cryptography

MDPI

Article

Sequential Hashing with Minimum Padding

Shoichi Hirose

Faculty of Engineering, University of Fukui, Fukui 910-8507, Japan; hrs_shch@u-fukui.ac.jp

Received: 10 May 2018; Accepted: 7 June 2018; Published: 10 June 2018

Abstract: This article presents a sequential domain extension scheme with minimum padding for hashing using a compression function. The proposed domain extension scheme is free from the length extension property. The collision resistance of a hash function using the proposed domain extension is shown to be reduced to the collision resistance and the everywhere preimage resistance of the underlying compression function in the standard model, where the compression function is assumed to be chosen at random from a function family in some efficient way. Its indifferentiability from a random oracle up to the birthday bound is also shown on the assumption that the underlying compression function is a fixed-input-length random oracle or the Davies-Meyer mode of a block cipher chosen uniformly at random. The proposed domain extension is also applied to the sponge construction and the resultant hash function is shown to be indifferentiable from a random oracle up to the birthday bound in the ideal permutation model. The proposed domain extension scheme is expected to be useful for processing short messages.

Keywords: hash function; domain extension; collision resistance; indifferentiability

1. Introduction

1.1. Background

A cryptographic hash function takes as input a sequence of arbitrary length and produces as output a sequence of fixed length. It usually consists of a primitive and a domain extension scheme. A primitive is a compression function or a permutation, which takes a fixed-length input and produces a fixed-length output. A domain extension scheme specifies how to process an input sequence with arbitrary length using a primitive with fixed input length.

The standardized hash functions SHA-2 [1] use dedicated compression functions and a domain extension scheme due to Merkle [2] and Damgård [3]. The domain extension scheme is called strengthened Merkle-Damgård (SMD). It is a sequential iteration of a compression function and its padding algorithm appends the binary representation of the length of an input message, which is called MD strengthening.

A positive point of SMD is its preservation of collision resistance. Namely, a hash function using SMD satisfies collision resistance if its underlying compression function satisfies it. On the other hand, a negative point of SMD is its length extension property. Due to this property, the MAC function HMAC [4] invokes the underlying hash function twice. It causes inefficiency for short messages. The other negative point is that message blocks after padding may include a message block consisting only of a padding sequence, which needs an additional call to the compression function.

A domain extension scheme with minimum padding and free from the length extension property seems useful especially for processing short messages. Informally, we say that padding is minimum if the produced message blocks include no message block only with the padding sequence for any non-empty input message.

1.2. Our Contribution

This article first presents a sequential domain extension scheme with minimum padding for hashing using a compression function. The padding function of the domain extension is not injective. It extends the MDP domain extension [5] and uses two distinct permutations for domain separation. The permutations also prevent the length extension property. The permutations need not be cryptographic transformations. A typical candidate for them is bitwise XOR with a nonzero constant.

Then, the security properties of a hash function using the proposed domain extension are analyzed. The properties considered are the collision resistance and the indifferentiability.

The proposed domain extension does not preserve the collision resistance. However, it is shown that the collision resistance of a hash function using the domain extension is reduced to the collision resistance and the everywhere preimage resistance of the underlying compression function.

It is also shown that a hash function using the domain extension is indifferentiable from a variable-input-length random oracle (VIL RO) up to the birthday bound if the underlying compression function is a fixed-input-length random oracle (FIL RO) or the Davies-Meyer mode of a block cipher chosen uniformly at random.

The proposed domain extension scheme can also be applied to the sponge construction in a straightforward way. It is shown that the resultant hash function is indifferentiable from a VIL RO up to the birthday bound if the underlying permutation is chosen uniformly at random.

1.3. Related Work

The presented domain extension of hashing was first considered for a pseudorandom function using a compression function [6]. It is shown in [6] that keying via IV to the domain extension presented in the current article produces a pseudorandom function if the underlying compression function is a pseudorandom function against related-key attacks with respect to the permutations used in the domain extension.

There are many proposals for domain extension of hashing. On the other hand, little attention has been paid to padding.

The most related work was done by Bagheri et al. [7]. They proposed a generic scheme to construct an iterated hash function which requires neither a fixed IV nor the MD strengthening. Their scheme uses three distinct compression functions to get prefix-free and suffix-free property. It assumes injective padding function. They also showed that their hash function is indifferentiable from a VIL RO if the underlying compression functions are FIL ROs.

Nandi [8] showed that the suffix-free property of padding is necessary and sufficient for the plain MD domain extension to preserve the collision resistance. He also presented a suffix-free padding scheme which works for any input message M of arbitrary length. It appends $O(\log |M|)$ bits to M. The padding scheme for SHA-2, which is based on Merkle's [2], also appends only $O(\log |M|)$ bits. However, it works only for input messages of bounded length.

Coron et al. [9] formalized the indifferentiability notion for hash functions in the framework by Maurer et al. [10]. They also showed the indifferentiability of the following domain extension schemes: prefix-free plain MD, plain MD with output truncation (chopMD), NMAC construction, and HMAC construction, where HMAC construction is rather different from the MAC function HMAC [4]. They assumed injective padding. Their work was followed by Chang et al. [11,12].

Bellare and Ristenpart introduced the notion of multi-property preservation for domain extension [13]. They also presented the EMD (enveloped MD) domain extension and showed that it preserves collision resistance, pseudorandom function, and indifferentiability assuming injective padding.

Merkle-Damgård with permutation (MDP) [5] is a variant of plain MD preventing its length-extension property. A typical example of MDP was presented by Kelsey in [14]. It uses bitwise XOR with a nonzero constant for the permutation.

Minimum padding is already common among MAC functions based on a block cipher such as CMAC [15] and PMAC [16]. The idea to finalize the iteration with multiple non-cryptographic transformations for domain separation is used in the secure CBC-MAC variants GCBC1 and GCBC2 [17].

Sarkar [18] presented a domain extension scheme preserving the collision resistance based on directed acyclic graphs. Bertoni et al. [19] formulated sufficient conditions for domain extension schemes covering both tree and sequential structures to be indifferentiable up to the birthday bound. Based on the sufficient conditions, a coding scheme for tree domain extension schemes is specified in [20], which also covers sequential domain extension schemes.

The sponge construction [21] is a scheme to construct a hash function using a function with its input length equal to its output length, which is typically a permutation. It was invented for the SHA-3 hash function [22]. It is adopted by lightweight hash functions such as PHOTON [23] and SPONGENT [24]. It is also extended to design cryptographic schemes such as authenticated encryption [25].

1.4. Organization

Section 2 gives notations used in this article and defines some security properties required of cryptographic hash functions. The proposed scheme is described in Section 3. The collision resistance of the proposed hash function is discussed in the standard model in Section 4. The indifferentiability is discussed in Section 5. The proposed domain extension is applied to the sponge construction in Section 6. A concluding remark is given in Section 7.

2. Preliminaries

2.1. Notations

Let $\Sigma = \{0,1\}$. Let $\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i$, and $(\Sigma^n)^+ = \bigcup_{i=1}^{\infty} \Sigma^{ni}$.

For binary sequences x and y, let $x\|y$ be their concatenation. The empty sequence is denoted by ε.

The operation of selecting an element from set S uniformly at random and assigning it to s is denoted by $s \twoheadleftarrow S$.

2.2. Collision Resistance and Preimage Resistance

In this section, the collision resistance and everywhere preimage resistance [26] are defined in the standard model. To do so, a family of hash functions should be introduced. Suppose that h is a hash function chosen at random from some set of hash functions from $\mathcal{X}$ to $\mathcal{Y}$ in some efficient way.

Let A be an adversary which is given h as input and tries to find a collision pair for h. A collision pair for h are a pair of distinct inputs mapped to the same output by h. The col-advantage of A against h is given by

$$\mathrm{Adv}_h^{\mathrm{col}}(A) = \Pr[(M, M') \leftarrow A(h) : h(M) = h(M') \wedge M \neq M'],$$

where the probability is taken over the coin tosses by A and the distribution of h.

Let A be an adversary which is given h as input and tries to find a preimage of an output for h. The pre-advantage of A against h is given by

$$\mathrm{Adv}_h^{\mathrm{epre}}(A) = \max_{Y \in \mathcal{Y}} \{\Pr[M \leftarrow A(h) : h(M) = Y]\},$$

where the probability is taken over the coin tosses by A and the distribution of h.

2.3. Indifferentiability from Random Oracle

Maurer et al. [10] formalized the notion of indifferentiability as a generalized notion of indistinguishability. Then, Coron et al. [9] tailored it for the security analysis of hash functions.

Let C be an algorithm with oracle access to an ideal primitive $\mathcal{P}$. Here in this article, C is a domain extension scheme using $\mathcal{P}$ with fixed input length and $C^{\mathcal{P}}$ defines a hash function. Let $\mathcal{R}$ be a VIL

random oracle and S be a simulator which has oracle access to $\mathcal{R}$. $S^{\mathcal{R}}$ simulates $\mathcal{P}$ in order to convince an adversary that $\mathcal{R}$ is $C^{\mathcal{P}}$. The indiff-advantage of adversary A against (C, S) is given by

$$\mathrm{Adv}^{\mathrm{indiff}}_{C,S}(A) = \left|\Pr[A^{C^{\mathcal{P}},\mathcal{P}} = 1] - \Pr[A^{\mathcal{R},S^{\mathcal{R}}} = 1]\right|,$$

where the probabilities are taken over the coin tosses by A, S and the oracles $\mathcal{R}$ and $\mathcal{P}$. $C^{\mathcal{P}}$ and $\mathcal{R}$ are called VIL oracles, and $\mathcal{P}$ and $S^{\mathcal{R}}$ are called FIL oracles.

3. Proposed Scheme

The proposed hash function consists of a compression function $F : \Sigma^n \times \Sigma^w \to \Sigma^n$, permutations π_0 and π_1 over Σ^n, and an initialization vector $IV \in \Sigma^n$. For π_0 and π_1, it is assumed that $\pi_0(v) \neq v$, $\pi_1(v) \neq v$ and $\pi_0(v) \neq \pi_1(v)$ for any $v \in \Sigma^n$.

Remark 1. *Let c_0 and c_1 be distinct constants in $\Sigma^n \setminus \{\mathbf{0}\}$. Let $\pi_i(v) = v \oplus c_i$ for $i = 0, 1$. Then, for any $v \in \Sigma^n$, $\pi_0(v) \neq v$, $\pi_1(v) \neq v$ and $\pi_0(v) \neq \pi_1(v)$.*

Let π be a permutation over Σ^n. For $1 \leq i \leq x$, let $X_i \in \Sigma^w$. The MDP domain extension [5] $\mathsf{C}^{F,\pi}_{IV} : (\Sigma^w)^+ \to \Sigma^n$ for F is defined as follows: $\mathsf{C}^{F,\pi}_{IV}(X_1 \| X_2 \| \cdots \| X_x) = v_x$, where $v_0 \leftarrow IV$, $v_i \leftarrow F(v_{i-1}, X_i)$ for $1 \leq i \leq x-1$, and $v_x \leftarrow F(\pi(v_{x-1}), X_x)$.

For $M \in \Sigma^*$, the padding function is defined as follows:

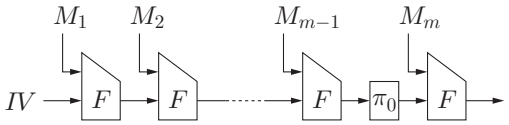
$$\mathtt{pad}(M) = \begin{cases} M & \text{if } |M| > 0 \text{ and } |M| \equiv 0 \pmod{w}, \\ M \| 10^d & \text{otherwise,} \end{cases}$$

where d is the smallest non-negative integer such that $|M| + 1 + d \equiv 0 \pmod{w}$. The length of any output of pad is a positive multiple of w. In particular, $\mathtt{pad}(\varepsilon) = 10^{w-1}$. If $|M| > 0$, then $|\mathtt{pad}(M)| = w\lceil |M|/w \rceil$.

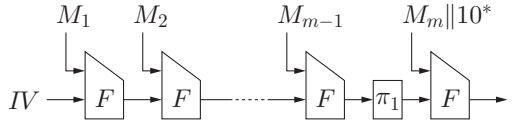
The proposed hash function $H^{F,\{\pi_0,\pi_1\}}_{IV} : \Sigma^* \to \Sigma^n$ is defined as follows:

$$H^{F,\{\pi_0,\pi_1\}}_{IV}(M) = \begin{cases} \mathsf{C}^{F,\pi_0}_{IV}(\mathtt{pad}(M)) & \text{if } |M| > 0 \text{ and } |M| \equiv 0 \pmod{w}, \\ \mathsf{C}^{F,\pi_1}_{IV}(\mathtt{pad}(M)) & \text{otherwise.} \end{cases}$$

It is also depicted in Figure 1.



(**a**) For M such that $|M| > 0$ and $|M| \equiv 0 \pmod{w}$. $|M_m| = w$.



(**b**) For M such that $|M| = 0$ or $|M| \not\equiv 0 \pmod{w}$. $|M_m| = 0$ if $|M| = 0$ and $1 \leq |M_m| \leq w - 1$ otherwise.

Figure 1. The proposed hash function. $M = M_1 \| M_2 \| \cdots \| M_m$, where $|M_i| = w$ for $1 \leq i \leq m - 1$.

4. Collision Resistance

The collision resistance of $H_{IV}^{F,\{\pi_0,\pi_1\}}$ is discussed in the standard model. It is assumed that the compression function F is chosen at random from some set of functions from $\Sigma^n \times \Sigma^w$ to Σ^n in some efficient way.

The collision resistance of $H_{IV}^{F,\{\pi_0,\pi_1\}}$ needs a new security requirement for F, which is a kind of collision resistance. A pair of distinct inputs (v, X) and (v', X') for F are called a $\{\pi_0, \pi_1\}$-pseudo-collision pair if $\pi_0(F(v, X)) = \pi_1(F(v', X'))$. The advantage of adversary A against F with respect to $\{\pi_0, \pi_1\}$-pseudo-collision is defined similarly to the col-advantage. It is denoted by $\mathrm{Adv}^{\mathrm{pcol}}_{F,\{\pi_0,\pi_1\}}(A)$.

It will be shown that the collision resistance of $H_{IV}^{F,\{\pi_0,\pi_1\}}$ is reduced to the collision resistance, the $\{\pi_0, \pi_1\}$-pseudo-collision resistance and the everywhere preimage resistance of F.

Lemma 1. *Any collision pair for $H_{IV}^{F,\{\pi_0,\pi_1\}}$ implies a collision pair, a $\{\pi_0, \pi_1\}$-pseudo-collision pair, or a preimage of IV, $\pi_0^{-1}(\pi_1(IV))$, or $\pi_1^{-1}(\pi_0(IV))$ for F.*

Proof. Let M and M' be any collision pair for $H_{IV}^{F,\{\pi_0,\pi_1\}}$. It is shown below that, by tracing back the computation of $H_{IV}^{F,\{\pi_0,\pi_1\}}(M)$ and $H_{IV}^{F,\{\pi_0,\pi_1\}}(M')$, one can find a collision pair for F, a $\{\pi_0, \pi_1\}$-pseudo-collision pair for F, or a preimage of IV, $\pi_0^{-1}(\pi_1(IV))$, or $\pi_1^{-1}(\pi_0(IV))$ for F. Let $|\mathtt{pad}(M)|/w = m$ and $|\mathtt{pad}(M')|/w = m'$.

Suppose that $\mathtt{pad}(M) = \mathtt{pad}(M')$. Then, one of $H_{IV}^{F,\{\pi_0,\pi_1\}}(M)$ and $H_{IV}^{F,\{\pi_0,\pi_1\}}(M')$ uses π_0 and the other uses π_1. Notice that $\pi_0(v) \neq \pi_1(v)$ for any $v \in \Sigma^n$. If $m = m' = 1$, then one finds a collision pair for F since $\pi_0(IV) \neq \pi_1(IV)$. If $m = m' \geq 2$, then one finds a collision pair or a $\{\pi_0, \pi_1\}$-pseudo-collision pair for F since $\pi_0(v) = \pi_1(v')$ implies $v \neq v'$ for any $v, v' \in \Sigma^n$.

Suppose that $\mathtt{pad}(M) \neq \mathtt{pad}(M')$.

(i) Suppose that one of $H_{IV}^{F,\{\pi_0,\pi_1\}}(M)$ and $H_{IV}^{F,\{\pi_0,\pi_1\}}(M')$ uses π_0 and the other uses π_1. Assume that $H_{IV}^{F,\{\pi_0,\pi_1\}}(M)$ uses π_0 and $H_{IV}^{F,\{\pi_0,\pi_1\}}(M')$ uses π_1 without loss of generality. If $m = m' = 1$, then one finds a collision pair for F. If $m = 1$ and $m' \geq 2$, then one finds a collision pair for F or a preimage of $\pi_1^{-1}(\pi_0(IV))$ for F. If $m \geq 2$ and $m' = 1$, then one finds a collision pair for F or a preimage of $\pi_0^{-1}(\pi_1(IV))$ for F. If $m \geq 2$ and $m' \geq 2$, then one finds a collision pair or a $\{\pi_0, \pi_1\}$-pseudo-collision pair for F.

(ii) Suppose that both of $H_{IV}^{F,\{\pi_0,\pi_1\}}(M)$ and $H_{IV}^{F,\{\pi_0,\pi_1\}}(M')$ uses a same permutation. If $m = m' = 1$, then one finds a collision pair for F. If $m = 1$ and $m' \geq 2$, or $m \geq 2$ and $m' = 1$, then one finds a collision pair for F or a preimage of IV for F. If $m \geq 2$ and $m' \geq 2$, then one finds a collision pair or a preimage of IV for F.

□

Theorem 1. *For any adversary A trying to find a collision pair for $H_{IV}^{F,\{\pi_0,\pi_1\}}$ with run time t, there exist adversaries B_1, B_2 and B_3 such that*

$$\mathrm{Adv}^{\mathrm{col}}_{H_{IV}^{F,\{\pi_0,\pi_1\}}}(A) \leq \mathrm{Adv}^{\mathrm{col}}_{F}(B_1) + \mathrm{Adv}^{\mathrm{pcol}}_{F,\{\pi_0,\pi_1\}}(B_2) + 3\,\mathrm{Adv}^{\mathrm{epre}}_{F}(B_3).$$

The run times of B_1, B_2 and B_3 are about $t + O((|\mathtt{pad}(M)| + |\mathtt{pad}(M')|)T_F/w)$, where M and M' are a collision pair of $H_{IV}^{F,\{\pi_0,\pi_1\}}$ output by A and T_F is the time required to compute F.

Proof. Let B be an algorithm which works as follows. B takes F as input. It first runs A with input $H_{IV}^{F,\{\pi_0,\pi_1\}}$. If A fails to find a collision pair for $H_{IV}^{F,\{\pi_0,\pi_1\}}$, then it aborts. Otherwise, for a collision pair M and M' output by A, it computes $H_{IV}^{F,\{\pi_0,\pi_1\}}(M)$ and $H_{IV}^{F,\{\pi_0,\pi_1\}}(M')$.

Let B_1 be an adversary trying to find a collision pair for F. Let B_2 be an adversary trying to find a $\{\pi_0, \pi_1\}$-pseudo-collision pair for F. Let B_3 be an adversary trying to find a preimage of IV, $\pi_0^{-1}(\pi_1(IV))$, or $\pi_1^{-1}(\pi_0(IV))$ for F. All of them first run B. From Lemma 1, if A succeeds in finding a collision pair for $H_{IV}^{F,\{\pi_0,\pi_1\}}$, then B_1, B_2 or B_3 succeed. □

5. Indifferentiability from Random Oracle

5.1. In the Random Oracle Model

In this section, to discuss the indifferentiability, the compression function F is assumed to be chosen uniformly at random from all the functions from $\Sigma^n \times \Sigma^w$ to Σ^n.

The following theorem implies that the proposed hash function is indifferentiable from a random oracle up to the birthday bound. The game-playing technique [27] is used for the proof.

Theorem 2. *Suppose that the compression function $F : \Sigma^n \times \Sigma^w \to \Sigma^n$ is chosen uniformly at random. Then, for the hash function $H_{IV}^{F,\{\pi_0,\pi_1\}}$, there exists a simulator S of F such that, for any adversary A making at most q queries to its FIL oracle and queries to its VIL oracle which cost at most σ message blocks in total,*

$$\mathrm{Adv}^{\mathrm{indiff}}_{H_{IV}^{F,\{\pi_0,\pi_1\}},S}(A) \leq \frac{5(\sigma+q)^2}{2^n} + \frac{3\sigma q}{2^n - 6q + 1},$$

and S makes at most q queries.

Proof. Each game provides two interfaces to adversary A: $\mathcal{H}$ for the hash function and $\mathcal{F}$ for the compression function. It is assumed without loss of generality that A makes no repeated queries both to $\mathcal{H}$ and to $\mathcal{F}$.

The game G1 is given in Figure 2. $\mathcal{F}$ simply calls F, which implements the compression function F by lazy evaluation. F uses a partial function $\mathtt{F}$. Initially, $\mathtt{F}[v, X] = \perp$ for every $(v, X) \in \Sigma^n \times \Sigma^w$. $\mathcal{H}$ computes $H_{IV}^{F,\{\pi_0,\pi_1\}}$ with the aid of F. Thus,

$$\Pr\left[A^{H_{IV}^{F,\{\pi_0,\pi_1\}},F} = 1\right] = \Pr\left[A^{\mathrm{G1}} = 1\right].$$

Notice that F may receive repeated queries since $\mathcal{H}$ also calls F as well as $\mathcal{F}$.

The game G2 is given in Figure 3a. $\mathcal{F}$ and $\mathcal{H}$ are not changed and omitted.

In G2, F constructs and maintains a directed graph $(\boldsymbol{V}, \boldsymbol{E})$ based on the queries to F. It also uses a function findM, which will be described later. Initially, $\boldsymbol{V} = \{\}$ and $\boldsymbol{E} = \{\}$. For a new query (v, X), if $\mathsf{findM}(v, X) \neq \perp$, then F replaces $\boldsymbol{V}$ with $\boldsymbol{V} \cup \{v\}$. On the other hand, if $\mathsf{findM}(v, X) = \perp$, then F replaces $\boldsymbol{V}$ with $\boldsymbol{V} \cup \{v, \mathtt{F}[v, X]\}$ and $\boldsymbol{E}$ with $\boldsymbol{E} \cup \{(v, \mathtt{F}[v, X])\}$. The edge $(v, \mathtt{F}[v, X])$ is labeled with X. $\boldsymbol{T}$ and $\boldsymbol{H}$ are the sets of tails and heads of edges in $(\boldsymbol{V}, \boldsymbol{E})$, respectively. Vertices with no adjacent edges in $(\boldsymbol{V}, \boldsymbol{E})$ are also included in $\boldsymbol{T}$. Initially, $\boldsymbol{T} = \{\}$ and $\boldsymbol{H} = \{\}$.

Interface $\mathcal{H}(M)$**:**

100: $X_1 \| X_2 \| \cdots \| X_x \leftarrow \mathtt{pad}(M)$
101: $v_0 \leftarrow IV$
102: **for** $1 \le i \le x-1$ **do**
103: $\quad v_i \leftarrow \mathsf{F}(v_{i-1}, X_i)$
104: **end for**
105: **if** $|M| > 0$ and $|M| \equiv 0 \pmod{w}$ **then**
106: $\quad v_x \leftarrow \mathsf{F}(\pi_0(v_{x-1}), X_x)$
107: **else**
108: $\quad v_x \leftarrow \mathsf{F}(\pi_1(v_{x-1}), X_x)$
109: **end if**
110: **return** v_x

Interface $\mathcal{F}(v, X)$**:**

200: **return** $\mathsf{F}(v, X)$

Function $\mathsf{F}(v, X)$**:**

600: **if** $\mathtt{F}[v, X] = \perp$ **then**
601: $\quad \mathtt{F}[v, X] \twoheadleftarrow \Sigma^n$
602: **end if**
603: **return** $\mathtt{F}[v, X]$

Figure 2. Game G1. For the partial function $\mathtt{F}$ used in F, initially, $\mathtt{F}[v, X] = \perp$ for every $(v, X) \in \Sigma^n \times \Sigma^w$.

Function $\mathsf{F}(v, X)$**:**

600: **if** $\mathtt{F}[v, X] = \perp$ **then**
601: $\quad M \leftarrow \mathsf{findM}(v, X)$
602: $\quad$ **if** $M \neq \perp$ **then**
603: $\quad\quad \mathtt{F}[v, X] \twoheadleftarrow \Sigma^n$
604: $\quad$ **else**
605: $\quad\quad \mathtt{F}[v, X] \twoheadleftarrow \Sigma^n$
606: $\quad\quad$ **if** $\mathtt{F}[v, X] \in \boldsymbol{B}$ **then**
607: $\quad\quad\quad \mathit{bad} \leftarrow \mathtt{true}$
608: $\quad\quad\quad \mathtt{F}[v, X] \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}$
609: $\quad\quad$ **end if**
610: $\quad\quad \boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{\mathtt{F}[v, X]\}$
611: $\quad$ **end if**
612: $\quad \boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
613: **end if**
614: **return** $\mathtt{F}[v, X]$

(**a**) F of G2

Function $\mathsf{F}(v, X)$**:**

600: **if** $\mathtt{F}[v, X] = \perp$ **then**
601: $\quad M \leftarrow \mathsf{findM}(v, X)$
602: $\quad$ **if** $M \neq \perp$ **then**
603: $\quad\quad \mathtt{F}[v, X] \twoheadleftarrow \Sigma^n$
604: $\quad$ **else**
605: $\quad\quad \mathtt{F}[v, X] \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}$
606: $\quad\quad \boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{\mathtt{F}[v, X]\}$
607: $\quad$ **end if**
608: $\quad \boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
609: **end if**
610: **return** $\mathtt{F}[v, X]$

(**b**) F of G3

Figure 3. Games G2 and G3. $\mathcal{F}$ and $\mathcal{H}$ are omitted, which are identical to those of G1. $\boldsymbol{B} = \boldsymbol{T} \cup \pi_0^{-1}(\boldsymbol{T}) \cup \pi_1^{-1}(\boldsymbol{T}) \cup \boldsymbol{H} \cup \pi_0^{-1}(\pi_1(\boldsymbol{H})) \cup \pi_1^{-1}(\pi_0(\boldsymbol{H})) \cup \{IV, \pi_0^{-1}(IV), \pi_1^{-1}(IV), \pi_0^{-1}(\pi_1(IV)), \pi_1^{-1}(\pi_0(IV))\}$. Initially, $\boldsymbol{T} = \{\}$ and $\boldsymbol{H} = \{\}$.

findM tries to find a path in $(\boldsymbol{V}, \boldsymbol{E})$ corresponding to the computation $H_{IV}^{F,\{\pi_0,\pi_1\}}(M)$ for some M. Given (v, X) as input, findM first searches a path from IV to $\pi_0^{-1}(v)$ or $\pi_1^{-1}(v)$ in $(\boldsymbol{V}, \boldsymbol{E})$. If IV equals $\pi_0^{-1}(v)$ or $\pi_1^{-1}(v)$, then the single vertex IV is regarded as a path. If findM finds a path, then let $X_1, X_2, \ldots, X_l$ be the labels of the edges on the path. If the path is IV, then $l = 0$, that is, $X_1 \| X_2 \| \cdots \| X_l = \varepsilon$. If there exists some $M \in \Sigma^*$ such that $\mathtt{pad}(M) = X_1 \| X_2 \| \cdots \| X_l \| X$, which depends on whether the terminal of the path is $\pi_0^{-1}(v)$ or $\pi_1^{-1}(v)$, then findM returns M. Otherwise, findM returns $\perp$. It will be shown that $\mathsf{findM}(v, X)$ finds at most one path.

F of G2 differs from F of G1 only if *bad* gets $\mathtt{true}$ in G2. This is because $\mathtt{F}[v, X]$ is chosen uniformly at random in G2 until *bad* gets $\mathtt{true}$. For the i-th call to F, $|\boldsymbol{B}| \le 6i - 1$ since

$$\boldsymbol{B} = \boldsymbol{T} \cup \pi_0^{-1}(\boldsymbol{T}) \cup \pi_1^{-1}(\boldsymbol{T}) \cup \boldsymbol{H} \cup \pi_0^{-1}(\pi_1(\boldsymbol{H})) \cup \pi_1^{-1}(\pi_0(\boldsymbol{H})) \cup \left\{IV, \pi_0^{-1}(IV), \pi_1^{-1}(IV), \pi_0^{-1}(\pi_1(IV)), \pi_1^{-1}(\pi_0(IV))\right\} ,$$

$|\boldsymbol{T}| \le i - 1$ and $|\boldsymbol{H}| \le i - 1$. F is called at most $(\sigma + q)$ times. Thus,

$$\left|\Pr\left[A^{\mathrm{G1}} = 1\right] - \Pr\left[A^{\mathrm{G2}} = 1\right]\right| \le \Pr\left[A^{\mathrm{G2}} \text{ sets } \mathit{bad}\right] \le \sum_{i=1}^{\sigma+q} \frac{6i-1}{2^n} = \frac{3(\sigma+q)^2 + 2(\sigma+q)}{2^n} .$$

For the game G3 in Figure 3b, the lines from 605 to 609 in G2 are replaced with the line 605 in G3. Since they are equivalent, $\Pr\left[A^{\mathrm{G2}}=1\right]=\Pr\left[A^{\mathrm{G3}}=1\right]$.

The game G4 is given in Figure 4. It introduces a variable-input-length random oracle H, which is implemented by lazy evaluation. Initially, $\mathtt{H}[M]=\perp$ for every $M\in\Sigma^*$. H may receive repeated queries since it is called by both $\mathcal{H}$ and $\mathcal{F}$. Different from F of G3, F assigns $\mathsf{H}(M)$ to $\mathtt{F}[v,X]$ at the line 603 in G4. Different from $\mathcal{H}$ of G3, $\mathcal{H}(M)$ returns $\mathsf{H}(M)$ in G4. We will see that G4 is actually equivalent to G3 in spite of these changes.

Interface $\mathcal{H}(M)$**:**

100: $X_1\|X_2\|\cdots\|X_x\leftarrow\mathtt{pad}(M)$
101: $v_0\leftarrow IV$
102: **for** $1\le i\le x-1$ **do**
103: $\quad v_i\leftarrow\mathsf{F}(v_{i-1},X_i)$
104: **end for**
105: **if** $|M|>0$ and $|M|\equiv 0 \pmod w$ **then**
106: $\quad v_x\leftarrow\mathsf{F}(\pi_0(v_{x-1}),X_x)$
107: **else**
108: $\quad v_x\leftarrow\mathsf{F}(\pi_1(v_{x-1}),X_x)$
109: **end if**
110: **return** $\mathsf{H}(M)$

Function $\mathsf{H}(M)$**:**

500: **if** $\mathtt{H}[M]=\perp$ **then**
501: $\quad \mathtt{H}[M]\twoheadleftarrow\Sigma^n$
502: **end if**
503: **return** $\mathtt{H}[M]$

Interface $\mathcal{F}(v,X)$**:**

200: **return** $\mathsf{F}(v,X)$

Function $\mathsf{F}(v,X)$**:**

600: **if** $\mathtt{F}[v,X]=\perp$ **then**
601: $\quad M\leftarrow\mathsf{findM}(v,X)$
602: $\quad$ **if** $M\neq\perp$ **then**
603: $\quad\quad \mathtt{F}[v,X]\leftarrow\mathsf{H}(M)$
604: $\quad$ **else**
605: $\quad\quad \mathtt{F}[v,X]\twoheadleftarrow\Sigma^n\setminus\boldsymbol{B}$
606: $\quad\quad \boldsymbol{H}\leftarrow\boldsymbol{H}\cup\{\mathtt{F}[v,X]\}$
607: $\quad$ **end if**
608: $\quad \boldsymbol{T}\leftarrow\boldsymbol{T}\cup\{v\}$
609: **end if**
610: **return** $\mathtt{F}[v,X]$

Figure 4. Game G4. Initially, $\mathtt{H}[M]=\perp$ for every $M\in\Sigma^*$.

First, let us see some properties of the graph $(\boldsymbol{V},\boldsymbol{E})$. Both in G3 and in G4, at the beginning of each run of F with (v,X) such that $\mathtt{F}[v,X]=\perp$, $\boldsymbol{V}\subseteq\boldsymbol{T}\cup\boldsymbol{H}$. Then, whenever this run adds $\mathtt{F}[v,X]$ to both $\boldsymbol{V}$ and $\boldsymbol{H}$, $\mathtt{F}[v,X]$ is chosen from $\Sigma^n\setminus\boldsymbol{B}$, where $\{IV\}\cup\boldsymbol{T}\cup\boldsymbol{H}\subseteq\boldsymbol{B}$. Thus, every vertex in $(\boldsymbol{V},\boldsymbol{E})$ has at most one incoming edge, and IV has no incoming edge. It implies that every vertex in $(\boldsymbol{V},\boldsymbol{E})$ has at most one simple path from IV. In addition, for every path $(v_1,v_2,\ldots,v_l)$ with $v_1=IV$, v_i's are added to $(\boldsymbol{V},\boldsymbol{E})$ in this order. Furthermore, before v_l is added to $(\boldsymbol{V},\boldsymbol{E})$, neither $(\pi_0(v_l),X')$ nor $(\pi_1(v_l),X')$ were asked to F for any $X'\in\Sigma^w$ since $\{\pi_0^{-1}(IV),\pi_1^{-1}(IV)\}\cup\pi_0^{-1}(\boldsymbol{T})\cup\pi_1^{-1}(\boldsymbol{T})\subseteq\boldsymbol{B}$.

Suppose that $\mathsf{findM}(v,X)$ finds two paths in $(\boldsymbol{V},\boldsymbol{E})$. Then, one is from IV to $\pi_0^{-1}(v)$ and the other is from IV to $\pi_1^{-1}(v)$. Notice that $\pi_0^{-1}(v)\neq\pi_1^{-1}(v)$ since $\pi_0(u)\neq\pi_1(u)$ for every $u\in\Sigma^n$. Suppose that both paths have two or more vertices. Then, both $\pi_0^{-1}(v)$ and $\pi_1^{-1}(v)$ are elements of $\boldsymbol{H}$, which implies that one was added to $\boldsymbol{H}$ after the other since at most one vertex is added to $\boldsymbol{H}$ during each run of F. It contradicts $\pi_{1\oplus b}^{-1}(\pi_b(\boldsymbol{H}))\subseteq\boldsymbol{B}$ for $b\in\Sigma$. Suppose that one path is the single vertex IV and the other has two or more vertices. $\pi_b^{-1}(v)=IV$ contradicts $\pi_{1\oplus b}^{-1}(\pi_b(IV))\subseteq\boldsymbol{B}$ for $b\in\Sigma$. Thus, $\mathsf{findM}(v,X)$ finds at most a single path in $(\boldsymbol{V},\boldsymbol{E})$.

In G4, for a new query (v,X) to F, suppose that findM finds a path in $(\boldsymbol{V},\boldsymbol{E})$ and returns M corresponding to the path and (v,X). Then, M is a new query to H, that is $\mathtt{H}[M]=\perp$, and it is assigned an element chosen uniformly at random from Σ^n. On the other hand, for $\mathcal{H}$, $v_x=\mathsf{H}(M)$. Thus, G4 is equivalent to G3, and $\Pr\left[A^{\mathrm{G4}}=1\right]=\Pr\left[A^{\mathrm{G3}}=1\right]$.

From G4 to G5, only F changes, which is given in Figure 5a. F of G5 is augmented with the lines from 600 to 606 and the lines from 614 to 616. $\boldsymbol{H}_A$ is the set of heads of edges in $(\boldsymbol{V},\boldsymbol{E})$ in the view of A. Initially, $\boldsymbol{H}_A=\{\}$. These changes do not affect the output of F. Thus, G5 is equivalent to G4, and $\Pr\left[A^{\mathrm{G5}}=1\right]=\Pr\left[A^{\mathrm{G4}}=1\right]$.

Function $\mathsf{F}(v, X)$**:**

600: **if** $\mathtt{F}[v, X] \neq \perp$ **then**
601: **if** (v, X) is from $\mathcal{F}$ **then**
602: **if** $v \in \boldsymbol{B}_{\mathrm{a}}$ **then**
603: $bad \leftarrow \mathtt{true}$
604: **end if**
605: **end if**
606: **end if**
607: **if** $\mathtt{F}[v, X] = \perp$ **then**
608: $M \leftarrow \mathsf{findM}(v, X)$
609: **if** $M \neq \perp$ **then**
610: $\mathtt{F}[v, X] \leftarrow \mathsf{H}(M)$
611: **else**
612: $\mathtt{F}[v, X] \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}$
613: $\boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{\mathtt{F}[v, X]\}$
614: **if** (v, X) is from $\mathcal{F}$ **then**
615: $\boldsymbol{H}_A \leftarrow \boldsymbol{H}_A \cup \{\mathtt{F}[v, X]\}$
616: **end if**
617: **end if**
618: $\boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
619: **end if**
620: **return** $\mathtt{F}[v, X]$

(**a**) F of G5 and G6. $\boldsymbol{B}_{\mathrm{a}} = (\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_0(\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_1(\boldsymbol{H} \setminus \boldsymbol{H}_A)$. Initially, $\boldsymbol{H}_A = \{\}$.

Interface $\mathcal{H}(M)$**:**

100: **return** $\mathsf{H}(M)$

Function $\mathsf{H}(M)$**:**

500: **if** $\mathtt{H}[M] = \perp$ **then**
501: $\mathtt{H}[M] \twoheadleftarrow \Sigma^n$
502: **end if**
503: **return** $\mathtt{H}[M]$

Interface $\mathcal{F}(v, X)$**:**

200: **return** $\mathsf{F}(v, X)$

Function $\mathsf{F}(v, X)$**:**

600: $M \leftarrow \mathsf{findM}(v, X)$
601: **if** $M \neq \perp$ **then**
602: $\mathtt{F}[v, X] \leftarrow \mathsf{H}(M)$
603: **else**
604: $\mathtt{F}[v, X] \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}$
605: $\boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{\mathtt{F}[v, X]\}$
606: **end if**
607: $\boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
608: **return** $\mathtt{F}[v, X]$

(**b**) Game G7

Figure 5. Games G5, G6 and G7.

From G5 to G6, only $\mathcal{H}$ changes. $\mathcal{H}$ of G6 is identical to that of G7, which is given in Figure 5b. In G6, $\mathcal{H}(M)$ does not call F and just returns $\mathsf{H}(M)$. In G6, F is called only by $\mathcal{F}$ and it does not receive any repeated queries, which implies that *bad* never gets `true`. On the other hand, *bad* may get `true` in G5. If *bad* gets `true` in G5, then A may trace some computation path of $H_{IV}^{F,\{\pi_0,\pi_1\}}$ in $(\boldsymbol{V}, \boldsymbol{E})$ from its middle. $|\boldsymbol{B}_{\mathrm{a}}| \leq 3\sigma$ since $\boldsymbol{B}_{\mathrm{a}} = (\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_0(\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_1(\boldsymbol{H} \setminus \boldsymbol{H}_A)$ and $|\boldsymbol{H} \setminus \boldsymbol{H}_A| \leq \sigma$. A knows at most $6q - 1$ elements in $\boldsymbol{B}$. Thus,

$$\left|\Pr\left[A^{\mathrm{G5}} = 1\right] - \Pr\left[A^{\mathrm{G6}} = 1\right]\right| \leq \Pr\left[A^{\mathrm{G5}} \text{ sets } bad\right] \leq \frac{3\sigma q}{2^n - 6q + 1} \ .$$

From G6 to G7, only F changes. G7 is given in Figure 5b. F of G7 is obtained from F of G6 by removing the lines from 600 to 606 and the lines from 614 to 616. Since F does not receive any repeated queries, the lines 607 and 619 are also removed. These changes do not affect the output of F. Thus, $\Pr\left[A^{\mathrm{G7}} = 1\right] = \Pr\left[A^{\mathrm{G6}} = 1\right]$. F of G7 works as a simulator S of F.

From the discussion above, we have

$$\begin{aligned}\mathrm{Adv}^{\mathrm{indiff}}_{H_{IV}^{F,\{\pi_0,\pi_1\}},S}(A) &= \left|\Pr\left[A^{\mathrm{G1}} = 1\right] - \Pr\left[A^{\mathrm{G7}} = 1\right]\right| \leq \Pr\left[A^{\mathrm{G2}} \text{ sets } bad\right] + \Pr\left[A^{\mathrm{G5}} \text{ sets } bad\right] \\ &\leq \frac{3(\sigma+q)^2 + 2(\sigma+q)}{2^n} + \frac{3\sigma q}{2^n - 6q + 1} \leq \frac{5(\sigma+q)^2}{2^n} + \frac{3\sigma q}{2^n - 6q + 1} \ .\end{aligned}$$

□

5.2. In the Ideal Cipher Model

In this section, $F : \Sigma^n \times \Sigma^w \to \Sigma^n$ is assumed to be the Davies-Meyer compression function [28] using a block cipher $E : \Sigma^w \times \Sigma^n \to \Sigma^n$, where the key space of E is Σ^w. Namely, $F(V, X) = E(X, V) \oplus V$. E is assumed to be chosen uniformly at random.

Theorem 3. *Suppose that the compression function $F : \Sigma^n \times \Sigma^w \to \Sigma^n$ is the Davies-Meyer mode of a block cipher E chosen uniformly at random. Let D be the decryption function of E. Then, for the hash function $H_{IV}^{F,\{\pi_0,\pi_1\}}$, there exists a simulator S of (E, D) such that, for any adversary A making at most q_e queries to its FIL encryption oracle, q_d queries to its FIL decryption oracle, and queries to its VIL oracle which cost at most σ message blocks in total,*

$$\mathrm{Adv}^{\mathrm{indiff}}_{H_{IV}^{F,\{\pi_0,\pi_1\}},S}(A) \leq \frac{12(\sigma + q_e + q_d)^2}{2^n} + \frac{3\sigma(q_e + q_d)}{2^n - 6(q_e + q_d) - 5} \ ,$$

and S makes at most q_e queries.

Proof. Each game provides three interfaces to adversary A: $\mathcal{H}$ for the hash function, $\mathcal{E}$ for the encryption and $\mathcal{D}$ for the decryption. It is assumed without loss of generality that A makes no repeated queries both to $\mathcal{H}$ and to $(\mathcal{E}, \mathcal{D})$. For $\mathcal{E}$ and $\mathcal{D}$, once A gets a tuple (key, pt, ct) such that $E(key, pt) = ct$ by a query to $\mathcal{E}$ or $\mathcal{D}$, A never makes any query on the tuple.

The game G1 is given in Figure 6. $\mathcal{E}$ and $\mathcal{D}$ simply call E and D, respectively. E and D implement the encryption function and the decryption function by lazy evaluation, respectively. $\mathcal{H}$ computes $H_{IV}^{F,\{\pi_0,\pi_1\}}$ with the aid of E. Thus,

$$\Pr\left[A^{H_{IV}^{F,\{\pi_0,\pi_1\}},(E,D)} = 1\right] = \Pr\left[A^{\mathrm{G1}} = 1\right] \ .$$

Notice that E and D may receive repeated queries since $\mathcal{H}$ also calls E as well as $\mathcal{E}$.

Interface $\mathcal{H}(M)$:

100: $X_1 \| X_2 \| \cdots \| X_x \leftarrow \mathtt{pad}(M)$
101: $v_0 \leftarrow IV$
102: **for** $1 \leq i \leq x - 1$ **do**
103: $\quad v_i \leftarrow \mathsf{E}(X_i, v_{i-1}) \oplus v_{i-1}$
104: **end for**
105: **if** $|M| > 0$ and $|M| \equiv 0 \pmod{w}$ **then**
106: $\quad v_x \leftarrow \mathsf{E}(X_x, \pi_0(v_{x-1})) \oplus \pi_0(v_{x-1})$
107: **else**
108: $\quad v_x \leftarrow \mathsf{E}(X_x, \pi_1(v_{x-1})) \oplus \pi_1(v_{x-1})$
109: **end if**
110: **return** v_x

Interface $\mathcal{E}(X, v)$:

200: **return** $\mathsf{E}(X, v)$

Function $\mathsf{E}(X, v)$:

600: **if** $\mathtt{E}[X, v] = \perp$ **then**
601: $\quad \mathtt{E}[X, v] \twoheadleftarrow \Sigma^n \setminus \boldsymbol{C}_X$
602: **end if**
603: **return** $\mathtt{E}[X, v]$

Interface $\mathcal{D}(X, u)$:

300: **return** $\mathsf{D}(X, u)$

Function $\mathsf{D}(X, u)$:

700: **if** $\mathtt{D}[X, u] = \perp$ **then**
701: $\quad \mathtt{D}[X, u] \twoheadleftarrow \Sigma^n \setminus \boldsymbol{P}_X$
702: **end if**
703: **return** $\mathtt{D}[X, v]$

Figure 6. Game G1. For the partial functions $\mathtt{E}$ and $\mathtt{D}$, initially, $\mathtt{E}[X, v] = \perp$ for every $(X, v) \in \Sigma^w \times \Sigma^n$ and $\mathtt{D}[X, u] = \perp$ for every $(X, u) \in \Sigma^w \times \Sigma^n$. If u is assigned to $\mathtt{E}[X, v]$, then v is assigned to $\mathtt{D}[X, u]$. If v is assigned to $\mathtt{D}[X, u]$, then u is assigned to $\mathtt{E}[X, v]$. $\boldsymbol{P}_X$ and $\boldsymbol{C}_X$ are the sets of values already assigned as plaintexts and ciphertexts for key X, respectively.

From G1 to G2, only E and D are changed, which are given in Figure 7. In G2, $\mathtt{E}[X, v]$ and $\mathtt{D}[X, u]$ are chosen uniformly at random from Σ^n. G1 and G2 are identical until *bad* gets `true` in G2. Since E and D are called at most $\sigma + q_e + q_d$ times in total and $|\boldsymbol{P}_X| = |\boldsymbol{C}_X| \leq \sigma + q_e + q_d$,

$$\left|\Pr\left[A^{\mathrm{G1}} = 1\right] - \Pr\left[A^{\mathrm{G2}} = 1\right]\right| \leq \Pr\left[A^{\mathrm{G2}} \text{ sets } bad\right] \leq \frac{(\sigma + q_e + q_d)^2}{2^n} \ .$$

Function $\mathsf{E}(X, v)$**:**

600: **if** $\mathtt{E}[X, v] = \perp$ **then**
601: $\quad \mathtt{E}[X, v] \twoheadleftarrow \Sigma^n$
602: $\quad$ **if** $\mathtt{E}[X, v] \in \boldsymbol{C}_X$ **then**
603: $\quad\quad bad \leftarrow \mathtt{true}$
604: $\quad$ **end if**
605: **end if**
606: **return** $\mathtt{E}[X, v]$

Function $\mathsf{D}(X, u)$**:**

700: **if** $\mathtt{D}[X, u] = \perp$ **then**
701: $\quad \mathtt{D}[X, u] \twoheadleftarrow \Sigma^n$
702: $\quad$ **if** $\mathtt{D}[X, u] \in \boldsymbol{P}_X$ **then**
703: $\quad\quad bad \leftarrow \mathtt{true}$
704: $\quad$ **end if**
705: **end if**
706: **return** $\mathtt{D}[X, u]$

Figure 7. Game G2. $\mathcal{H}$, $\mathcal{E}$ and $\mathcal{D}$, which are not changed, are omitted.

From G2 to G3, only E and D are changed, which are given in Figure 8. In G3, E and D constructs and maintains a directed graph $(\boldsymbol{V}, \boldsymbol{E})$ based on the queries to them. Initially, $\boldsymbol{V} = \{\}$ and $\boldsymbol{E} = \{\}$. For a new query (X, v), if $\mathsf{findM}(v, X) \neq \perp$, then E replaces $\boldsymbol{V}$ with $\boldsymbol{V} \cup \{v\}$. If $\mathsf{findM}(v, X) = \perp$, then E replaces $\boldsymbol{V}$ with $\boldsymbol{V} \cup \{v, u'\}$ and $\boldsymbol{E}$ with $\boldsymbol{E} \cup \{(v, u')\}$, where $u' = \mathsf{E}(X, v) \oplus v$. The edge (v, u') is labeled with X. On the other hand, for a new query (X, u), D replaces $\boldsymbol{V}$ with $\boldsymbol{V} \cup \{v, v \oplus u\}$ and $\boldsymbol{E}$ with $\boldsymbol{E} \cup \{(v, v \oplus u)\}$, where $v = \mathsf{D}(X, u)$.

Function $\mathsf{E}(X, v)$**:**

600: **if** $\mathtt{E}[X, v] = \perp$ **then**
601: $\quad M \leftarrow \mathsf{findM}(v, X)$
602: $\quad$ **if** $M \neq \perp$ **then**
603: $\quad\quad u' \twoheadleftarrow \Sigma^n$
604: $\quad$ **else**
605: $\quad\quad u' \twoheadleftarrow \Sigma^n$
606: $\quad\quad$ **if** $u' \in \boldsymbol{B}_\mathrm{e}$ **then**
607: $\quad\quad\quad bad \leftarrow \mathtt{true}$
608: $\quad\quad\quad u' \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}_\mathrm{e}$
609: $\quad\quad$ **end if**
610: $\quad\quad \boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{u'\}$
611: $\quad$ **end if**
612: $\quad \boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
613: **end if**
614: $\mathtt{E}[X, v] \leftarrow u' \oplus v$
615: **return** $\mathtt{E}[X, v]$

Function $\mathsf{D}(X, u)$**:**

700: **if** $\mathtt{D}[X, u] = \perp$ **then**
701: $\quad v \twoheadleftarrow \Sigma^n$
702: $\quad$ **if** $v \in \boldsymbol{B}_\mathrm{d}$ **then**
703: $\quad\quad bad \leftarrow \mathtt{true}$
704: $\quad\quad v \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}_\mathrm{d}$
705: $\quad$ **end if**
706: **end if**
707: $\boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
708: $\boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{v \oplus u\}$
709: $\mathtt{D}[X, u] \leftarrow v$
710: **return** $\mathtt{D}[X, u]$

Figure 8. Game G3. $\mathcal{H}$, $\mathcal{E}$ and $\mathcal{D}$ are not changed and omitted. $\boldsymbol{B}_\mathrm{e} = \boldsymbol{T} \cup \pi_0^{-1}(\boldsymbol{T}) \cup \pi_1^{-1}(\boldsymbol{T}) \cup \boldsymbol{H} \cup \pi_0^{-1}(\pi_1(\boldsymbol{H})) \cup \pi_1^{-1}(\pi_0(\boldsymbol{H})) \cup \{IV, \pi_0^{-1}(IV), \pi_1^{-1}(IV), \pi_0^{-1}(\pi_1(IV)), \pi_1^{-1}(\pi_0(IV))\}$. $\boldsymbol{B}_\mathrm{d} = \boldsymbol{T} \cup \boldsymbol{H} \cup (u \oplus \boldsymbol{T}) \cup (u \oplus \boldsymbol{H}) \cup \pi_0(\boldsymbol{H}) \cup \pi_1(\boldsymbol{H}) \cup \{IV, u \oplus IV, \pi_0(IV), \pi_1(IV)\}$. Initially, $\boldsymbol{T} = \boldsymbol{H} = \{\}$.

$\boldsymbol{T}$ and $\boldsymbol{H}$ are the sets of tails and heads of edges in $(\boldsymbol{V}, \boldsymbol{E})$, respectively. Vertices with no adjacent edges in $(\boldsymbol{V}, \boldsymbol{E})$ are also in $\boldsymbol{T}$. Initially, $\boldsymbol{T} = \boldsymbol{H} = \{\}$.

findM tries to find a path in $(\boldsymbol{V}, \boldsymbol{E})$ corresponding to the computation $H_{IV}^{F, \{\pi_0, \pi_1\}}(M)$ for some M. Given (v, X) as input, findM first searches a path from IV to $\pi_0^{-1}(v)$ or $\pi_1^{-1}(v)$ in $(\boldsymbol{V}, \boldsymbol{E})$. If IV equals $\pi_0^{-1}(v)$ or $\pi_1^{-1}(v)$, then the single vertex IV is regarded as a path. If findM finds a path, then let $X_1, X_2, \ldots, X_l$ be the labels of the edges on the path. If the path is IV, then $l = 0$, that is, $X_1 \| X_2 \| \cdots \| X_l = \varepsilon$. If there exists some $M \in \Sigma^*$ such that $\mathtt{pad}(M) = X_1 \| X_2 \| \cdots \| X_l \| X$, which depends on whether the terminal of the path is $\pi_0^{-1}(v)$ or $\pi_1^{-1}(v)$, then findM returns M. Otherwise, findM returns $\perp$.

E of G3 always assigns to $\mathtt{E}[X, v]$ a value chosen uniformly at random from Σ^n until *bad* gets true at line 607. D of G3 always assigns to $\mathtt{D}[X, u]$ a value chosen uniformly at random from Σ^n until *bad* gets true at line 703. Thus, G3 is identical to G2 until *bad* gets `true` in G3. Since $|\boldsymbol{T}| \leq \sigma + q_\mathrm{e} + q_\mathrm{d}$

and $|\boldsymbol{H}| \leq \sigma + q_{\mathrm{e}} + q_{\mathrm{d}}$, $|\boldsymbol{B}_{\mathrm{e}}| \leq 6(\sigma + q_{\mathrm{e}} + q_{\mathrm{d}}) + 5$ and $|\boldsymbol{B}_{\mathrm{d}}| \leq 6(\sigma + q_{\mathrm{e}} + q_{\mathrm{d}}) + 4$. E is called at most $(\sigma + q_{\mathrm{e}})$ times and D is called at most q_{d} times and Thus,

$$\begin{aligned}\left|\Pr\left[A^{\mathrm{G2}} = 1\right] - \Pr\left[A^{\mathrm{G3}} = 1\right]\right| &\leq \Pr\left[A^{\mathrm{G3}} \text{ sets } \mathit{bad}\right] \\ &\leq \frac{(6(\sigma + q_{\mathrm{e}} + q_{\mathrm{d}}) + 5)(\sigma + q_{\mathrm{e}})}{2^n} + \frac{(6(\sigma + q_{\mathrm{e}} + q_{\mathrm{d}}) + 4)q_{\mathrm{d}}}{2^n} \\ &= \frac{6(\sigma + q_{\mathrm{e}} + q_{\mathrm{d}})^2 + 5\sigma + 5q_{\mathrm{e}} + 4q_{\mathrm{d}}}{2^n} \ .\end{aligned}$$

For the game G4 in Figure 9, the lines from 605 to 609 of G3 are replaced with the line 605 of G4, and the lines from 701 to 705 of G3 are replaced with the line 701 of G4. Since these changes do not affect the behavior, $\Pr[A^{\mathrm{G3}} = 1] = \Pr[A^{\mathrm{G4}} = 1]$.

Function $\mathsf{E}(X, v)$**:**

600: **if** $\mathtt{E}[X, v] = \perp$ **then**
601: $\quad M \leftarrow \mathsf{findM}(v, X)$
602: $\quad$ **if** $M \neq \perp$ **then**
603: $\quad\quad u' \twoheadleftarrow \Sigma^n$
604: $\quad$ **else**
605: $\quad\quad u' \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}_{\mathrm{e}}$
606: $\quad\quad \boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{u'\}$
607: $\quad$ **end if**
608: $\quad \boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
609: **end if**
610: $\mathtt{E}[X, v] \leftarrow u' \oplus v$
611: **return** $\mathtt{E}[X, v]$

Function $\mathsf{D}(X, u)$**:**

700: **if** $\mathtt{D}[X, u] = \perp$ **then**
701: $\quad v \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}_{\mathrm{d}}$
702: **end if**
703: $\boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
704: $\boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{v \oplus u\}$
705: $\mathtt{D}[X, u] \leftarrow v$
706: **return** $\mathtt{D}[X, u]$

Figure 9. Game G4. $\mathcal{H}$, $\mathcal{E}$ and $\mathcal{D}$ are not changed and omitted.

The game G5 is given in Figure 10. It introduces a variable-input-length random oracle H, which is implemented by lazy evaluation. Initially, $\mathtt{H}[M] = \perp$ for every $M \in \Sigma^*$. H may receive repeated queries since it is called by both $\mathcal{H}$ and $\mathcal{F}$. Different from E of G4, E of G5 assigns $\mathsf{H}(M)$ to u' at the line 603. Different from $\mathcal{H}$ of G4, $\mathcal{H}$ of G5 returns $\mathsf{H}(M)$. We will see that G5 is actually equivalent to G4 in spite of these changes.

First, let us see some properties of the graph $(\boldsymbol{V}, \boldsymbol{E})$. At the beginning of each run of E with (X, v) such that $\mathtt{E}[X, v] = \perp$, $\boldsymbol{V} \subseteq \boldsymbol{T} \cup \boldsymbol{H}$. Whenever u' is added to both $\boldsymbol{V}$ and $\boldsymbol{H}$ by this run, it is chosen from $\Sigma^n \setminus \boldsymbol{B}_{\mathrm{e}}$, where $\boldsymbol{T} \cup \boldsymbol{H} \cup \{IV\} \subseteq \boldsymbol{B}_{\mathrm{e}}$. On the other hand, at the beginning of each run of D with (X, u) such that $\mathtt{D}[X, u] = \perp$, $\boldsymbol{V} \subseteq \boldsymbol{T} \cup \boldsymbol{H}$. Then, v is chosen from $\Sigma^n \setminus \boldsymbol{B}_{\mathrm{d}}$, and $v \oplus u$ is added to both $\boldsymbol{V}$ and $\boldsymbol{H}$ by this run, where $\boldsymbol{T} \cup \boldsymbol{H} \cup \{IV\} \cup (u \oplus (\boldsymbol{T} \cup \boldsymbol{H} \cup \{IV\})) \subseteq \boldsymbol{B}_{\mathrm{d}}$. Thus, every vertex in $(\boldsymbol{V}, \boldsymbol{E})$ has at most one incoming edge, and IV has no incoming edge. It implies that every vertex in $(\boldsymbol{V}, \boldsymbol{E})$ has at most one simple path from IV. In addition, every path $(v_1, v_2, \ldots, v_l)$ with $v_1 = IV$ is constructed only by queries to E, and v_i's are added to $(\boldsymbol{V}, \boldsymbol{E})$ in this order. Furthermore, before v_i is added to $(\boldsymbol{V}, \boldsymbol{E})$, neither $\pi_0(v_i)$ nor $\pi_1(v_i)$ existed in $(\boldsymbol{V}, \boldsymbol{E})$ since $\pi_0^{-1}(\boldsymbol{T}) \cup \pi_1^{-1}(\boldsymbol{T}) \cup \{\pi_0^{-1}(IV), \pi_1^{-1}(IV)\} \subseteq \boldsymbol{B}_{\mathrm{e}}$. Neither $\pi_0(v_i)$ nor $\pi_1(v_i)$ are added to $(\boldsymbol{V}, \boldsymbol{E})$ as tails by the queries to D after v_i since $\pi_0(\boldsymbol{H}) \cup \pi_1(\boldsymbol{H}) \cup \{\pi_0(IV), \pi_1(IV)\} \subseteq \boldsymbol{B}_{\mathrm{d}}$.

Interface $\mathcal{H}(M)$**:**

100: $X_1 \| X_2 \| \cdots \| X_x \leftarrow \mathsf{pad}(M)$
101: $v_0 \leftarrow IV$
102: **for** $1 \le i \le x-1$ **do**
103: $\quad v_i \leftarrow \mathsf{E}(X_i, v_{i-1}) \oplus v_{i-1}$
104: **end for**
105: **if** $|M| > 0$ and $|M| \equiv 0 \pmod{w}$ **then**
106: $\quad v_x \leftarrow \mathsf{E}(X_x, \pi_0(v_{x-1})) \oplus \pi_0(v_{x-1})$
107: **else**
108: $\quad v_x \leftarrow \mathsf{E}(X_x, \pi_1(v_{x-1})) \oplus \pi_1(v_{x-1})$
109: **end if**
110: **return** $\mathsf{H}(M)$

Function $\mathsf{H}(M)$**:**

500: **if** $\mathtt{H}[M] = \perp$ **then**
501: $\quad \mathtt{H}[M] \twoheadleftarrow \Sigma^n$
502: **end if**
503: **return** $\mathtt{H}[M]$

Interface $\mathcal{E}(X, v)$**:**

200: **return** $\mathsf{E}(v, X)$

Function $\mathsf{E}(X, v)$**:**

600: **if** $\mathtt{E}[X, v] = \perp$ **then**
601: $\quad M \leftarrow \mathsf{findM}(v, X)$
602: $\quad$ **if** $M \neq \perp$ **then**
603: $\quad\quad u' \leftarrow \mathsf{H}(M)$
604: $\quad$ **else**
605: $\quad\quad u' \twoheadleftarrow \Sigma^n \setminus \boldsymbol{B}_{\mathrm{e}}$
606: $\quad\quad \boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{u'\}$
607: $\quad$ **end if**
608: $\quad \boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{v\}$
609: **end if**
610: $\mathtt{E}[X, v] \leftarrow u' \oplus v$
611: **return** $\mathtt{E}[X, v]$

Figure 10. Game G5. $\mathcal{D}$ and D are not changed and omitted. Initially, $\mathtt{H}[M] = \perp$ for every $M \in \Sigma^*$.

Suppose that $\mathsf{findM}(v, X)$ finds two paths in $(\boldsymbol{V}, \boldsymbol{E})$. Then, one is from IV to $\pi_0^{-1}(v)$ and the other is from IV to $\pi_1^{-1}(v)$. Notice that $\pi_0^{-1}(v) \neq \pi_1^{-1}(v)$ since $\pi_0(u) \neq \pi_1(u)$ for every $u \in \Sigma^n$. Suppose that both paths have two or more vertices. Then, both $\pi_0^{-1}(v)$ and $\pi_1^{-1}(v)$ are elements of $\boldsymbol{H}$, which implies that one was added to $\boldsymbol{H}$ after the other since at most one vertex is added to $\boldsymbol{H}$ during each run of E. It contradicts $\pi_{1\oplus b}^{-1}(\pi_b(\boldsymbol{H})) \subseteq \boldsymbol{B}_{\mathrm{e}}$ for $b \in \Sigma$. Suppose that one path is the single vertex IV and the other has two or more vertices. $\pi_b^{-1}(v) = IV$ contradicts $\pi_{1\oplus b}^{-1}(\pi_b(IV)) \subseteq \boldsymbol{B}_{\mathrm{e}}$ for $b \in \Sigma$. Thus, $\mathsf{findM}(v, X)$ finds at most a single path in $(\boldsymbol{V}, \boldsymbol{E})$.

In G5, for a new query (v, X) to E, suppose that findM finds a path in $(\boldsymbol{V}, \boldsymbol{E})$ and returns M corresponding to the path and (v, X). Then, M is a new query to H, that is $\mathtt{H}[M] = \perp$, and it is assigned an element chosen uniformly at random from Σ^n. On the other hand, for $\mathcal{H}$, $v_x = \mathsf{H}(M)$. Thus, G5 is equivalent to G4, and $\Pr[A^{\mathrm{G5}} = 1] = \Pr[A^{\mathrm{G4}} = 1]$.

From G5 to G6, E and D change, which are given in Figure 11. E of G6 is augmented with the lines from 600 to 606 and the lines from 614 to 616. $\boldsymbol{H}_A$ is the set of heads of edges in $(\boldsymbol{V}, \boldsymbol{E})$ in the view of A. Initially, $\boldsymbol{H}_A = \{\}$. These changes do not affect the output of E. D of G6 is augmented with the lines from 700 to 704 and the line 710. These changes do not affect the output of D, either. Thus, G6 is equivalent to G5, and $\Pr[A^{\mathrm{G6}} = 1] = \Pr[A^{\mathrm{G5}} = 1]$.

From G6 to G7, only $\mathcal{H}$ changes. $\mathcal{H}$ of G7 is identical to that of G8, which is given in Figure 12. In G7, $\mathcal{H}(M)$ does not call E and just returns $\mathsf{H}(M)$. In G7, E is called only by $\mathcal{E}$ and it does not receive any repeated queries. D does not receive any repeated queries, either. Thus, *bad* never gets `true` in G7. On the other hand, *bad* may get `true` in G6. $|\boldsymbol{B}_{\mathrm{ae}}| \le 3\sigma$ and $|\boldsymbol{B}_{\mathrm{ad}}| \le 3\sigma$ since $\boldsymbol{B}_{\mathrm{ae}} = (\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_0(\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_1(\boldsymbol{H} \setminus \boldsymbol{H}_A)$, $\boldsymbol{B}_{\mathrm{ad}} = (v \oplus (\boldsymbol{H} \setminus \boldsymbol{H}_A)) \cup (\mathsf{H}(M) \oplus (\pi_0(\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_1(\boldsymbol{H} \setminus \boldsymbol{H}_A)))$, and $|\boldsymbol{H} \setminus \boldsymbol{H}_A| \le \sigma$. A knows at most $6(q_{\mathrm{e}} + q_{\mathrm{d}}) + 5$ elements in $\boldsymbol{B}_{\mathrm{e}}$. Thus,

$$\left|\Pr\left[A^{\mathrm{G6}} = 1\right] - \Pr\left[A^{\mathrm{G7}} = 1\right]\right| \le \Pr\left[A^{\mathrm{G6}} \text{ sets } \mathit{bad}\right] \le \frac{3\sigma(q_{\mathrm{e}} + q_{\mathrm{d}})}{2^n - 6(q_{\mathrm{e}} + q_{\mathrm{d}}) - 5} \ .$$

From G7 to G8, E and D changes. G8 is given in Figure 12. E of G8 is obtained from E of G7 by removing the lines from 600 to 606 and the lines from 614 to 616. Since E does not receive any repeated queries, the lines 607 and 619 are also removed. These changes do not affect the output of E. Similarly, D of G8 is obtained from D of G7 by removing the lines from 700 to 704, the lines 705, 707, and 710. These changes do not affect the output of D. Thus, $\Pr[A^{\mathrm{G8}} = 1] = \Pr[A^{\mathrm{G7}} = 1]$. (E, D) of G8 works as a simulator S of (E, D).

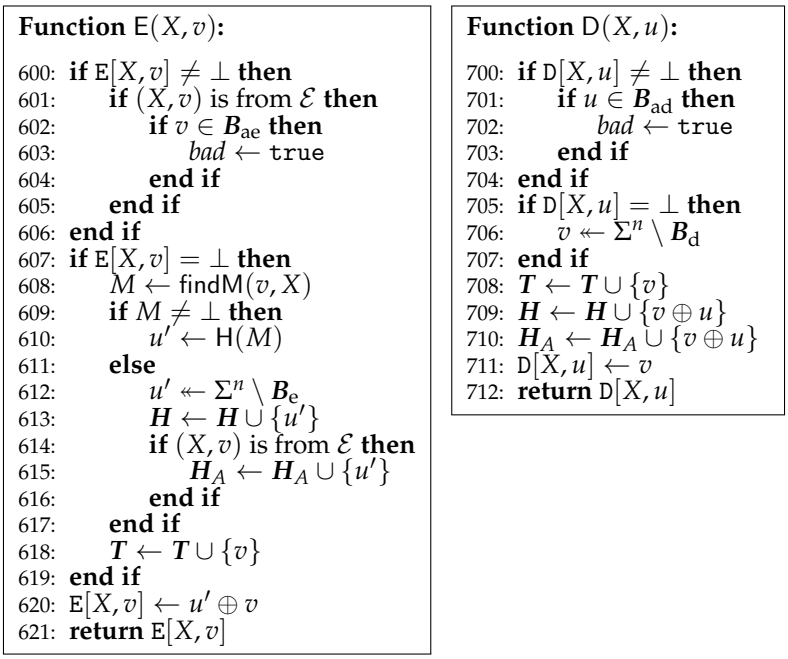


Figure 11. E and D of G6 and G7. $\boldsymbol{B}_{\mathrm{ae}} = (\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_0(\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_1(\boldsymbol{H} \setminus \boldsymbol{H}_A)$. $\boldsymbol{B}_{\mathrm{ad}} = (v \oplus (\boldsymbol{H} \setminus \boldsymbol{H}_A)) \cup (\mathsf{H}(M) \oplus (\pi_0(\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_1(\boldsymbol{H} \setminus \boldsymbol{H}_A)))$, where $\mathtt{D}[X, u] = v$ and $\mathsf{findM}(v, X) = M$. Initially, $\boldsymbol{H}_A = \{\}$.

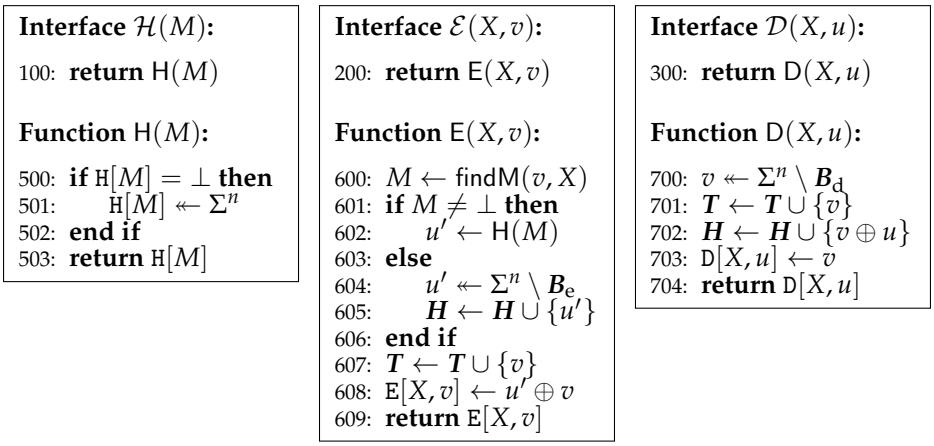


Figure 12. Game G8.

From the discussion above, we have

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{indiff}}_{H^{F,\{\pi_0,\pi_1\}}_{IV},S}(A) &= \left|\Pr\left[A^{\mathrm{G1}} = 1\right] - \Pr\left[A^{\mathrm{G8}} = 1\right]\right| \\
&\leq \Pr\left[A^{\mathrm{G2}} \text{ sets } bad\right] + \Pr\left[A^{\mathrm{G3}} \text{ sets } bad\right] + \Pr\left[A^{\mathrm{G6}} \text{ sets } bad\right] \\
&\leq \frac{7(\sigma + q_{\mathrm{e}} + q_{\mathrm{d}})^2 + 5\sigma + 5q_{\mathrm{e}} + 4q_{\mathrm{d}}}{2^n} + \frac{3\sigma(q_{\mathrm{e}} + q_{\mathrm{d}})}{2^n - 6(q_{\mathrm{e}} + q_{\mathrm{d}}) - 5} \\
&\leq \frac{12(\sigma + q_{\mathrm{e}} + q_{\mathrm{d}})^2}{2^n} + \frac{3\sigma(q_{\mathrm{e}} + q_{\mathrm{d}})}{2^n - 6(q_{\mathrm{e}} + q_{\mathrm{d}}) - 5} \ .
\end{aligned}$$

□

6. Application to Sponge Construction

6.1. Scheme

Let $P : \Sigma^b \to \Sigma^b$ be a permutation and $b = w + c$, where b, w and c are positive integers. The sponge hash function using the proposed domain extension consists of the permutation P, permutations π_0 and π_1 over Σ^c, and an initialization vector $IV \in \Sigma^b$. For π_0 and π_1, it is assumed that $\pi_0(u) \neq u$, $\pi_1(u) \neq u$ and $\pi_0(u) \neq \pi_1(u)$ for every $u \in \Sigma^c$.

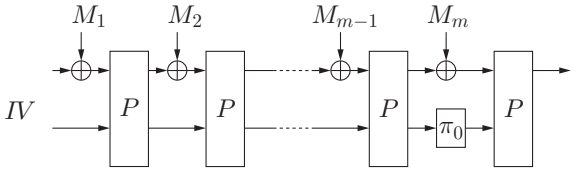
For $y \in \Sigma^b$, let $y = y_\mathrm{r} \| y_\mathrm{c}$, where $y_\mathrm{r} \in \Sigma^w$ and $y_\mathrm{c} \in \Sigma^c$. In the remaining parts, some notations are abused for simplicity. For permutation π over Σ^c and string $y \in \Sigma^b$, $\pi(y)$ represents $y_\mathrm{r} \| \pi(y_\mathrm{c})$. Namely, π is applied to the c least significant bits (LSBs) of y. For strings $y \in \Sigma^b$ and $X \in \Sigma^w$, $y \oplus X$ represents $(y_\mathrm{r} \oplus X) \| y_\mathrm{c}$.

Let π be a permutation over Σ^c. For $1 \leq i \leq x$, let $X_i \in \Sigma^w$. The tweaked sponge construction $\mathsf{S}_{IV}^{P,\pi} : (\Sigma^w)^+ \to \Sigma^n$ is defined as follows: $\mathsf{S}_{IV}^{P,\pi}(X_1 \| X_2 \| \cdots \| X_x) = \hat{v}_x$, where $v_0 \leftarrow IV$, $v_i \leftarrow P(v_{i-1} \oplus X_i)$ for $1 \leq i \leq x-1$, $v_x \leftarrow P(\pi(v_{x-1}) \oplus X_x)$, and $\hat{v}_x$ is the n most significant bits (MSBs) of v_x.

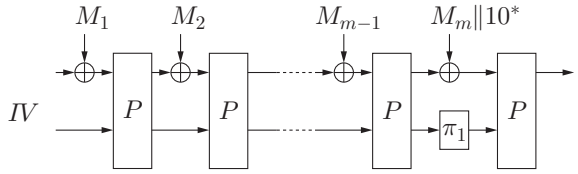
The sponge hash function $G_{IV}^{P,\{\pi_0,\pi_1\}} : \Sigma^* \to \Sigma^n$ based on the proposed domain extension is defined as follows:

$$G_{IV}^{P,\{\pi_0,\pi_1\}}(M) = \begin{cases} \mathsf{S}_{IV}^{P,\pi_0}(\mathtt{pad}(M)) & \text{if } |M| > 0 \text{ and } |M| \equiv 0 \pmod{w}, \\ \mathsf{S}_{IV}^{P,\pi_1}(\mathtt{pad}(M)) & \text{otherwise.} \end{cases}$$

It is also depicted in Figure 13.



(**a**) For M such that $|M| > 0$ and $|M| \equiv 0 \pmod{w}$. $|M_m| = w$.



(**b**) For M such that $|M| = 0$ or $|M| \not\equiv 0 \pmod{w}$. $|M_m| = 0$ if $|M| = 0$ and $1 \leq |M_m| \leq w - 1$ otherwise.

Figure 13. The sponge hash function based on the proposed domain extension. $M = M_1 \| M_2 \| \cdots \| M_m$, where $|M_i| = w$ for $1 \leq i \leq m - 1$.

6.2. IRO in the Ideal Permutation Model

In this section, $P : \Sigma^b \to \Sigma^b$ is assumed to be chosen uniformly at random. The following theorem implies that the proposed hash function is indifferentiable from a random oracle up to the birthday bound.

Theorem 4. *Suppose that the permutation $P : \Sigma^b \to \Sigma^b$ is chosen uniformly at random. Then, for the hash function $G_{IV}^{P,\{\pi_0,\pi_1\}}$, there exists a simulator S of (P, P^{-1}) such that, for any adversary A making at most q_f queries to its FIL forward oracle, q_b queries to its FIL backward oracle, and queries to its VIL oracle which cost at most σ message blocks in total,*

$$\mathrm{Adv}^{\mathrm{indiff}}_{G_{IV}^{P,\{\pi_0,\pi_1\}},S}(A) \le \frac{12(\sigma+q_\mathrm{f}+q_\mathrm{b})^2}{2^c} + \frac{3\sigma(q_\mathrm{f}+q_\mathrm{b})}{2^c-6(q_\mathrm{f}+q_\mathrm{b})-5} \ ,$$

and S makes at most q_f queries.

Proof. Each game provides three interfaces to adversary A: $\mathcal{H}$ for the hash function, $\mathcal{P}$ for the permutation and $\mathcal{P}^{-1}$ for its inverse. It is assumed without loss of generality that A makes no repeated queries both to $\mathcal{H}$ and to $(\mathcal{P},\mathcal{P}^{-1})$. For $\mathcal{P}$ and $\mathcal{P}^{-1}$, once A gets a pair (y,z) such that $P(y)=z$ by a query to $\mathcal{P}$ or $\mathcal{P}^{-1}$, A never makes any query on the pair.

The game G1 is given in Figure 14. $\mathcal{P}$ and $\mathcal{P}^{-1}$ simply call P and P^{-1}, respectively. P and P^{-1} implement P and P^{-1} by lazy evaluation, respectively. $\mathcal{H}$ computes $G_{IV}^{P,\{\pi_0,\pi_1\}}$ with the aid of P and P^{-1}. Thus,

$$\Pr\left[A^{G_{IV}^{P,\{\pi_0,\pi_1\}},(P,P^{-1})}=1\right] = \Pr\left[A^{\mathrm{G1}}=1\right] \ .$$

Notice that P and P^{-1} may receive repeated queries since $\mathcal{H}$ also calls P as well as $\mathcal{P}$.

Interface $\mathcal{H}(M)$:

100: $X_1\|X_2\|\cdots\|X_x \leftarrow \mathtt{pad}(M)$
101: $v_0 \leftarrow IV$
102: **for** $1 \le i \le x-1$ **do**
103: $\quad v_i \leftarrow \mathsf{P}(v_{i-1}\oplus X_i)$
104: **end for**
105: **if** $|M|>0$ and $|M| \equiv 0 \pmod{w}$ **then**
106: $\quad v_x \leftarrow \mathsf{P}(\pi_0(v_{x-1})\oplus X_x)$
107: **else**
108: $\quad v_x \leftarrow \mathsf{P}(\pi_1(v_{x-1})\oplus X_x)$
109: **end if**
110: **return** n MSBs of v_x

Interface $\mathcal{P}(Y)$:

200: **return** $\mathsf{P}(Y)$

Function $\mathsf{P}(Y)$:

600: **if** $\mathtt{P}[Y]=\perp$ **then**
601: $\quad \mathtt{P}[Y] \twoheadleftarrow \Sigma^b \setminus \boldsymbol{Z}$
602: **end if**
603: **return** $\mathtt{P}[Y]$

Interface $\mathcal{P}^{-1}(Z)$:

300: **return** $\mathsf{P}^{-1}(Z)$

Function $\mathsf{P}^{-1}(Z)$:

700: **if** $\mathtt{P}^{-1}[Z]=\perp$ **then**
701: $\quad \mathtt{P}^{-1}[Z] \twoheadleftarrow \Sigma^b \setminus \boldsymbol{Y}$
702: **end if**
703: **return** $\mathtt{P}^{-1}[Z]$

Figure 14. Game G1. For the partial function $\mathtt{P}$ and its inverse $\mathtt{P}^{-1}$, initially, $\mathtt{P}[Y]=\perp$ for every $Y\in\Sigma^b$ and $\mathtt{P}^{-1}[Z]=\perp$ for every $Z\in\Sigma^b$. If Z is assigned to $\mathtt{P}[Y]$, then Y is assigned to $\mathtt{P}^{-1}[Z]$. If Y is assigned to $\mathtt{P}^{-1}[Z]$, then Z is assigned to $\mathtt{P}[Y]$. $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are the sets of values already assigned as inputs and outputs of P and P^{-1}, respectively. Initially, $\boldsymbol{Y}=\boldsymbol{Z}=\{\}$.

From G1 to G2, only P and P^{-1} are changed, which are given in Figure 15. In G2, $\mathtt{P}[Y]$ and $\mathtt{P}^{-1}[Z]$ are chosen uniformly at random from Σ^b. G1 and G2 are identical until *bad* gets `true` in G2. Since P and P^{-1} are called at most $\sigma+q_\mathrm{f}+q_\mathrm{b}$ times in total and $|\boldsymbol{Y}|=|\boldsymbol{Z}|\le\sigma+q_\mathrm{f}+q_\mathrm{b}$,

$$\left|\Pr\left[A^{\mathrm{G1}}=1\right]-\Pr\left[A^{\mathrm{G2}}=1\right]\right| \le \Pr\left[A^{\mathrm{G2}}\text{ sets }\mathit{bad}\right] \le \frac{(\sigma+q_\mathrm{f}+q_\mathrm{b})^2}{2^b} \ .$$

From G2 to G3, only P and P^{-1} are changed, which are given in Figure 16. In G3, P and P^{-1} constructs and maintains a directed graph $(\boldsymbol{V},\boldsymbol{E})$ based on the queries to them. Initially, $\boldsymbol{V}=\{\}$ and $\boldsymbol{E}=\{\}$. For a new query Y, if $\mathsf{findM}(Y)=\perp$, then P replaces $\boldsymbol{V}$ with $\boldsymbol{V}\cup\{Y_\mathrm{c},Z_\mathrm{c}\}$ and $\boldsymbol{E}$ with $\boldsymbol{E}\cup\{(Y_\mathrm{c},Z_\mathrm{c})\}$. If there exists some Z' such that $Z'=IV$ or $\mathtt{P}^{-1}[Z']\ne\perp$, and $Z'_\mathrm{c}=Y_\mathrm{c}$, then the edge $(Y_\mathrm{c},Z_\mathrm{c})$ is labeled with $Z'_\mathrm{r}\oplus Y_\mathrm{r}$. Otherwise, it is labeled with $\perp$. If $\mathsf{findM}(Y)\ne\perp$, then P replaces $\boldsymbol{V}$ with $\boldsymbol{V}\cup\{Y_\mathrm{c}\}$. On the other hand, for a new query Z, P^{-1} replaces $\boldsymbol{V}$ with $\boldsymbol{V}\cup\{Y_\mathrm{c},Z_\mathrm{c}\}$ and $\boldsymbol{E}$ with

$\boldsymbol{E} \cup \{(Y_\mathrm{c}, Z_\mathrm{c})\}$. If there exists some Z' such that $Z' = IV$ or $\mathsf{P}^{-1}[Z'] \neq \perp$, and $Z'_\mathrm{c} = Y_\mathrm{c}$, then the edge $(Y_\mathrm{c}, Z_\mathrm{c})$ is labeled with $Z'_\mathrm{r} \oplus Y_\mathrm{r}$. Otherwise, it is labeled with $\perp$.

Function $\mathsf{P}(Y)$**:**

600: **if** $\mathsf{P}[Y] = \perp$ **then**
601: $\quad \mathsf{P}[Y] \twoheadleftarrow \Sigma^b$
602: $\quad$ **if** $\mathsf{P}[Y] \in \boldsymbol{Z}$ **then**
603: $\quad\quad bad \leftarrow$ `true`
604: $\quad$ **end if**
605: **end if**
606: **return** $\mathsf{P}[Y]$

Function $\mathsf{P}^{-1}(Z)$**:**

700: **if** $\mathsf{P}^{-1}[Z] = \perp$ **then**
701: $\quad \mathsf{P}^{-1}[Z] \twoheadleftarrow \Sigma^b$
702: $\quad$ **if** $\mathsf{P}^{-1}[Z] \in \boldsymbol{Y}$ **then**
703: $\quad\quad bad \leftarrow$ `true`
704: $\quad$ **end if**
705: **end if**
706: **return** $\mathsf{P}^{-1}[Z]$

Figure 15. Game G2. $\mathcal{H}$, $\mathcal{P}$ and $\mathcal{P}^{-1}$, which are not changed, are omitted.

Function $\mathsf{P}(Y)$**:**

600: **if** $\mathsf{P}[Y] = \perp$ **then**
601: $\quad M \leftarrow \mathsf{findM}(Y)$
602: $\quad$ **if** $M \neq \perp$ **then**
603: $\quad\quad Z \twoheadleftarrow \Sigma^b$
604: $\quad$ **else**
605: $\quad\quad Z_\mathrm{r} \twoheadleftarrow \Sigma^w$
606: $\quad\quad Z_\mathrm{c} \twoheadleftarrow \Sigma^c \quad \triangleright\ Z = Z_\mathrm{r} \| Z_\mathrm{c}$
607: $\quad\quad$ **if** $Z_\mathrm{c} \in \boldsymbol{B}_\mathrm{f}$ **then**
608: $\quad\quad\quad bad \leftarrow$ `true`
609: $\quad\quad\quad Z_\mathrm{c} \twoheadleftarrow \Sigma^c \setminus \boldsymbol{B}_\mathrm{f}$
610: $\quad\quad$ **end if**
611: $\quad\quad \boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{Z_\mathrm{c}\}$
612: $\quad$ **end if**
613: $\quad \boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{Y_\mathrm{c}\}$
614: **end if**
615: $\mathsf{P}[Y] \leftarrow Z$
616: **return** $\mathsf{P}[Y]$

Function $\mathsf{P}^{-1}(Z)$**:**

700: **if** $\mathsf{P}^{-1}[Z] = \perp$ **then**
701: $\quad Y_\mathrm{r} \twoheadleftarrow \Sigma^w$
702: $\quad Y_\mathrm{c} \twoheadleftarrow \Sigma^c \quad \triangleright\ Y = Y_\mathrm{r} \| Y_\mathrm{c}$
703: $\quad$ **if** $Y_\mathrm{c} \in \boldsymbol{B}_\mathrm{b}$ **then**
704: $\quad\quad bad \leftarrow$ `true`
705: $\quad\quad Y_\mathrm{c} \twoheadleftarrow \Sigma^b \setminus \boldsymbol{B}_\mathrm{b}$
706: $\quad$ **end if**
707: **end if**
708: $\boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{Y_\mathrm{c}\}$
709: $\boldsymbol{H} \leftarrow \boldsymbol{H} \cup \{Z_\mathrm{c}\}$
710: $\mathsf{P}^{-1}[Z] \leftarrow Y$
711: **return** $\mathsf{P}^{-1}[Z]$

Figure 16. Game G3. $\mathcal{H}$, $\mathcal{P}$ and $\mathcal{P}^{-1}$ are not changed and omitted. $\boldsymbol{B}_\mathrm{f} = \boldsymbol{T} \cup \pi_0^{-1}(\boldsymbol{T}) \cup \pi_1^{-1}(\boldsymbol{T}) \cup \boldsymbol{H} \cup \pi_0^{-1}(\pi_1(\boldsymbol{H})) \cup \pi_1^{-1}(\pi_0(\boldsymbol{H})) \cup \{IV_\mathrm{c}, \pi_0^{-1}(IV_\mathrm{c}), \pi_1^{-1}(IV_\mathrm{c}), \pi_0^{-1}(\pi_1(IV_\mathrm{c})), \pi_1^{-1}(\pi_0(IV_\mathrm{c}))\}$. $\boldsymbol{B}_\mathrm{b} = \boldsymbol{H} \cup \pi_0(\boldsymbol{H}) \cup \pi_1(\boldsymbol{H}) \cup \{IV_\mathrm{c}, \pi_0(IV_\mathrm{c}), \pi_1(IV_\mathrm{c})\}$. Initially, $\boldsymbol{T} = \boldsymbol{H} = \{\}$.

$\boldsymbol{T}$ and $\boldsymbol{H}$ are the sets of tails and heads of edges in $(\boldsymbol{V}, \boldsymbol{E})$, respectively. Vertices with no adjacent edges in $(\boldsymbol{V}, \boldsymbol{E})$ are also in $\boldsymbol{T}$. Initially, $\boldsymbol{T} = \boldsymbol{H} = \{\}$.

findM tries to find a path in $(\boldsymbol{V}, \boldsymbol{E})$ corresponding to the computation $G_{IV}^{P,\{\pi_0,\pi_1\}}(M)$ for some M. Given Y as input, findM first searches a path from IV_c to $\pi_0^{-1}(Y_\mathrm{c})$ or $\pi_1^{-1}(Y_\mathrm{c})$ in $(\boldsymbol{V}, \boldsymbol{E})$. If IV_c equals $\pi_0^{-1}(Y_\mathrm{c})$ or $\pi_1^{-1}(Y_\mathrm{c})$, then the single vertex IV_c is regarded as a path. If findM finds a path, then let $X_1, X_2, \ldots, X_l$ be the labels of the edges on the path. If the path is IV_c, then $l = 0$, that is, $X_1 \| X_2 \| \cdots \| X_l = \varepsilon$. Suppose that $\tilde{Z}_\mathrm{c}$ is the terminal of the path and $\mathsf{P}^{-1}[\tilde{Z}_\mathrm{r} \| \tilde{Z}_\mathrm{c}] \neq \perp$ for some $\tilde{Z}_\mathrm{r}$. If there exists some $M \in \Sigma^*$ such that $\mathsf{pad}(M) = X_1 \| X_2 \| \cdots \| X_l \| (\tilde{Z}_\mathrm{r} \oplus Y_\mathrm{r})$, which depends on whether $\tilde{Z}_\mathrm{c}$ equals $\pi_0^{-1}(Y_\mathrm{c})$ or $\pi_1^{-1}(Y_\mathrm{c})$, then findM returns M. Otherwise, findM returns $\perp$.

P of G3 always assigns to $\mathsf{P}[Y]$ a value chosen uniformly at random from Σ^b until *bad* gets true at line 608. P^{-1} of G3 always assigns to $\mathsf{P}^{-1}[Z]$ a value chosen uniformly at random from Σ^b until *bad* gets true at line 704. Thus, G3 is identical to G2 until *bad* gets `true` in G3. Since $|\boldsymbol{T}| \leq \sigma + q_\mathrm{f} + q_\mathrm{b}$ and $|\boldsymbol{H}| \leq \sigma + q_\mathrm{f} + q_\mathrm{b}$, $|\boldsymbol{B}_\mathrm{f}| \leq 6(\sigma + q_\mathrm{f} + q_\mathrm{b}) + 5$ and $|\boldsymbol{B}_\mathrm{b}| \leq 3(\sigma + q_\mathrm{f} + q_\mathrm{b}) + 3$. P is called at most $(\sigma + q_\mathrm{f})$ times and P^{-1} is called at most q_b times. Thus,

$$\begin{aligned}\left|\Pr[A^{\mathrm{G2}}=1]-\Pr[A^{\mathrm{G3}}=1]\right| &\le \Pr[A^{\mathrm{G3}}\text{ sets }bad]\\ &\le \frac{(6(\sigma+q_{\mathrm{f}}+q_{\mathrm{b}})+5)(\sigma+q_{\mathrm{f}})}{2^c}+\frac{(3(\sigma+q_{\mathrm{f}}+q_{\mathrm{b}})+3)q_{\mathrm{b}}}{2^c}\\ &\le \frac{6(\sigma+q_{\mathrm{f}}+q_{\mathrm{b}})^2+5(\sigma+q_{\mathrm{f}}+q_{\mathrm{b}})}{2^c}\ .\end{aligned}$$

For the game G4 in Figure 17, the lines from 606 to 610 of G3 are replaced with the line 606 of G4, and the lines from 702 to 706 of G3 are replaced with the line 702 of G4. Since these changes do not affect the behaviour, $\Pr[A^{\mathrm{G3}}=1]=\Pr[A^{\mathrm{G4}}=1]$.

Function $\mathsf{P}(Y)$**:**

600: **if** $\mathtt{P}[Y]=\perp$ **then**
601: $\quad M\leftarrow \mathsf{findM}(Y)$
602: $\quad$ **if** $M\neq\perp$ **then**
603: $\quad\quad Z \twoheadleftarrow \Sigma^b$
604: $\quad$ **else**
605: $\quad\quad Z_{\mathrm{r}} \twoheadleftarrow \Sigma^w$
606: $\quad\quad Z_{\mathrm{c}} \twoheadleftarrow \Sigma^c\setminus \boldsymbol{B}_{\mathrm{f}}$
607: $\quad\quad \boldsymbol{H}\leftarrow \boldsymbol{H}\cup\{Z_{\mathrm{c}}\}$
608: $\quad$ **end if**
609: $\quad \boldsymbol{T}\leftarrow \boldsymbol{T}\cup\{Y_{\mathrm{c}}\}$
610: **end if**
611: $\mathtt{P}[Y]\leftarrow Z$
612: **return** $\mathtt{P}[Y]$

Function $\mathsf{P}^{-1}(Z)$**:**

700: **if** $\mathtt{P}^{-1}[Z]=\perp$ **then**
701: $\quad Y_{\mathrm{r}} \twoheadleftarrow \Sigma^w$
702: $\quad Y_{\mathrm{c}} \twoheadleftarrow \Sigma^c\setminus \boldsymbol{B}_{\mathrm{b}}$
703: **end if**
704: $\boldsymbol{T}\leftarrow \boldsymbol{T}\cup\{Y_{\mathrm{c}}\}$
705: $\boldsymbol{H}\leftarrow \boldsymbol{H}\cup\{Z_{\mathrm{c}}\}$
706: $\mathtt{P}^{-1}[Z]\leftarrow Y$
707: **return** $\mathtt{P}^{-1}[Z]$

Figure 17. Game G4. $\mathcal{H}$, $\mathcal{P}$ and $\mathcal{P}^{-1}$ are not changed and omitted.

The game G5 is given in Figure 18. It introduces a variable-input-length random oracle H, which is implemented by lazy evaluation. Initially, $\mathtt{H}[M]=\perp$ for every $M\in\Sigma^*$. H may receive repeated queries since it is called by both $\mathcal{H}$ and $\mathcal{P}$. Different from P of G4, P of G5 assigns to Z an element chosen uniformly at random from $\{\mathsf{H}(M)\}\times\Sigma^{b-n}$ at the line 603. Different from $\mathcal{H}$ of G4, $\mathcal{H}$ of G5 returns $\mathsf{H}(M)$. We will see that G5 is actually equivalent to G4 in spite of these changes.

Interface $\mathcal{H}(M)$**:**

100: $X_1\|X_2\|\cdots\|X_x\leftarrow \mathtt{pad}(M)$
101: $v_0\leftarrow IV$
102: **for** $1\le i\le x-1$ **do**
103: $\quad v_i\leftarrow \mathsf{P}(v_{i-1}\oplus X_i)$
104: **end for**
105: **if** $|M|>0$ and $|M|\equiv 0 \pmod w$ **then**
106: $\quad v_x\leftarrow \mathsf{P}(\pi_0(v_{x-1})\oplus X_x)$
107: **else**
108: $\quad v_x\leftarrow \mathsf{P}(\pi_1(v_{x-1})\oplus X_x)$
109: **end if**
110: **return** $\mathsf{H}(M)$

Function $\mathsf{H}(M)$**:**

500: **if** $\mathtt{H}[M]=\perp$ **then**
501: $\quad \mathtt{H}[M]\twoheadleftarrow \Sigma^n$
502: **end if**
503: **return** $\mathtt{H}[M]$

Interface $\mathcal{P}(Y)$**:**

200: **return** $\mathsf{P}(Y)$

Function $\mathsf{P}(Y)$**:**

600: **if** $\mathtt{P}[Y]=\perp$ **then**
601: $\quad M\leftarrow \mathsf{findM}(Y)$
602: $\quad$ **if** $M\neq\perp$ **then**
603: $\quad\quad Z\twoheadleftarrow \{\mathsf{H}(M)\}\times\Sigma^{b-n}$
604: $\quad$ **else**
605: $\quad\quad Z_{\mathrm{r}}\twoheadleftarrow \Sigma^w$
606: $\quad\quad Z_{\mathrm{c}}\twoheadleftarrow \Sigma^c\setminus \boldsymbol{B}_{\mathrm{f}}$
607: $\quad\quad \boldsymbol{H}\leftarrow \boldsymbol{H}\cup\{Z_{\mathrm{c}}\}$
608: $\quad$ **end if**
609: $\quad \boldsymbol{T}\leftarrow \boldsymbol{T}\cup\{Y_{\mathrm{c}}\}$
610: **end if**
611: $\mathtt{P}[Y]\leftarrow Z$
612: **return** $\mathtt{P}[Y]$

Figure 18. Game G5. $\mathcal{P}^{-1}$ and P^{-1} are not changed and omitted. Initially, $\mathtt{H}[M]=\perp$ for every $M\in\Sigma^*$.

First, let us see some properties of the graph $(\boldsymbol{V},\boldsymbol{E})$. At the beginning of each run of P with Y such that $\mathtt{P}[Y]=\perp$, $\boldsymbol{V}\subseteq \boldsymbol{T}\cup\boldsymbol{H}$. Whenever Z_{c} is added to both $\boldsymbol{V}$ and $\boldsymbol{H}$ by this run, it is chosen from $\Sigma^c\setminus\boldsymbol{B}_{\mathrm{f}}$, where $\boldsymbol{T}\cup\boldsymbol{H}\cup\{IV_{\mathrm{c}}\}\subseteq\boldsymbol{B}_{\mathrm{f}}$. On the other hand, at the beginning of each run of P^{-1} with Z such that

$\mathsf{P}^{-1}[Z] = \perp$, $\boldsymbol{V} \subseteq \boldsymbol{T} \cup \boldsymbol{H}$. Then, Y_c is chosen from $\Sigma^n \setminus \boldsymbol{B}_\mathrm{b}$, where $\boldsymbol{H} \cup \{IV_\mathrm{c}\} \subseteq \boldsymbol{B}_\mathrm{b}$. Thus, every vertex in $(\boldsymbol{V}, \boldsymbol{E})$ has at most one incoming edge labeled with some element in Σ^w, and every incoming edge of IV_c is labeled with $\perp$. It implies that every vertex in $(\boldsymbol{V}, \boldsymbol{E})$ has at most one simple path from IV_c without edges labeled by $\perp$. In addition, every path $(v_1, v_2, \ldots, v_l)$ with $v_1 = IV_\mathrm{c}$ is constructed only by queries to P, and v_i's are added to $(\boldsymbol{V}, \boldsymbol{E})$ in this order. Furthermore, before v_i is added to $(\boldsymbol{V}, \boldsymbol{E})$, neither $\pi_0(v_i)$ nor $\pi_1(v_i)$ existed in $(\boldsymbol{V}, \boldsymbol{E})$ since $\pi_0^{-1}(\boldsymbol{T}) \cup \pi_1^{-1}(\boldsymbol{T}) \cup \{\pi_0^{-1}(IV_\mathrm{c}), \pi_1^{-1}(IV_\mathrm{c})\} \subseteq \boldsymbol{B}_\mathrm{f}$. Neither $\pi_0(v_i)$ nor $\pi_1(v_i)$ are added to $(\boldsymbol{V}, \boldsymbol{E})$ as tails by the queries to P^{-1} after v_i since $\pi_0(\boldsymbol{H}) \cup \pi_1(\boldsymbol{H}) \cup \{\pi_0(IV_\mathrm{c}), \pi_1(IV_\mathrm{c})\} \subseteq \boldsymbol{B}_\mathrm{b}$.

Suppose that $\mathsf{findM}(Y)$ finds two paths in $(\boldsymbol{V}, \boldsymbol{E})$ without edges labeled by $\perp$. Then, one is from IV_c to $\pi_0^{-1}(Y_\mathrm{c})$ and the other is from IV_c to $\pi_1^{-1}(Y_\mathrm{c})$. Notice that $\pi_0^{-1}(Y_\mathrm{c}) \neq \pi_1^{-1}(Y_\mathrm{c})$ since $\pi_0(v) \neq \pi_1(v)$ for every $v \in \Sigma^c$. Suppose that both paths have two or more vertices. Then, both $\pi_0^{-1}(Y_\mathrm{c})$ and $\pi_1^{-1}(Y_\mathrm{c})$ are elements of $\boldsymbol{H}$, which implies that one was added to $\boldsymbol{H}$ after the other since at most one vertex is added to $\boldsymbol{H}$ during each run of P. It contradicts $\pi_{1\oplus a}^{-1}(\pi_a(\boldsymbol{H})) \subseteq \boldsymbol{B}_\mathrm{f}$ for $a \in \Sigma$. Suppose that one path is the single vertex IV_c and the other has two or more vertices. $\pi_a^{-1}(Y_\mathrm{c}) = IV_\mathrm{c}$ contradicts $\pi_{1\oplus a}^{-1}(\pi_a(IV_\mathrm{c})) \subseteq \boldsymbol{B}_\mathrm{f}$ for $a \in \Sigma$. Thus, $\mathsf{findM}(Y)$ finds at most a single path in $(\boldsymbol{V}, \boldsymbol{E})$ without edges labeled by $\perp$.

In G5, for a new query Y to P, suppose that findM finds a path in $(\boldsymbol{V}, \boldsymbol{E})$ and returns M corresponding to the path and Y. Then, M is a new query to H, that is, $\mathtt{H}[M] = \perp$, and it is assigned an element chosen uniformly at random from Σ^n. On the other hand, for $\mathcal{H}$, the n MSBs of v_x equals $\mathsf{H}(M)$. Thus, G5 is equivalent to G4, and $\Pr[A^{\mathrm{G5}} = 1] = \Pr[A^{\mathrm{G4}} = 1]$.

From G5 to G6, P and P^{-1} change, which are given in Figure 19. P of G6 is augmented with the lines from 600 to 606 and the lines from 615 to 617. $\boldsymbol{H}_A$ is the set of heads of edges in $(\boldsymbol{V}, \boldsymbol{E})$ in the view of A. Initially, $\boldsymbol{H}_A = \{\}$. These changes do not affect the output of P. P^{-1} of G6 is augmented with the lines from 700 to 704 and the line 711. These changes do not affect the output of P^{-1}. Thus, G6 is equivalent to G5, and $\Pr[A^{\mathrm{G6}} = 1] = \Pr[A^{\mathrm{G5}} = 1]$.

From G6 to G7, only $\mathcal{H}$ changes. $\mathcal{H}$ of G7 is identical to that of G8, which is given in Figure 20. In G7, $\mathcal{H}(M)$ does not call P and just returns $\mathsf{H}(M)$. In G7, P is called only by $\mathcal{P}$ and it does not receive any repeated queries. P^{-1} does not receive any repeated queries, either. Thus, *bad* never gets `true` in G7. On the other hand, *bad* may get `true` in G6. $|\boldsymbol{B}_\mathrm{a}| \leq 3\sigma$ since $\boldsymbol{B}_\mathrm{a} = (\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_0(\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_1(\boldsymbol{H} \setminus \boldsymbol{H}_A)$ and $|\boldsymbol{H} \setminus \boldsymbol{H}_A| \leq \sigma$. A knows at most $6(q_\mathrm{f} + q_\mathrm{b}) + 5$ elements in $\boldsymbol{B}_\mathrm{f}$. Thus,

$$\left|\Pr\left[A^{\mathrm{G6}} = 1\right] - \Pr\left[A^{\mathrm{G7}} = 1\right]\right| \leq \Pr\left[A^{\mathrm{G6}} \text{ sets } \mathit{bad}\right] \leq \frac{3\sigma(q_\mathrm{f} + q_\mathrm{b})}{2^c - 6(q_\mathrm{f} + q_\mathrm{b}) - 5}.$$

From G7 to G8, P and P^{-1} change. G8 is given in Figure 20. P of G8 is obtained from P of G7 by removing the lines from 600 to 606 and the lines from 615 to 617. Since P does not receive any repeated queries, the lines 607 and 620 are also removed. These changes do not affect the output of P. Similarly, P^{-1} of G8 is obtained from P^{-1} of G7 by removing the lines from 700 to 704, the lines 705, 708 and 711. These changes do not affect the output of P^{-1}. Thus, $\Pr[A^{\mathrm{G8}} = 1] = \Pr[A^{\mathrm{G7}} = 1]$. $(\mathsf{P}, \mathsf{P}^{-1})$ of G8 works as a simulator S of (P, P^{-1}).

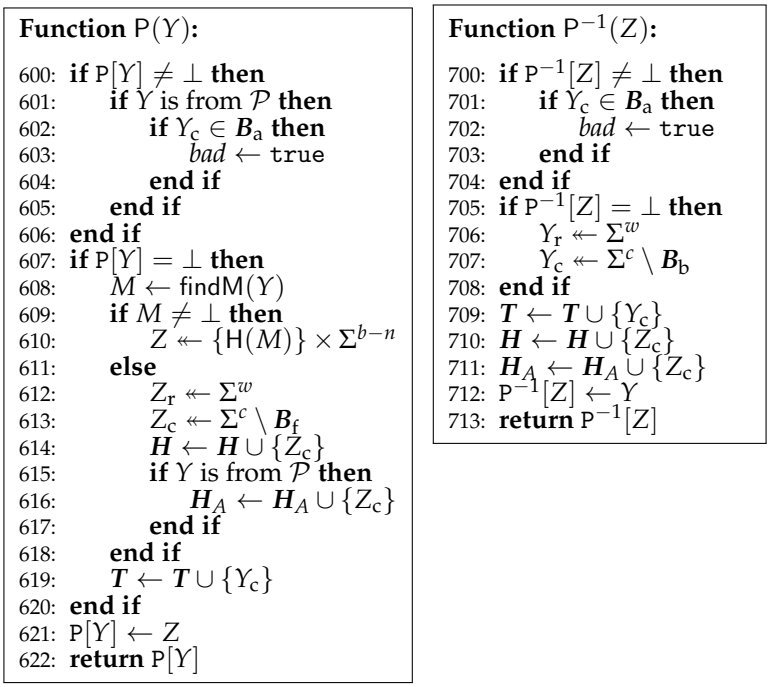


Figure 19. P and P^{-1} of G6 and G7. $\boldsymbol{B}_{\mathrm{a}} = (\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_0(\boldsymbol{H} \setminus \boldsymbol{H}_A) \cup \pi_1(\boldsymbol{H} \setminus \boldsymbol{H}_A)$. Initially, $\boldsymbol{H}_A = \{\}$.

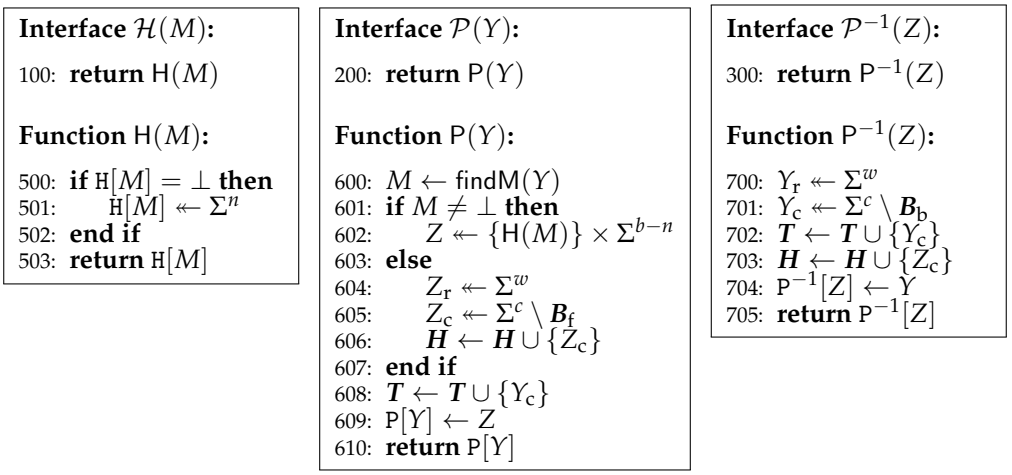


Figure 20. Game G8.

From the discussion above, we have

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{indiff}}_{H^{F,\{\pi_0,\pi_1\}}_{IV},S}(A) &= \left|\Pr\left[A^{\mathrm{G1}} = 1\right] - \Pr\left[A^{\mathrm{G8}} = 1\right]\right| \\
&\leq \Pr\left[A^{\mathrm{G2}} \text{ sets } bad\right] + \Pr\left[A^{\mathrm{G3}} \text{ sets } bad\right] + \Pr\left[A^{\mathrm{G6}} \text{ sets } bad\right] \\
&\leq \frac{(\sigma + q_{\mathrm{f}} + q_{\mathrm{b}})^2}{2^b} + \frac{6(\sigma + q_{\mathrm{f}} + q_{\mathrm{b}})^2 + 5(\sigma + q_{\mathrm{f}} + q_{\mathrm{b}})}{2^c} + \frac{3\sigma(q_{\mathrm{f}} + q_{\mathrm{b}})}{2^c - 6(q_{\mathrm{f}} + q_{\mathrm{b}}) - 5} \\
&\leq \frac{12(\sigma + q_{\mathrm{f}} + q_{\mathrm{b}})^2}{2^c} + \frac{3\sigma(q_{\mathrm{f}} + q_{\mathrm{b}})}{2^c - 6(q_{\mathrm{f}} + q_{\mathrm{b}}) - 5} \ .
\end{aligned}$$

□

7. Conclusions

In this article, a domain extension scheme which extends MDP [5] has been presented for iterated hashing. The collision resistance and indifferentiability from a random oracle of an iterated hash function using the domain extension have been confirmed under reasonable assumptions. For the pseudorandom-function property of the iterated hash function keyed via IV, readers are asked to see [6] for details.

The domain extension can also be applied to the sponge construction. The indifferentiability from a random oracle of the resultant hash function has been confirmed in the ideal permutation model.

The presented domain extension is simple and efficient. It is expected to be useful for lightweight cryptography.

Acknowledgments: This work was supported in part by JSPS KAKENHI Grant Number JP16H02828.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Dang, Q.H. *Secure Hash Standard (SHS)*; FIPS PUB 180-4; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2012.
2. Merkle, R.C. One Way Hash Functions and DES. In *Advances in Cryptology—CRYPTO 89, Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology, Santa Barbara, CA, USA, 20–24 August 1989*; Brassard, G., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1990; Volume 435, pp. 428–446.
3. Damgård, I. A Design Principle for Hash Functions. In *Advances in Cryptology—CRYPTO 89, Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology, Santa Barbara, CA, USA, 20–24 August 1989*; Brassard, G., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1990; Volume 435, pp. 416–427.
4. Bellare, M.; Canetti, R.; Krawczyk, H. Keying Hash Functions for Message Authentication. In *Advances in Cryptology—CRYPTO 96, Proceedings of the 16th Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 1996*; Koblitz, N., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1996; Volume 1109, pp. 1–15.
5. Hirose, S.; Park, J.H.; Yun, A. A Simple Variant of the Merkle-Damgård Scheme with a Permutation. In *Advances in Cryptology—ASIACRYPT 2007, Proceedings of the 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, 2–6 December 2007*; Kurosawa, K., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4833, pp. 113–129.
6. Hirose, S.; Yabumoto, A. A Tweak for a PRF Mode of a Compression Function and Its Applications. In *Innovative Security Solutions for Information Technology and Communications, Proceedings of the 9th International Conference, SECITC 2016, Bucharest, Romania, 9–10 June 2016*; Bica, I., Reyhanitabar, R., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2016; Volume 10006, pp. 103–114.
7. Bagheri, N.; Gauravaram, P.; Knudsen, L.R.; Zenner, E. The suffix-free-prefix-free hash function construction and its indifferentiability security analysis. *Int. J. Inf. Secur.* **2012**, *11*, 419–434. [CrossRef]
8. Nandi, M. Characterizing Padding Rules of MD Hash Functions Preserving Collision Security. In *Information Security and Privacy, Proceedings of the 14th Australasian Conference, ACISP 2009, Brisbane, Australia, 1–3 July 2009*; Boyd, C., Nieto, J.M.G., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5594, pp. 171–184.
9. Coron, J.S.; Dodis, Y.; Malinaud, C.; Puniya, P. Merkle-Damgård Revisited: How to Construct a Hash Function. In *Advances in Cryptology—CRYPTO 2005, Proceedings of the 25th Annual International Cryptology Conference, Santa Barbara, CA, USA, 14-18 August 2005*; Shoup, V., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3621, pp. 430–448.
10. Maurer, U.M.; Renner, R.; Holenstein, C. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In *Theory of Cryptography, Proceedings of the First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, 19-21 February 2004*; Naor, M., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2004; Volume 2951, pp. 21–39.

11. Chang, D.; Lee, S.; Nandi, M.; Yung, M. Indifferentiable Security Analysis of Popular Hash Functions with Prefix-Free Padding. In *Advances in Cryptology—ASIACRYPT 2006, Proceedings of the 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, 3–7 December 2006*; Lai, X., Chen, K., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4284, pp. 283–298.
12. Chang, D.; Nandi, M. Improved Indifferentiability Security Analysis of chopMD Hash Function. In *Fast Software Encryption, Proceedings of the 15th International Workshop, FSE 2008, Lausanne, Switzerland, 10–13 February 2008*; Nyberg, K., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2008; Volume 5086, pp. 429–443.
13. Bellare, M.; Ristenpart, T. Multi-property-preserving hash domain extension and the EMD transform. In *Advances in Cryptology—ASIACRYPT 2006, Proceedings of the 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, 3–7 December 2006*; Lai, X., Chen, K., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4284, pp. 299–314.
14. Kelsey, J. Public Comments on the Draft Federal Information Processing Standard (FIPS) Draft FIPS 180-2, Secure Hash Standard (SHS), 2001. Available online: http://www.cs.utsa.edu/~wagner/CS4363/SHS/dfips-180-2-comments1.pdf (accessed on 9 June 2018).
15. Dworkin, M.J. *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*; NIST Special Publication 800-38B; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2005.
16. Black, J.; Rogaway, P. A Block-Cipher Mode of Operation for Parallelizable Message Authentication. In *Advances in Cryptology—EUROCRYPT 2002, Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, 28 April–2 May 2002*; Knudsen, L.R., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2332, pp. 384–397.
17. Nandi, M. Fast and Secure CBC-Type MAC Algorithms. In *Fast Software Encryption, Proceedings of the 16th International Workshop, FSE 2009, Leuven, Belgium, 22–25 February 2009*; Dunkelman, O., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5665, pp. 375–393.
18. Sarkar, P. Domain extender for collision resistant hash functions: Improving upon Merkle-Damgård iteration. *Discret. Appl. Math.* **2009**, *157*, 1086–1097. [CrossRef]
19. Bertoni, G.; Daemen, J.; Peeters, M.; Assche, G.V. Sufficient conditions for sound tree and sequential hashing modes. *Int. J. Inf. Secur.* **2014**, *13*, 335–353. [CrossRef]
20. Bertoni, G.; Daemen, J.; Peeters, M.; Assche, G.V. Sakura: A Flexible Coding for Tree Hashing. In *Applied Cryptography and Network Security, Proceedings of the 12th International Conference, ACNS 2014, Lausanne, Switzerland, 10–13 June 2014*; Boureanu, I., Owesarski, P., Vaudenay, S., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8479, pp. 217–234.
21. Bertoni, G.; Daemen, J.; Peeters, M.; Van Assche, G. Sponge Functions. In Proceedings of the ECRYPT Hash Workshop 2007, Barcelona, Spain, 24–25 May 2007.
22. Dworkin, M.J. *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*; FIPS PUB 202; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2015.
23. Guo, J.; Peyrin, T.; Poschmann, A. The PHOTON Family of Lightweight Hash Functions. In *Advances in Cryptology—CRYPTO 2011, Proceedings of the 31st Annual Cryptology Conference, Santa Barbara, CA, USA, 14–18 August 2011*; Rogaway, P., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6841, pp. 222–239.
24. Bogdanov, A.; Knezevic, M.; Leander, G.; Toz, D.; Varici, K.; Verbauwhede, I. SPONGENT: A Lightweight Hash Function. In *Cryptographic Hardware and Embedded Systems—CHES 2011, Proceedings of the 13th International Workshop, Nara, Japan, 28 September–1 October 2011*; Preneel, B., Takagi, T., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6917, pp. 312–325.
25. Bertoni, G.; Daemen, J.; Peeters, M.; Van Assche, G. Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In *Selected Areas in Cryptography, Proceedings of the 18th International Workshop, SAC 2011, Toronto, ON, Canada, 11–12 August 2011*; Miri, A., Vaudenay, S., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2011; Volume 7118, pp. 320–337.

26. Rogaway, P.; Shrimpton, T. Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In *Fast Software Encryption, Proceedings of the 11th International Workshop, FSE 2004, Delhi, India, 5–7 February 2004*; Roy, B.K., Meier, W., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3017, pp. 371–388.
27. Bellare, M.; Rogaway, P. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In *Advances in Cryptology—EUROCRYPT 2006, Proceedings of the 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, 28 May–1 June 2006*; Vaudenay, S., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4004, pp. 409–426.
28. Quisquater, J.; Girault, M. 2n-Bit Hash-Functions Using n-Bit Symmetric Block Cipher Algorithms. In *Advances in Cryptology—EUROCRYPT '89, Proceedings of the Workshop on the Theory and Application of of Cryptographic Techniques, Houthalen, Belgium, 10–13 April 1989*; Quisquater, J.J., Vandewalle, J., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1989; Volume 434, pp. 102–109.