efficiently. These routines are viewed as an algorithmic representation of FUNC' and the distinguishers on $\mathcal{F}$ can be reformulated by making their sampling algorithms explicit.

4.1.1 Relatives of a set of inputs and atypical inputs

As previously discussed, the existence of distinguishers with large success probability (or even with probability one) is quite common for practical implementations of $\mathcal{F}$, but this usually does not threaten the security of the hash construction. Such distinguishers correspond to some sets of input/ouput pairs the knowledge of which provides some information on the outputs of related inputs. For instance (informally), if there is a differential distinguisher with probability one for $\mathcal{F}$, *i.e.*, there exists (α, β) such that $\mathcal{F}(x + \alpha) = \mathcal{F}(x) + \beta$ for all x, then the knowledge of any pair (x, y) such that $\mathcal{F}(x) = y$ discloses the value of $\mathcal{F}(x + \alpha)$. The inputs which output distribution is strongly biased by the knowledge of the history $\mathcal{L}$ must then be handled separately. Those inputs, which we call the *relatives* of the inputs in $\mathcal{L}$, are defined as follows, for a given threshold on the bias on the output distribution which is fixed by the simulator.

Definition 2 (ε-relatives of a set of inputs)**.** *Let ε be a given real in $[0, 1]$. Let X be set of inputs in $\{0,1\}^{n+\ell_m}$. We define the set of all ε-relatives of X as follows*

$$\mathtt{REL}(X, \varepsilon) = \{u = (M, A, B, C) \notin X \text{ such that } ||D_B(u, \mathcal{L}) - U|| > \varepsilon,\ \forall \mathcal{L} \text{ with } \{x,\ (x, y) \in \mathcal{L}\} = X\}\ ,$$

where U is the uniform distribution over $\{0,1\}^{\ell_h}$ and $D_B(u, \mathcal{L}) = \{B' \leftarrow \mathtt{AdmSamp}(u, \mathcal{L})\}$.

For instance, in the case of the previously mentioned differential distinguisher, we have $\mathtt{REL}(X, \varepsilon) \supset \{x + \alpha, x \in X\}$ for any $\varepsilon < 1$. For a given value of ε, the inputs in the set $\mathtt{REL}(\emptyset, \varepsilon)$ play a very particular role. They correspond to the inputs which lead to a B' whose distribution highly differs from the uniform distribution, even if no further information is known on $\mathcal{F}$. For instance, for the inner permutation of Shabal, these inputs include the "partial fixed points" exhibited in [11], *i.e.*, some inputs which, under a very particular but explicit condition, lead to an output with $B' = B$. It turns out that these inputs have a single admissible B'.

Definition 3 (ε-atypical inputs.)**.** *Let ε be a given real in $[0, 1]$. The* ε-atypical inputs *for* FUNC' *are the inputs*

$$\mathtt{AT}(\varepsilon) = \mathtt{REL}(\emptyset, \varepsilon) = \{u = (M, A, B, C),\ ||D_B(u, \emptyset) - U|| > \varepsilon\}\ .$$

Within the simulator, atypical inputs will receive a special treatment: since the statistical distribution of the B-part of the corresponding outputs significantly differs from the uniform distribution, they cannot be consistent with the output of a random oracle. Therefore we must guarantee that in the graph constructed by the simulator, there is no state that admits a path which can lead to an atypical input after the insertion of some new message block. Hence we assume the existence of a test algorithm ε-`LeadToAtypTest` which checks for a given tuple (A', B', C) whether there exists some $m \in \{0,1\}^{\ell_m}$ such that

$$(m, \mathsf{Insert}[m](A', B', C)) \in \mathtt{AT}(\varepsilon) \ .$$

The previous definitions enable us to define, for each $x \notin X$, the set of all ε-relatives of x with respect to X:

Definition 4 (ε-relatives of an input)**.** *Let ε be a given real in $[0,1]$. Let X be set of inputs in $\{0,1\}^{n+\ell_m}$ and $x \in \{0,1\}^{n+\ell_m}$ be an input not belonging to X. We define the set of all ε-relatives of x, denoted by $\mathcal{R}(x, X, \varepsilon)$, as the smallest set S of inputs such that $x \in S$ and*

$$\mathtt{REL}(X \cup S, \varepsilon) \subset \mathtt{AT}(\varepsilon) \ .$$

We can check that this definition induces a symmetric relation: for any set of inputs X and any (x, x') not in X, we have $x' \in \mathcal{R}(x, X, \varepsilon)$ if and only if $x \in \mathcal{R}(x', X, \varepsilon)$. Now, the basic idea is that, at each new query to the simulator, not only the image of the query is added to the graph, but also the images of all relatives of the query since they are strongly constrained (maybe even completely determined) by the knowledge of the output of the query and of the history. However, problems may arise when the outputs of two relatives of the same value must both be made consistent with the random oracle while they are heavily correlated by the definition of the notion of relatives. This situation occurs when the graph constructed during the simulation contains two states x and x' both with a path from x_0, such that the insertion of some message block leads to relative inputs *i.e.*, such that there exist $m, m' \in \{0,1\}^{\ell_m}$ with

$$(m, \mathsf{Insert}[m](x)) \in \mathcal{R}(m', \mathsf{Insert}[m'](x'), \mathcal{L}, \varepsilon) \ .$$

Again, we assume that this property on x and x' can be tested efficiently by a test algorithm ε-`LeadToRelTest` which decides whether a pair $m, m' \in \{0,1\}^{\ell_m}$ satisfying the previous property exists.

In the rest of the paper, we focus on the case where a threshold ε can be chosen such that the set all of ε-relatives of a given input does not depend on the previous inputs *i.e.*,

$$\mathcal{R}(x, X, \varepsilon) = \mathcal{R}(x, X', \varepsilon) \text{ for all } X \text{ and } X' \ ,$$

where the sizes of both sets X and X' are small compared to $2^{\ell_a + \ell_h}$. For this reason, the parameter X will be omitted when defining $\mathcal{R}(x, X, \varepsilon)$. This situation holds in particular when any relation between the inputs and outputs of $\mathcal{F}$ which allows to distinguish between $\mathcal{F}$ and an ideal function involves at most two input/output pairs. In other words, we restrict ourselves to the existence of distinguishers of degree 1 or 2. Most notably, this includes the existence of atypical inputs, and of linear, differential or rotational distinguishers.

4.1.2 Defining the bias on the distribution of the truncated part

The previous notions aim at quantifying the bias on the distribution of the B-part of the output of $\mathcal{F}$. Since only the B-part of the internal state is chopped when the hash value is computed, we can use a less restrictive metric for estimating how severe is the bias on the A-part of the output of $\mathcal{F}$.

Definition 5 (Forward bias)**.** *The forward bias is the smallest real $\tau(\varepsilon) \in [0, \ell_a]$ such that, for any $\mathcal{L}$, for any input $u = (M, A, B, C)$ such that $||D_B(u, \mathcal{L}) - U|| \leq \varepsilon$ where U is the uniform distribution over $\{0,1\}^{\ell_h}$, for any $B' \in \mathrm{Adm}(M, A, B, C, \mathcal{L})$, it holds that*

$$\max_{A' \in \{0,1\}^{\ell_a}} \Pr\left[\mathtt{ForSamp}(M, A, B, C, B', \mathcal{L}) = A'\right] \leq 2^{-(\ell_a - \tau(\varepsilon))} \ ,$$

where the probability is taken over the internal coins of `ForSamp` *in the random selection of $\mathcal{F} \leftarrow \mathrm{FUNC}[\mathcal{L}]$ with $\mathcal{F}(M, A, B, C) = (A', B')$ for some A'.*

4.2 An indifferentiability proof with distinguishers

In this section we summarize the main ideas of the security proof of Shabal with distinguishers. A more detailed version of the game-hopping proof can be found in the full version of this paper.

Original Game. This is the original game where $\mathcal{D}$ interacts with $\mathcal{C}^{\mathcal{F}}$ and with the function $\mathcal{F}$, which is uniformly chosen at random in FUNC$'$. By definition of Game 0, we have

$$\Pr[W_0] = \Pr\left[\mathcal{D}^S = 1 \mid S = (\mathcal{C}^{\mathcal{F}}, \mathcal{F})\right] .$$

Games 1–2. We replace $\mathcal{F}$ by a simulator $\mathcal{S}$ with the same interface as $\mathcal{F}$. To make sure that the answers $\mathcal{S}$ gives to the different queries follow the same distribution than the answers $\mathcal{F}$ returns in the original game, we make use of the sampling algorithms `AdmSamp` et `ForSamp`. More precisely, to compute $\mathcal{F}(M, A, B, C)$ if it has not already been defined, the simulator computes $B' \leftarrow \texttt{AdmSamp}(M, A, B, C, \mathcal{L})$ and $A' \leftarrow \texttt{ForSamp}(M, A, B, C, B', \mathcal{L})$. The definition of the sampling algorithms, and in particular the fact that their output depends on past queries, guarantees that the output distribution is the same as in the original game. We therefore have:

$$\Pr[W_2] = \Pr[W_0] .$$

Games 3–7. In Games 1 and 2 the only concern was to keep the same distribution as in Game 0 for the answers to the requests to $\mathcal{F}$. As the hash requests will be replaced by calls to a random oracle in later games, we now have to modify the simulator to make sure it is consistent with the answers given by the random oracle.

To achieve this, the first modification of $\mathcal{S}$ is to make $\mathcal{S}$ compute the value of $\mathcal{F}$ simultaneously on the requested input (M, A, B, C) and on all its ε-relatives, because their distribution once $\mathcal{F}(M, A, B, C)$ is defined is too far from the uniform distribution. The idea is that after the replacement of $\mathcal{C}$ by the random oracle, $\mathcal{S}$ can check if one of these values is constrained by the random oracle, and define all the outputs of $\mathcal{F}$ simultaneously according to the answer of the random oracle.

We then have to identify the cases in which the simulator cannot be consistent with the answers of the random oracle. We then define the following events:

- Abort_a : if $(\mathcal{F}(M, A, B, C), C)$ or $\mathcal{F}((M', A', B', C'), C')$ collides with a value of the state that was already the start or the end of an edge, we might be in the case in which two paths starting from x_0 lead to the same value of the state, or in the case where a path starting from x_0 reaches a state from which the compression function has already been computed.

- Abort_b : also, we upper bound the number of ε-relatives of each state by R_{MAX}. Therefore $\mathcal{S}$ aborts if the number of ε-relatives of (M, A, B, C) exceeds R_{MAX}. In other words, we have to assume that if some states have more than R_{MAX} relatives, the distinguisher cannot identify them offline.

- Abort_c : if $\mathcal{S}$ computes $\mathcal{F}(M, A, B, C)$ and if there exists M^*, M' and a state (A', B', C') that has a path in the graph such that $(M^*, \mathsf{Insert}[M^*](\mathcal{Q}(M, A, B, C), C))$ and $(M', \mathsf{Insert}[M'](A', B', C'))$ are ε-relatives, we might be in the case in which the distinguisher can find two ε-relatives that both have a path in the graph.

- Abort_d : $\mathcal{S}$ outputs a value for $(A', B') = \mathcal{F}(M, A, B, C)$ such that (A', B', C) can be transformed into an atypical input by an appropriate message insertion. Actually, the outputs corresponding to atypical inputs cannot be chosen to be consistent with the random oracle. Moreover, it must be assumed that the initial state x_0 cannot be transformed into an atypical value by inserting some message block.

Applying the difference lemma we then have:

$$|\Pr[W_7] - \Pr[W_2]| \leq \Pr[\mathsf{Abort}_a] + \Pr[\mathsf{Abort}_b] + \Pr[\mathsf{Abort}_c] + \Pr[\mathsf{Abort}_d] .$$

Game 8. In Game 8 we add the random oracle $\mathcal{H}$. Instead of defining a completely random response, the simulator will rather make a call to $\mathcal{H}$ to let $\mathcal{H}$ define the B-part of the output if the input (M, A, B, C) has a relative which has a path in the graph. Provided the previously described abortion cases do not occur, the value returned by `AdmSamp` for B' follows a distribution D which distance to the uniform distribution is smaller than ε. After N requests the distance between the uniform distribution and the outputs of `AdmSamp` is then smaller than $N\varepsilon$. We then have:

$$|\Pr[W_8] - \Pr[W_7]| \leq N\varepsilon \ .$$

Games 9–10. In the final game, the simulator does not have access to the hash requests issued by the distinguisher. We therefore define a new abortion event, Abort_d, that occurs when $\mathcal{S}$ notices that it has to use the part of its memory that was build to answer hash requests. This is equivalent to the length-extension attacks on the Merkle-Damgård mode. We therefore have:

$$|\Pr[W_{10}] - \Pr[W_8]| \leq \Pr[\mathsf{Abort}_e] \ .$$

Games 11–12. We remove the access $\mathcal{S}$ had to hash requests by $\mathcal{D}$, and reach the final Game. We have:

$$\Pr[W_{12}] = \Pr\left[\mathcal{D}^S = 1 \mid S = (\mathcal{H}^{\mathcal{S}}, \mathcal{S})\right] = \Pr[W_{10}] \ .$$

This leads to the simulator depicted on Figure 5.

4.2.1 Bounding the success probability of the distinguisher

Putting all these results together we get:

$$\begin{aligned}\mathsf{Adv}(\mathcal{D}) &= |\Pr\left[\mathcal{D}^S = 1 \mid S = (\mathcal{H}^{\mathcal{S}}, \mathcal{S})\right] - \Pr\left[\mathcal{D}^S = 1 \mid S = (\mathcal{C}^{\mathcal{F}}, \mathcal{F})\right]| \\ &\leq \Pr[\mathsf{Abort}_a] + \Pr[\mathsf{Abort}_b] + \Pr[\mathsf{Abort}_c] + \Pr[\mathsf{Abort}_d] + \Pr[\mathsf{Abort}_e] + N\varepsilon \ .\end{aligned}$$

We now give upper bounds for all these abortion probabilities, involving the following quantities and probability p_C defined in Theorem 1.

$$\begin{aligned}R_{\max} &= \max_{u \in \{0,1\}^{n+\ell_m}} |\mathcal{R}(u, \varepsilon)| \\ \mathcal{A} &= \max_{y \in \mathcal{X}, C \in \{0,1\}^{\ell_c}} |\{(a', b') : \mathsf{Insert}^{-1}[\tilde{m}](\tilde{v}) = (a', b', C),\ (\tilde{m}, \tilde{v}) \in \mathcal{R}(m, \mathsf{Insert}[m](y)),\ m \in \{0,1\}^{\ell_m}\}| \\ \mathcal{B} &= \max_{C} |\{(a, b) :\ \mathsf{Insert}^{-1}[\tilde{M}](\tilde{A}, \tilde{B}, \tilde{C}) = (a, b, C),\ (\tilde{M}, \tilde{A}, \tilde{B}, \tilde{C}) \in \mathtt{AT}(\varepsilon)\}\end{aligned}$$

The computation of these bounds, detailed in the full version of this paper, lead to:

$$\begin{aligned}\Pr[\mathsf{Abort}_a] &\leq 2^{-(\ell_a - \tau(\varepsilon))}\left(2^{-\ell_h} + 4\varepsilon\right)\left(R_{\max} N^2\right) \\ \Pr[\mathsf{Abort}_b] &\leq \max_{M,y}(\Pr[|\mathcal{R}(M, A, B, C, \varepsilon)| > R_{MAX}])N \\ \Pr[\mathsf{Abort}_c] &\leq \mathcal{A} \cdot 2^{-(\ell_a - \tau(\varepsilon))}\left(2^{-\ell_h} + 4\varepsilon\right)\frac{N(N+1)}{2} \\ \Pr[\mathsf{Abort}_d] &\leq \mathcal{B} \cdot 2^{-(\ell_a - \tau(\varepsilon))}\left(2^{-\ell_h} + 4\varepsilon\right)N \\ \Pr[\mathsf{Abort}_e] &= N^2 2^{-(\ell_a - \tau(\varepsilon))}\left[2^{-\ell_h}(1-\varepsilon)^{-1} + 2^{-\ell_h} p_C(1+e) + p_C\varepsilon\right] + N 2^{-(\ell_a - \tau(\varepsilon) - 1)}\ell_h p_C \ ,\end{aligned}$$

We then deduce the following theorem.

Initialization of $\mathcal{S}$

1. choose $\varepsilon \in [0,1]$
2. set $X = \{x_0\}$ and $E = \emptyset$

Simulation of $\mathcal{F}$

Input: (M, A, B, C)

Output: (A', B')

1. set $x = \mathsf{Insert}^{-1}[M](A, B, C)$
2. if there exists an edge $x \xrightarrow{M} y \in E$
   (a) return (A', B') where $y = (A', B', C)$
3. if $|\mathcal{R}(M, A, B, C, \varepsilon)| > R_{MAX}$ (event Abort_b), then abort.
4. if there exists (M^*, A^*, B^*, C^*) in $\mathcal{R}(M, A, B, C, \varepsilon)$ which has a path μ in the graph $\mathcal{G}$
   (a) set $x^* = \mathsf{Insert}^{-1}[M^*](A^*, B^*, C^*)$
   (b) compute $\mathcal{M} = \mathsf{unpad}(\mu || M^*)$
   (c) call $\mathcal{H}$ to get $h = \mathcal{H}(\mathcal{M})$
   (d) set $B'^* = h$
   (e) run $\mathtt{ForSamp}(M^*, A^*, B^*, C^*, B'^*, E)$ to get A'^*
   (f) set $y^* = (A'^*, B'^*, C^*)$
   (g) if $y^* \in X$ (event Abort_a), then abort
   (h) if there exists y' with a path in $\mathcal{G}$ such that ε-$\mathtt{LeadToRelTest}(y^*, y')$ (event Abort_c), then abort
   (i) if ε-$\mathtt{LeadToAtypTest}(y^*)$ (event Abort_d), then abort
   (j) add node x^* and y^* to X and edge $x^* \xrightarrow{M^*} y^*$ to E
5. for all $(\tilde{M}, \tilde{A}, \tilde{B}, \tilde{C}) \in \mathcal{R}(M, A, B, C, \varepsilon) \setminus \{(M^*, A^*, B^*, C^*)\}$
   (a) set $\tilde{x} = \mathsf{Insert}^{-1}[\tilde{M}](\tilde{A}, \tilde{B}, \tilde{C})$
   (b) run $\mathtt{AdmSamp}(\tilde{M}, \tilde{A}, \tilde{B}, \tilde{C}, E)$ to get $\tilde{B}'$
   (c) run $\mathtt{ForSamp}(\tilde{M}, \tilde{A}, \tilde{B}, \tilde{C}, \tilde{B}', E)$ to get $\tilde{A}'$
   (d) set $\tilde{y} = (\tilde{A}', \tilde{B}', \tilde{C})$
   (e) add nodes $\tilde{x}$ and $\tilde{y}$ to X and edge $\tilde{x} \xrightarrow{\tilde{M}} \tilde{y}$ to E
6. return (A', B') where (A', B') are the A- and B-parts of the output state corresponding to $x = \mathsf{Insert}^{-1}[M](A, B, C)$.

Figure 5: Simulator $\mathcal{S}$ for $\mathcal{F}$ in the biased case.

Theorem 3. *Assume that the initial state x_0 is such that* $\texttt{LeadToAtypTest}(x_0)$ *equals false. Assume $\mathcal{F}$ is a random function in* FUNC$'$ *and let $\mathcal{H}$ be a random oracle. Then for any distinguisher $\mathcal{D}$ interacting with $\mathcal{S}$ and making at most N calls to $\mathcal{F}$, one has*

$$\begin{aligned}\mathsf{Adv}(\mathcal{D}) \quad \leq \quad & N^2 2^{-(\ell_a-\tau(\varepsilon))}\left\{2^{-\ell_h}\left[R_{\max}+(1-\varepsilon)^{-1}+p_C(1+e)+\frac{\mathcal{A}}{2}\right]+\varepsilon\left[p_C+4R_{\max}+2\mathcal{A}\right]\right\} \\ & +N\left[\mathcal{B}\cdot 2^{-(\ell_a-\tau(\varepsilon))}\left(2^{-\ell_h}+4\varepsilon\right)+2\varepsilon+2^{-(\ell_a-\tau(\varepsilon))}\ell_h p_C+\max_{M,y}(\Pr\left[|\mathcal{R}(M,A,B,C,\varepsilon)|>R_{MAX}\right])\right] .\end{aligned}$$

5 Application to Shabal

In this section we show how the indifferentiability proof with distinguishers applies to the hash function Shabal. As shown in the previous sections, the indifferentiability bound depends on the nature of the distinguishers we take into account. We therefore consider the distinguishers that are known at the time we write these lines. Further cryptanalytic headway could lead to lower security bounds.

5.1 The inner permutation of Shabal.

In this paper the description of the Shabal domain extension algorithm slightly differs from the one that was described in the submission document [7]. Therefore, the description of the permutation is also slightly different to obtain the same hash function. The message block M is divided into sixteen 32-bit words $M[0]..M[15]$. T_a consists of twelve 32-bit words $T_a[0]..T_a[11]$, and similarly T_b and C contain sixteen 32-bit words each, $T_b[0]..T_b[15]$ and $C[0]..C[15]$. The permutation $\mathcal{Q}$ can then be derived from the permutation $\mathcal{P}$ described in the submission document by adding the following step at the beginning of the computation:

$$\texttt{Forall}\ i \in [0..15], C[i] \leftarrow C[i] \boxplus M[i] .$$

We also have to compute the inverse operation at the end of the permutation so that C is not modified:

$$\texttt{Forall}\ i \in [0..15], C[i] \leftarrow C[i] \boxminus M[i] .$$

5.2 Known distinguishers for Shabal

Our analysis is based on four different distinguishers for the keyed permutation of Shabal.

Non-dependencies for the inverse permutation. In [13], Naya-Plasencia shows that when computing the inverse permutation $(T_a, T_b) = \mathcal{Q}^{-1}(M, T'_a, T'_b, C)$, $T_b[11]$ (resp. $T_b[14]$, $T_b[15]$) depends only on words $0, 4, 8, 12$ of M through $(M[0], M[4]+M[8], M[12])$ (resp. $(M[0]+M[4], M[8], M[12])$,$(M[0]+M[4]+M[12], M[4]+M[8]+M[12])$). Therefore, the knowledge of $\mathcal{Q}^{-1}(M, T'_a, T'_b, C)$ implies the knowledge of $T_b[11]$ (resp. $T_b[14]$, $T_b[15]$) of $\mathcal{Q}^{-1}(M', T'_a, T'_b, C)$ where M and M' differ only on values that are not involved in the computation of $T_b[11]$ (resp. $T_b[14]$, $T_b[15]$). As a result, one can know up to three output words of $\mathcal{Q}^{-1}$ before computing the inverse permutation, and the number of possible preimages for $\mathcal{Q}$ for a given (M, A', B', C) can in some cases be only $2^{\ell_a+\ell_h-96}$.

This property means that for a given (M, C), the probability that the preimage of (A', B') is (A, B) is either 0 or $2^{-(\ell_a+\ell_h-96)}$. This also means that the image of (A, B) by $\mathcal{Q}$ is (A', B') with probability 0 or $2^{-(\ell_a+\ell_h-96)}$. We can then assume that B' follows a uniform distribution ($\varepsilon = 0$), and that the forward bias is $\tau(\varepsilon) = 96$.

Differential distinguishers. In [4], Aumasson *et al.* showed that some differential properties hold with probability 1 on $\mathcal{Q}$. More precisely, if a bit δ_i is xored to the most significant bit of each word $M[i]$ of M, and $\delta_i \oplus \delta_{8-i}$ to each word $C[i]$ of C, (A, B) is not modified during the 3 rounds of $\mathcal{Q}$. Only the most significant bits of the U-part of the state are modified (deterministically) by the final update. As a result if we write $\Delta = (\delta_i 0...0)_{i=0..15}$ and $f(\Delta) = (\Delta_i \oplus \Delta_{8-i})$, we have that (M, A, B, C) and $((M \oplus \Delta), A, B, C \oplus f(\Delta))$ are

relatives with probability 1. As one has 2^{16} choices for Δ, each input (M, A, B, C) of $\mathcal{Q}$ defines a set of 2^{16} values, which are all relatives to each others independently of ε and $\mathcal{L}$.

Rotational distinguishers. In [1], van Assche shows that the relation

$$\mathcal{Q}(M \ggg 1, T_a \ggg 1, T_b \ggg 1, C \ggg 1) = \mathcal{Q}(M, T_a, T_b, C) \ggg 1$$

holds with probability up to 2^{-159} over all the values of (M, T_a, T_b, C). Therefore, each (M, T_a, T_b, C) has $(M \ggg 1, T_a \ggg 1, T_b \ggg 1, C \ggg 1)$ as a relative with probability 2^{-159}. Furthermore, these relations can be combined with the differential distinguishers to obtain larger sets of relatives. Assuming that these rotational events happen independently from each other, we can show that the number of relatives of (M, T_a, T_b, C) that can be reached by applying the rotational and differential distinguishers are bounded by 4×2^{16} with probability at least $1 - 2^{-564}$. This is true when the set of relatives of (M, T_a, T_b, C) contains at most 3 rotational pairs. For each relative (M', T'_a, T'_b, C') of (M, T_a, T_b, C), we can try the rotational distinguisher with a left- or a right 1-bit rotation. Therefore, for a set of 4×2^{16} relatives of (M, T_a, T_b, C), we can build at least four rotational pairs with probability at most $2^{19\times4}/4!2^{-159\times4} \leq 2^{-564}$. We then have $\Pr[|\mathcal{R}(M, A, B, C, \varepsilon)| > R_{MAX}] < 2^{-564}$.

Fixed points. In [11], Knudsen *et al.* found a set of atypical states for Shabal. More precisely, there are 2^{128} values of (A, B) such that for each value of M, there exists exactly one value of C such that the T_b-part of $\mathcal{Q}(M, T_a, T_b, C)$ is equal to T_b. As the value of the T_b-part of the output of $\mathcal{Q}$ is fixed, and the T_b part of the chaining value contains its B-part, these states are atypical whatever the value of ε. It can easily be seen that the differential and rotational relatives of a fixed point are also fixed points. Therefore, the results by Van Assche and Aumasson *et al.* do not increase the number of atypical states.

5.3 Computing the security bound for Shabal.

Defining ε and $\tau(\varepsilon)$. In the case of Shabal, considering the distinguishers described above, the output of `AdmSamp` is either set to a fixed value or uniformly distributed. Therefore we can set $\varepsilon = 0$. The distinguisher on the inverse permutation applies for every state, therefore we have to set $\tau(0) = 96$. $R_{\max}$ is the maximum number of relatives a state can have, which we establish to be 2^{18}.

Computing $\mathcal{A}$. We also need to compute $\mathcal{A}$ which can be informally defined as the maximum number of states (a', b', c') colliding on the C-part that can be obtained by starting from a given state $y = (a, b, c)$, choosing a message block m, computing $(m, A, B, C) = (m, \mathsf{Insert}[m](a, b, c))$, choosing a relative (m', A', B', C') of (m, A, B, C) and computing $(a', b', c') = \mathsf{Insert}^{-1}[m'](A', B', C')$.

The exact computation of $\mathcal{A}$ can be found in the extended version of this paper. The main idea is the following. We try to bound the number of (t'_a, t'_b) that one can reach doing such operations, starting from (t_a, t_b, c). Relatives of (m, T_a, T_b, C) are obtained by combining rotations on the state by α positions (between -3 and 3) and xoring of at most four 16-bit masks Δ and $f(\Delta)$ on m and C. Then the following holds:

$$\begin{aligned} t'_a &= t_a \ggg \alpha \\ t'_b &= (t_b \boxminus m) \ggg \alpha \boxplus ((m \ggg \alpha) \oplus D) \end{aligned}$$

For a given value of α, t'_a is fixed. One can then express $(t_b \boxminus m) \ggg \alpha$ as a function of $(t_b \ggg \alpha) \boxminus (m \ggg \alpha)$ and two carry bits. One can also express $((m \ggg \alpha) \oplus D)$ as $(m \ggg \alpha) \boxplus D'$, where D' can only take a small number of values. Putting it together, the values of m are cancelled, and t'_b depends on t_b, 2 carry bits and D', which can take at most $3^{4\cdot16}$ values for each rotation.

$\mathcal{A}$ is bounded by the total number of (t'_a, t'_b) one can obtain by such operations, therefore

$$\mathcal{A} \leq 7 \times 2^{32} \times 3^{64} \leq 2^{136.25} .$$

Computing $\mathcal{B}$. We recall that $\mathcal{B}$ is the maximal number of different states $x = \mathsf{Insert}^{-1}[\tilde{M}](\tilde{T}_a, \tilde{T}_b, \tilde{C})$ with a given C-part, when $u = (\tilde{M}, \tilde{T}_a, \tilde{T}_b, \tilde{C})$ varies in the set of atypical inputs. By definition of the insertion function, we have $x = (t_a, t_b, c)$ with $c = \tilde{T}_b \boxminus \tilde{M}$. Then, for a given value of c, there are exactly 2^{128} values of $\tilde{M}$ such that $c \boxplus \tilde{M}$ corresponds to the T_b-part of an atypical input. Moreover, each of these 2^{128} values of $\tilde{M}$ corresponds to a single atypical input. Therefore, there are 2^{128} atypical inputs u which lead to values of x with the same C-part. It follows that $\mathcal{B} = 2^{128}$.

Computing p_C. We now estimate the probability

$$p_C = \max_{u \in \mathcal{X}, \mathcal{L}, m, C, C'} \Pr_{\substack{b \leftarrow \mathrm{Adm}(m,u,\mathcal{L}) \\ a \leftarrow \mathtt{ForSamp}(m,u,b,\mathcal{L})}} [\mathsf{Insert}_C[m](a,b,C') = C] \ .$$

The insertion function in Shabal satisfies

$$\mathsf{Insert}_C[M](A,B,C) = T_b \boxminus M = (A_2 \boxminus M_1, B \boxminus M_2)$$

where A_2 denotes the last $(\ell_m - \ell_h)$ bits of A and M_1 (resp. M_2) denotes the first $(\ell_m - \ell_h)$ bits (resp. the last ℓ_h bits) of M. For a fixed message block, we then have

$$\begin{aligned} p_C &= \max_{u \in \mathcal{X}, \mathcal{L}, m, C} \Pr_{b \leftarrow \mathrm{Adm}(m,u,\mathcal{L})}[b \boxminus m_2 = C_2] \Pr_{a \leftarrow\leftarrow \mathtt{ForSamp}(m,u,b,\mathcal{L})}[a_2 \boxminus m_1 = C_1] \\ &= \max_{u \in \mathcal{X}, \mathcal{L}, m, C} |\mathrm{Adm}(m,u,\mathcal{L})| 2^{-(\ell_m - \ell_h - \tau(\varepsilon))} \leq 2^{-(\ell_m - \tau)}(1+\varepsilon)^{-1} \ . \end{aligned}$$

It follows that, for the parameters used in Shabal and $\varepsilon = 0$, one has $p_C = 2^{-416}$. Putting it together, with the same notation as in Theorem 3, we get:

$$\begin{aligned} \Pr[\mathsf{Abort}_a] &\leq 2^{-800} R_{\max} N^2 \leq 2^{-782} N^2 \\ \Pr[\mathsf{Abort}_b] &\leq 2^{-564} N \\ \Pr[\mathsf{Abort}_c] &\leq 2^{-801} \mathcal{A} N(N+1) \leq 2^{-664}(N^2 + N) \\ \Pr[\mathsf{Abort}_d] &\leq 2^{-800} \mathcal{B} N \leq 2^{-672} N \\ \Pr[\mathsf{Abort}_e] &\leq 2^{-800}\left[1 + 2^{-416}(1+e)\right] N^2 + \ell_h 2^{-(\ell_a - 96 + 416)} N \leq (2^{-800} + 2^{-1214}) N^2 + \ell_h 2^{-704} N \ . \end{aligned}$$

We can then conclude that, for any $\ell_h \leq 512$, Shabal remains indifferentiable from a random oracle up to $N = 2^{332}$ requests.

References

[1] G. Van Assche. A rotational distinguisher on Shabal's keyed permutation and its impact on the security proof. NIST mailing list, 2010.

[2] J.-P. Aumasson. On the pseudorandomness of Shabal's keyed permutation. Public comment on the NIST Hash competition, 2009.

[3] J.-P. Aumasson, E. Käsper, L.R. Knudsen, K. Matusiewicz, R. Odegaard, T. Peyrin, and M. Schläffer. Differential Distinguishers for the Compression Function and Output Transformation of Hamsi-256. Cryptology ePrint Archive, Report 2010/091, 2010.

[4] J.-P. Aumasson, A. Mashatan, and W. Meier. More on the pseudorandomness of Shabal's permutation. Public comment on the NIST Hash competition, 2009.

[5] J.-P. Aumasson and W. Meier. Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi. presented at the rump session of Cryptographic Hardware and Embedded Systems - CHES 2009, 2009.

[6] C. Boura and A. Canteaut. A Zero-Sum property for the Keccak-f Permutation with 18 Rounds. NIST mailing list, 2010.

[7] E. Bresson, A. Canteaut, B. Chevallier-Mames, C. Clavier, T. Fuhr, A. Gouget, T. Icart, J.-F. Misarsky, M. Naya-Plasencia, P. Paillier, T. Pornin, J.-R. Reinhard, C. Thuillet, and M. Videau. Shabal, a submission to NIST'cryptographic hash algorithm competition. Submission to the NIST Hash competition, 2008.

[8] C. Calik and M.S. Turan. Message Recovery and Pseudo-Preimage Attacks on the Compression Function of Hamsi-256. Cryptology ePrint Archive, Report 2010/057, 2010.

[9] D. Chang and M. Nandi. Improved indifferentiabily security analysis of chopMD hash function. In *Fast Software Encryption - FSE 2008*, volume 5086 of *Lecture Notes in Computer Science*, pages 429–443. Springer, 2008.

[10] J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Damgård revisited: how to construct a hash function. In *Advances in Cryptology — CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer, 2005.

[11] L. Knudsen, K. Matusiewicz, and S.S. Thomsen. Observations on the Shabal keyed permutation. Public comment on the NIST Hash competition, 2009.

[12] U. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In *Theory of cryptography – TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2004.

[13] M. Naya-Plasencia. *Chiffrement par flot et fonctions de hachage : conception et cryptanalyse.* PhD thesis, University Paris 6, France, 2009.