Enhanced Security Notions for Dedicated-Key Hash Functions: Definitions and Relationships

Mohammad Reza Reyhanitabar, Willy Susilo, and Yi Mu

Centre for Computer and Information Security Research,
School of Computer Science and Software Engineering
University of Wollongong, Australia
{rezar, wsusilo, ymu}@uow.edu.au

Abstract. In this paper, we revisit security notions for dedicated-key hash functions, considering two essential theoretical aspects; namely, formal definitions for security notions, and the relationships among them. Our contribution is twofold. First, we provide a new set of enhanced security notions for dedicated-key hash functions. The provision of this set of enhanced properties has been motivated by the introduction of enhanced target collision resistance (eTCR) property by Halevi and Krawczyk at Crypto 2006. We notice that the eTCR property does not belong to the set of the seven security notions previously investigated by Rogaway and Shrimpton at FSE 2004; namely: Coll, Sec, aSec, eSec, Pre, aPre and ePre. The fact that eTCR, as a new useful property, is the enhanced variant of the well-known TCR (a.k.a. eSec or UOWHF) property motivates one to investigate the possibility of providing enhanced variants for the other properties. We provide such an enhanced set of properties. Interestingly, there are six enhanced variants of security notions available, excluding "ePre" which can be demonstrated to be non-enhanceable. As the second and main part of our contribution, we provide a full picture of relationships (*i.e.* implications and separations) among the (thirteen) security properties including the (six) enhanced properties and the previously considered seven properties. The implications and separations are supported by formal proofs (reductions) and/or counterexamples in the concrete-security framework.

Key words: hash functions, security notions, definitions, relationships.

1 Introduction

Cryptographic hash functions are widely used in many applications, most importantly in digital signature schemes and message authentication codes (MACs), as well as commitment schemes, password protection, and key derivation, to mention some. Unlike many other cryptographic primitives which are usually aimed to fulfil a specific notion of security, hash functions, as workhorses of cryptography, are often assumed to satisfy a wide application-dependent spectrum of security properties ranging from merely being a one-way function, as the minimum security requirement, to acting as a truly random function in the random oracle model.

Cryptographic hash functions originally were used as "secure" compressing functions to make digital signatures more efficient [12, 26, 18, 19, 10, 11], and this application of hash functions in signature schemes, following hash-and-sign paradigm, requires them to satisfy three well-known classic security properties, namely collision resistance, second-preimage resistance and preimage resistance. These three properties have been traditionally considered as basic "*necessary*" security properties for a hash function to be used in signature schemes as well as several other applications of hash functions.

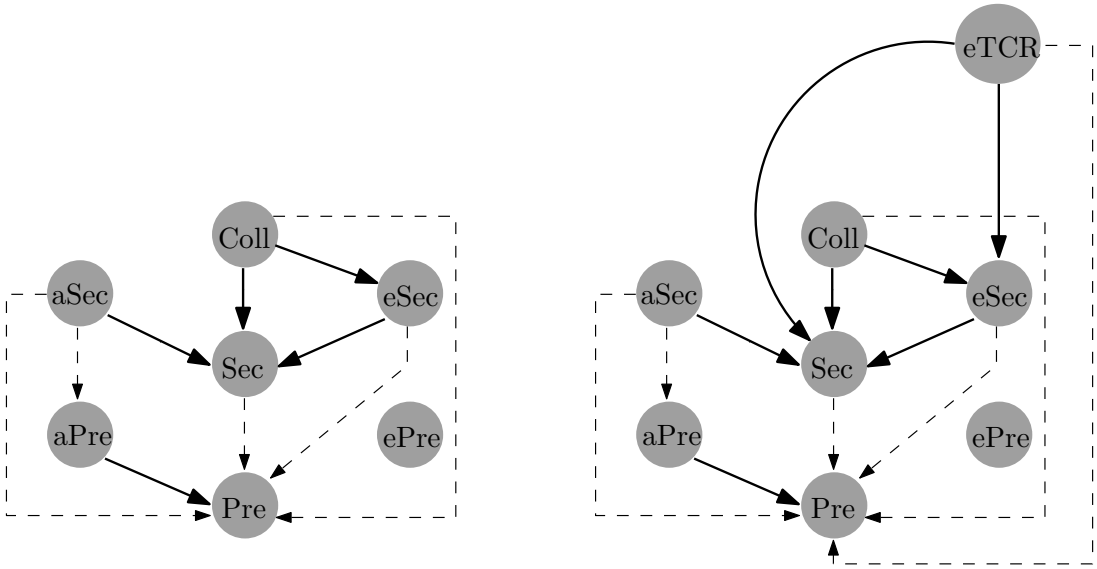
There seems to be no clear consensus on specification of a set of properties that can be considered as a *sufficient* property set for a hash function in the standard model of security [9], and the current literature contains many different informal and formal definitions for some basic and widely-used security properties of hash functions (such as [10, 11, 19, 20, 34, 25, 17, 31, 30]).

For a formal treatment of the security properties and their relationships, it is essential to clearly specify the hash function setting; that is, whether the hash function is specified as a keyless function $H : \mathcal{M} \rightarrow \mathcal{C}$ which only admits an input message, or it is a dedicated-key (*i.e.* two-argument) function $H : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{C}$

with an explicit key input in addition to a message input. A dedicated-key hash function $H : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{C}$ can also be viewed as a family of functions $\{H_K : \mathcal{M} \rightarrow \mathcal{C}\}_{K \in \mathcal{K}}$ by considering the key as the index for the instance functions. Although, historically, most of the widely used hash functions, like MD5 [29], SHA-xxx (for xxx=1, 224, 256, 384, 512) [23, 24], are keyless hash function, the situation seems to be changing in favor of the dedicated-key hash function setting, which has been more *popular* in rigorous formal treatments of hash functions; e.g. [10, 11, 13, 14, 1]. For example, several new (practical and efficient) dedicated-key hash functions have been proposed to the recent SHA-3 hash function competition run by NIST, e.g. SHAvite-3 and Skein, which do have (optional) dedicated-key inputs [21].

Rogaway and Shrimpton [31, 32] provided formal definitions for seven variants of the three basic properties; namely, collision resistance (denoted by 'Coll' in [31]), three variants of second-preimage resistance (Sec, aSec, eSec) and three variants of preimage resistance (Pre, aPre, ePre), as well as, all relationships between these seven properties, in the dedicated-key hash function setting. Figure 1 shows the overall picture of these relationships. We note that the original formal definition of collision resistance and UOWHF properties were proposed in the asymptotic-security framework, by Damgård in [10], and by Naor and Yung in [20]; respectively. UOWHF property was later called as target collision resistance (TCR) by Bellare and Rogaway in [4] (in the concert-security framework), and also renamed as "eSec" according to the nomenclature provided by Rogaway and Shrimpton in [31].

Halevi and Krawczyk at Crypto 2006 [15] introduced "enhanced target collision resistance (eTCR)" property, as a strengthened (or enhanced) variant of TCR property. eTCR is the property sought from the Randomized Hashing construction [15], as recently announced in NIST SP 800- 106 [22], for strengthening digital signatures. Reyhanitabar, Susilo and Mu at FSE 2009 [28] showed a separation between eTCR and Coll properties, and further completed the relationships between eTCR and each of the seven security notions in [27]. Figure 1 also depicts these relationships.



Seven security properties for hash functions and their relationships investigated by Rogaway and Shrimpton in [32].

Relationships between eTCR and each of the seven security properties for hash functions investigated by Reyhanitabar, Susilo, and Mu in [27, 28].

Fig. 1. Known relationships among security notions for dedicated-key hash functions: a directed path shows an implication (dashed lines represent "provisional implications" in which the strength of the implications depends on the amount of compression achieved by the hash function) and lack of a path shows a separation [31, 32, 28, 27].

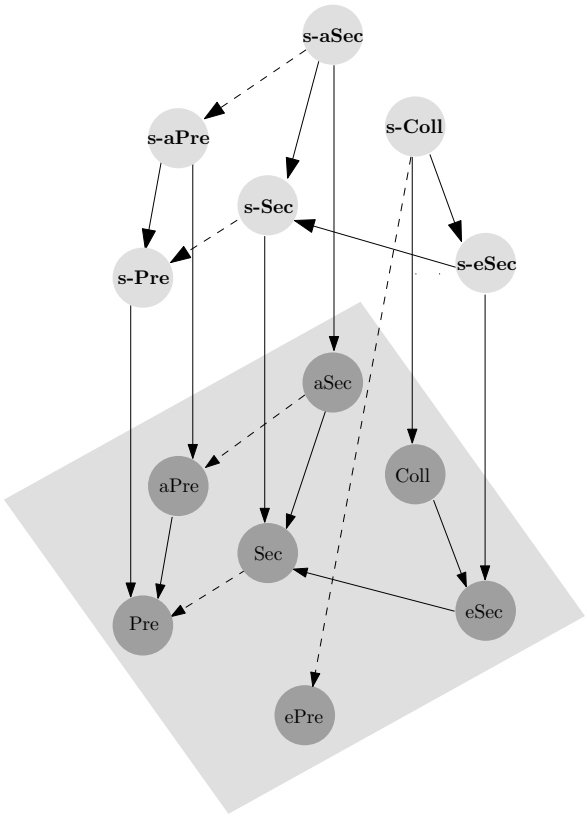
In this paper we continue this interesting line of research by further investigating the security notions for *dedicated-key* hash functions. The fact that the interesting eTCR property is an enhanced variant of the well-known TCR (a.k.a. eSec or UOWHF) property has been our main motivation to investigate the possibility of further completing the set of current security notions for dedicated-key hash functions, by providing enhanced variants for the other properties. We note that an enhanced variant of collision resistance property, called "eColl", also was recently noticed by Yasuda in [33].

NOMENCLATURE. For the seven security notions, we use the same nomenclature; *i.e.* Coll, Sec, aSec, eSec, Pre, aPre, ePre, as proposed by Rogaway and Shrimpton in [31]. The remaining six new strengthened variants (among the thirteen properties) are denoted by adding a prefix 's-' to the names of the related (weaker) notions; that is, s-Coll, s-Sec, s-aSec, s-eSec, s-Pre, s-aPre; respectively, where s-Coll is the strengthened variant of Coll, and so forth. We use prefix $s-$ (for 'strengthened') instead of $e-$ (for 'enhanced'), to prevent any ambiguity among the names, as the prefix 'e' has already been used by Rogaway and Shrimpton in [31] to stand for '*everywhere*' variants in eSec and ePre properties. Note that now according to our new notations, 's-eSec' stands for the 'eTCR' property of [15] and s-Coll is the same property as eColl in [33].

OUR CONTRIBUTIONS. First, we provide a new extended set of strengthened (enhanced) security notions for dedicated-key hash functions, which include the eTCR property put forth by Halevi and Krawczyk in [15] (denoted by 's-eSec' in this paper), eColl property introduced by Yasuda in [33] (denoted as 's-Coll' in this paper), as well as four new properties which we introduce in this paper; namely, s-Sec, s-aSec, s-Pre, s-aPre. Then, as our second and main contribution, we work out all relationships among the (thirteen) security properties including the (six) enhanced properties; *i.e.* s-Coll, s-Sec, s-aSec, s-eSec, s-Pre, s-aPre , and the well-known seven properties; *i.e.* Coll, Sec, aSec, eSec, Pre, aPre, ePre.

Figure 2 illustrates the relationships among the security notions. A solid directed edge '$A \rightarrow B$' shows a security-preserving reduction from the notion A to the notion B, and a dashed directed edge '$A \dashrightarrow B$' represents a provisional reduction (i.e. with some security loss) from A to B. (Formal definitions of the security-preserving and provisional implications are given in Sec. 3.) The top graph illustrates the *essential* "edges" that can be composed to construct the "paths" showing all other implications; for instance, combining Coll $\rightarrow$ eSec and eSec $\rightarrow$ Sec edges one gets Coll $\rightarrow$ Sec (which is not explicitly shown in the graph), and so on. The lack of a directed *path* from A to B in the graph means a separation. The three tables below the graph detail all the relationships, where an entry at row A and column B shows whether the property A implies the property B, or there is a separation; trivial equivalences are denoted by '='.

NOTATIONS. If A is a randomized algorithm then by $y = A(x_1, \cdots, x_n; R)$ it is meant that y is the output of A on inputs $x_1, \cdots, x_n$ when it is provided with random coins (tape) R. By $y \xleftarrow{\$} A(x_1, \cdots, x_n)$ it is meant that the tape R is chosen at random and y is set to be $y = A(x_1, \cdots, x_n; R)$. To show that an algorithm A is run without any input (*i.e.* when the input is an empty string) we use either the notation $y \xleftarrow{\$} A()$ or $y \xleftarrow{\$} A(\varnothing)$. By time complexity of an algorithm we mean the running time, relative to some fixed model of computation (e.g. RAM) plus the size of the description of the algorithm using some fixed encoding method. If X is a finite set, by $x \xleftarrow{\$} X$ it is meant that x is chosen from X uniformly at random. For a binary string $M = M_1||M_2||\cdots||M_m$, let $M_{1\ldots n}$ denote the first n bits of M and $|M|$ denote its length in bits (where $n \leq m = |M|$). Let $val(.)$ be a function that on input a binary string M, considered as an unsigned binary number (with some fixed bit position numbering), returns its decimal value. For a positive integer m, let $\langle m \rangle_b$ denotes binary representation of m by a string of length exactly b bits. If S is a finite set we denote size of S by $|S|$. The set of all binary strings of length n bits (for some positive integer n) is denoted as $\{0,1\}^n$, the set of all binary strings whose lengths are variable but upper-bounded by N is denoted by $\{0,1\}^{\leq N}$ and the set of all binary strings of arbitrary length is denoted by $\{0,1\}^*$.



	s-Coll (eColl)	s-Sec	s-aSec	s-eSec (eTCR)	s-Pre	s-aPre
s-Coll (eColl)	=	$\rightarrow$	$\not\rightarrow$	$\rightarrow$ [33]	$\dashrightarrow$	$\not\rightarrow$
s-Sec	$\not\rightarrow$	=	$\not\rightarrow$	$\not\rightarrow$	$\dashrightarrow$	$\not\rightarrow$
s-aSec	$\not\rightarrow$	$\rightarrow$	=	$\not\rightarrow$	$\dashrightarrow$	$\dashrightarrow$
s-eSec (eTCR)	$\not\rightarrow$	$\rightarrow$	$\not\rightarrow$	=	$\dashrightarrow$	$\not\rightarrow$
s-Pre	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	=	$\not\rightarrow$
s-aPre	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\rightarrow$	=

	Coll	Sec	aSec	eSec (TCR)	Pre	aPre	ePre
s-Coll (eColl)	$\rightarrow$	$\rightarrow$	$\not\rightarrow$	$\rightarrow$	$\dashrightarrow$	$\not\rightarrow$	$\dashrightarrow$
s-Sec	$\not\rightarrow$	$\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\dashrightarrow$	$\not\rightarrow$	$\not\rightarrow$
s-aSec	$\not\rightarrow$	$\rightarrow$	$\rightarrow$	$\not\rightarrow$	$\dashrightarrow$	$\dashrightarrow$	$\not\rightarrow$
s-eSec (eTCR)	$\not\rightarrow$ [28]	$\rightarrow$ [27]	$\not\rightarrow$ [27]	$\rightarrow$ [27]	$\dashrightarrow$ [27]	$\not\rightarrow$ [27]	$\not\rightarrow$ [27]
s-Pre	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\rightarrow$	$\not\rightarrow$	$\not\rightarrow$
s-aPre	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\rightarrow$	$\rightarrow$	$\not\rightarrow$

	s-Coll	s-Sec	s-aSec	s-eSec (eTCR)	s-Pre	s-aPre
Coll	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$ [28]	$\not\rightarrow$	$\not\rightarrow$
Sec	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$ [27]	$\not\rightarrow$	$\not\rightarrow$
aSec	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$ [27]	$\not\rightarrow$	$\not\rightarrow$
eSec (TCR)	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$ [27]	$\not\rightarrow$	$\not\rightarrow$
Pre	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$ [27]	$\not\rightarrow$	$\not\rightarrow$
aPre	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$ [27]	$\not\rightarrow$	$\not\rightarrow$
ePre	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$	$\not\rightarrow$ [27]	$\not\rightarrow$	$\not\rightarrow$

Fig. 2. A full picture of the relationships among the security notions. Note that the top graph only illustrates the *essential* "edges" that can be composed to construct the "paths" showing all other implications. The lack of a directed *path* in the graph means a separation, while separations are explicitly denoted by $\nrightarrow$ in the tables.

2 Definitions of Security Notions

In this section, adopting the conventions of the concrete-security framework [8, 3, 7, 6, 5] , we provide definitions of the security notions for a dedicated-key hash function $H : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$, where $\mathcal{C} = \{0,1\}^n$ for some positive integer n, the key space $\mathcal{K}$ is some nonempty *finite* set and the message space $\mathcal{M} \subseteq \{0,1\}^*$; such that $\{0,1\}^\delta \subseteq \mathcal{M}$ for at least a positive integer δ. For any $M \in \mathcal{M}$ and $K \in \mathcal{K}$, we use the notations $H_K(M)$ and $H(K,M)$ interchangeably. Note that this description of a hash function is generic enough to be applied when one considering: a Fixed-Input-Length (FIL) hash function (*i.e.* a compression function), where $\mathcal{M} = \{0,1\}^m$; a Variable-Input-Length (VIL) hash function, where $\mathcal{M} = \{0,1\}^{<\lambda}$ for some (huge) value λ (e.g. $\lambda = 2^{64}$ as in SHA-1); or even an Arbitrary-Input-Length (AIL) hash function, where $\mathcal{M} = \{0,1\}^*$.

Let $T_{H,\delta}$ denote the time complexity of the most efficient algorithm that can compute $H(K,M)$, for any $M \in \{0,1\}^\delta \subseteq \mathcal{M}$ and $K \in \mathcal{M}$, plus the time complexity of the most efficient algorithm that can sample from the finite set $\mathcal{K}$.

As usual in concrete-security definitions, we use the resource parameterized function $\mathrm{Adv}^{\mathsf{xxx}}_H(t,\ell)$ to denote the maximal value of the adversarial advantage (*i.e.* $\mathrm{Adv}^{\mathsf{xxx}}_H(t,\ell) = max_A\{\mathrm{Adv}^{\mathsf{xxx}}_H(A)\}$) over all adversaries A, attacking xxx property of H, that have time complexity at most t and use messages of length at most ℓ bits. We say that H is (t,ℓ,ϵ)-xxx secure if $\mathrm{Adv}^{\mathsf{xxx}}_H(t,\ell) < \epsilon$.

In the sequel, we firstly review the seven properties; namely, Coll, Sec, aSec, eSec, Pre, aPre and ePre, put forth by Rogaway and Shrimpton in [31]. Then, we proceed by providing a *new set of extended properties* for a dedicated-key hash function, which includes enhanced (or strengthened) variants of the security properties considered by Rogaway and Shrimpton in [31].

We remind that the security notions for a dedicated-key hash functions can be either *known-key* properties, or *secret-key* (a.k.a. hidden-key) properties. All the security properties considered in this paper belong to the known-key security setting where at some stage during the attack game, key(s) will be known to the adversary. There are some other applications of dedicated-key hash functions; e.g. as a MAC or PRF primitive, where the key must be kept secret throughout the attack game.

2.1 Previously Considered Seven Security Notions

The advantage measures for an adversary A, attacking any of the seven security properties of a dedicated-key hash function H, are defined (compactly) in Fig. 3.

Note that for some of the notions (namely, Sec[δ], aSec[δ], eSec[δ], Pre[δ], and aPre[δ]) the advantage function is parameterized by a parameter δ where $\{0,1\}^\delta \subseteq \mathcal{M}$. In the case of eSec property the parameter δ is implicit in the definition and assumed to be the length of the first (*i.e.* the target) message M output by the adversary.

2.2 Enhanced Security Notions

We have noticed that the newly emerged notion of "enhanced target collision resistance" (eTCR), put forth by Halevi and Krawczyk in [15], does not belong to the set of the seven properties, and actually eTCR is an strengthened variant of TCR (*i.e.* UOWHF or eSec) property. Considering the definition of eTCR property and its application, we are motivated to study whether it is possible to provide (sensible) enhanced variants for the other properties of the set of the seven security properties in [31], in a similar way that TCR (eSec) is enhanced to eTCR. That is, by giving the adversaries more freedom in selecting a new (second) key and relaxing the corresponding success (winning) conditions in the attack games defining the properties. Interestingly, all properties except 'ePre' are shown to be enhanceable. The definitions and discussions are provided in the sequel.

Definitions. For the six strengthened security notions, the advantage functions of an adversary A attacking H are defined in Fig. 4. For any property xxx $\in$ {s-Coll, s-Sec[δ], s-aSec[δ], s-eSec[δ], s-Pre[δ], s-aPre[δ]}, we

$$
\begin{aligned}
\mathrm{Adv}_H^{Coll}(A) &= \Pr\left[K \xleftarrow{\$} \mathcal{K}; (M, M') \xleftarrow{\$} A(K) : \quad M \neq M' \ \wedge \ H_K(M) = H_K(M')\right] \\
\mathrm{Adv}_H^{Sec[\delta]}(A) &= \Pr\left[\begin{array}{l} K \xleftarrow{\$} \mathcal{K}; M \xleftarrow{\$} \{0,1\}^{\delta}; \\ M' \xleftarrow{\$} A(K, M) \qquad : \quad M \neq M' \ \wedge \ H_K(M) = H_K(M') \end{array}\right] \\
\mathrm{Adv}_H^{aSec[\delta]}(A) &= \Pr\left[\begin{array}{l} (K, State) \xleftarrow{\$} A_1(); \\ M \xleftarrow{\$} \{0,1\}^{\delta}; \\ M' \xleftarrow{\$} A_2(M, State) \quad : \quad M \neq M' \ \wedge \ H_K(M) = H_K(M') \end{array}\right] \\
\mathrm{Adv}_H^{eSec[\delta]}(A) &= \Pr\left[\begin{array}{l} (M, State) \xleftarrow{\$} A_1(); \\ K \xleftarrow{\$} \mathcal{K}; \\ M' \xleftarrow{\$} A_2(K, State) \quad : \quad M \neq M' \ \wedge \ H_K(M) = H_K(M') \end{array}\right] \\
\mathrm{Adv}_H^{Pre[\delta]}(A) &= \Pr\left[\begin{array}{l} K \xleftarrow{\$} \mathcal{K}; M \xleftarrow{\$} \{0,1\}^{\delta}; Y \leftarrow H_K(M); \\ M' \xleftarrow{\$} A(K, Y) \qquad : \quad H_K(M') = Y \end{array}\right] \\
\mathrm{Adv}_H^{aPre[\delta]}(A) &= \Pr\left[\begin{array}{l} (K, State) \xleftarrow{\$} A_1(); \\ M \xleftarrow{\$} \{0,1\}^{\delta}; Y \leftarrow H_K(M); \\ M' \xleftarrow{\$} A_2(Y, State) \quad : \quad H_K(M') = Y \end{array}\right] \\
\mathrm{Adv}_H^{ePre}(A) &= \Pr\left[(Y, State) \xleftarrow{\$} A_1(); K \xleftarrow{\$} \mathcal{K}; M' \xleftarrow{\$} A_2(K, State) : \quad H_K(M') = Y\right]
\end{aligned}
$$

Fig. 3. Definitions of the seven security notions for a dedicated-key hash function [31].

say that H is (t, ℓ, ϵ)-xxx if $\mathrm{Adv}_H^{\mathsf{xxx}}(t, \ell) < \epsilon$. Note that some of the notions (namely, s-Sec[δ], s-aSec[δ], s-eSec[δ], s-Pre[δ] and s-aPre[δ]) are parameterized by δ where $\{0,1\}^{\delta} \subseteq \mathcal{M}$. In the case of s-eSec (*i.e.* eTCR) property the parameter δ is implicit in the definition and assumed to be the length of the first (*i.e.* target) message M output by the adversary. If H is a compression function (*i.e.* an FIL hash function), then parameters δ and ℓ will be the same as the (fixed) input length of the compression function and hence are omitted from the notations.

The Case for ePre. Unlike the other six properties, ePre notion of security cannot be strengthened by allowing the adversary to select a new key K' in the second phase of its attack, as used to define new enhanced variants in Fig. 4. This is because there will remain no random challenge to be given to the adversary in such a game and hence a trivial adversary will always exist. To make this clear, let's try to strengthen the ePre property in the same way that was done for other properties in Fig. 4. Doing so, one gets the following advantage measure:

$$\Pr\left[(Y, State) \xleftarrow{\$} A_1(); K \xleftarrow{\$} \mathcal{K}; (K', M') \xleftarrow{\$} A_2(K, State) : H_{K'}(M') = Y\right]$$

Clearly, as the winning condition (*i.e.* $H_{K'}(M') = Y$) does not involve the only random challenge (*i.e.* K) in the attack game, a trivial adversary, which selects arbitrary K' and M'; computes $H_{K'}(M') = Y$, and outputs Y and (K', M'), always wins this game with probability one.

Remark 1. We notice that the parametrization of some of the security properties by δ is mainly aimed to handle some subtle technical issues as follows:

– Efficient sampling from a set of messages according to the uniform distribution requires the set to be finite. For an arbitrary-input-length hash function, with $\mathcal{M} = \{0,1\}^*$, the message space is infinite, and hence cannot be sampled uniformly at random. Clearly, if H is an FIL hash function (*i.e.* a compression

$$\mathrm{Adv}_H^{\text{s-Coll}}(A) = \Pr\left[K \xleftarrow{\$} \mathcal{K}; (M, M', K') \xleftarrow{\$} A(K) : (K, M) \neq (K', M') \wedge H_K(M) = H_{K'}(M')\right]$$

$$\mathrm{Adv}_H^{\text{s-Sec}[\delta]}(A) = \Pr\left[\begin{array}{l} K \xleftarrow{\$} \mathcal{K}; M \xleftarrow{\$} \{0,1\}^{\delta}; \\ K', M' \xleftarrow{\$} A(K, M) \quad : (K, M) \neq (K', M') \wedge H_K(M) = H_{K'}(M') \end{array}\right]$$

$$\mathrm{Adv}_H^{\text{s-aSec}[\delta]}(A) = \Pr\left[\begin{array}{l} (K, State) \xleftarrow{\$} A_1(); \\ M \xleftarrow{\$} \{0,1\}^{\delta}; \\ K', M' \xleftarrow{\$} A_2(M, State) \quad : (K, M) \neq (K', M') \wedge H_K(M) = H_{K'}(M') \end{array}\right]$$

$$\mathrm{Adv}_H^{\text{s-eSec}[\delta]}(A) = \Pr\left[\begin{array}{l} (M, State) \xleftarrow{\$} A_1(); \\ K \xleftarrow{\$} \mathcal{K}; \\ K', M' \xleftarrow{\$} A_2(K, State) \quad : (K, M) \neq (K', M') \wedge H_K(M) = H_{K'}(M') \end{array}\right]$$

$$\mathrm{Adv}_H^{\text{s-Pre}[\delta]}(A) = \Pr\left[\begin{array}{l} K \xleftarrow{\$} \mathcal{K}; M \xleftarrow{\$} \{0,1\}^{\delta}; Y \leftarrow H_K(M); \\ K', M' \xleftarrow{\$} A(K, Y) \quad : \quad H_{K'}(M') = Y \end{array}\right]$$

$$\mathrm{Adv}_H^{\text{s-aPre}[\delta]}(A) = \Pr\left[\begin{array}{l} (K, State) \xleftarrow{\$} A_1(); \\ M \xleftarrow{\$} \{0,1\}^{\delta}; Y \leftarrow H_K(M); \\ K', M' \xleftarrow{\$} A_2(Y, State) \quad : \quad H_{K'}(M') = Y \end{array}\right]$$

Fig. 4. Definitions of enhanced properties for a dedicated-key hash function.

function), with $\mathcal{M} = \{0,1\}^m$, then parameter δ (and also the resource parameter ℓ) will be the same as the fixed input length of the compression function (*i.e.* $\delta = \ell = m$), and hence can be omitted.

– The ideal security level, measured in terms of time complexity of attacks, for (variants of) second preimage resistance and preimage resistance properties considering a hash function $H : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$ is 2^n, due to a simple generic (random search) attack. Clearly, if the length of target message strings is "too short" (e.g. $\delta < n$), then one will be able to simply search the input message space in less than 2^n steps. On the other hand, for iterated hash functions if the length of a target input message is "too long"; e.g. $\delta = 2^l$ blocks for some large l, then there are generic long message second preimage attacks, put forth by Kelsey and Schneier [16], with reduced time complexity compared to the ideal 2^n level for too long target messages, e.g. when $l = n/2$. Therefore, explicitly parameterizing the properties by the length of the target messages, *i.e.* δ, can clearly show these dependencies of the advantage functions on the target message length.

3 Relationships among the Security Notions

In this section, we provide the details of the "*new*" relationships (implications and separations) between any two properties among the thirteen security properties as defined in Fig. 3 and Fig. 4. Noticing that the relationships among the seven security properties in Fig. 3 were shown in [31, 32], and the relationships between eTCR (s-eSec) and other properties were demonstrated in [27, 28, 33], we complete all the remaining new relationships. The summary of our results is shown in Fig. 2, where we use the conventions explained in the sequel to represent the relationships.

3.1 Security Preserving Implications

A solid directed line from a security notion xxx to a security notion yyy (*i.e.* xxx $\to$ yyy) is used to represent a security-preserving reduction from xxx to yyy. All security-preserving implications *in this paper* are easily

provable by a tight concrete bound of the form $\mathrm{Adv}^{\mathsf{yyy}}_H(t') \leq \mathrm{Adv}^{\mathsf{xxx}}_H(t)$, where $t' = t - c$ for some small constant c. That is, for any hash function $H : \mathcal{K} \times \mathcal{M} \rightarrow \{0,1\}^n$, if H is secure in the xxx sense then it is also secure in the yyy sense.

Lemma 1. *For any dedicated-key hash function $H : \mathcal{K} \times \mathcal{M} \rightarrow \{0,1\}^n$, and for any fixed value of δ such that $\{0,1\}^\delta \subseteq \mathcal{M}$, let* xxx *be any property $\in$ {Coll, Sec[δ], aSec[δ], eSec[δ], Pre[δ], aPre[δ] }; we have s-*xxx *$\rightarrow$* xxx*.*

Proof. It is straightforward to see that $\mathrm{Adv}^{\mathsf{xxx}}_H(t) \leq \mathrm{Adv}^{s-\mathsf{xxx}}_H(t)$, just by considering the definitions of the security notions and their "*strengthened*" variants in Fig. 3 and Fig. 4, respectively. Note that in defining game for a strengthened notion s-xxx, the adversary is given more power by being allowed to choose a different key K', and hence any adversary A that can succeed in playing xxx game (where it does not get to choose a second key) will clearly succeed in the game defining s-xxx (where it gets to choose a second key at will). □

The following implications are also straightforward to show by simple security-preserving reductions.

Theorem 1. *For any dedicated-key hash function $H : \mathcal{K} \times \mathcal{M} \rightarrow \{0,1\}^n$, and for any fixed value of δ such that $\{0,1\}^\delta \subseteq \mathcal{M}$, we have:*

1. *s-Coll $\rightarrow$ s-Sec[δ]*
2. *s-Coll $\rightarrow$ s-eSec[δ]*
3. *s-Coll $\rightarrow$ Sec[δ]*
4. *s-Coll $\rightarrow$ eSec[δ]*
5. *s-aSec[δ] $\rightarrow$ s-Sec[δ]*
6. *s-aSec[δ] $\rightarrow$ Sec[δ]*
7. *s-eSec[δ] $\rightarrow$ s-Sec[δ]*
8. *s-eSec[δ] $\rightarrow$ Sec[δ]*
9. *s-aPre[δ] $\rightarrow$ s-Pre[δ]*
10. *s-aPre[δ] $\rightarrow$ Pre[δ]*

Proof. We prove the case for s-Coll $\rightarrow$ s-Sec[δ]. Let A be an adversary that can break s-Sec[δ] property of H with advantage ϵ and using time t. It is easy to construct an adversary B against s-Coll property of H with the same advantage, as follows:

Algorithm $B(K)$
$M \xleftarrow{\$} \{0,1\}^\delta$
$(K', M') \xleftarrow{\$} A(K, M)$;
return (M, M', K')

When A wins in its s-Sec[δ] attack; *i.e.* $(K, M) \neq (K', M')$ and $H_K(M) = H_{K'}(M')$, clearly A also wins in its s-Coll attack. Hence $\mathrm{Adv}^{s-Coll}_H(B) = \mathrm{Adv}^{s-Sec[\delta]}_H(A)$, and the time complexity of B is that of A plus a small constant time.

Other reductions showing the remaining security preserving implications are also quite straightforward and omitted here. □

3.2 Provisional Implications

A provisional implication from a security notion xxx to a security notion yyy; denoted by xxx ⇢ yyy, means that there is a reduction from xxx to yyy, but the reduction is not security-preserving. That is, we can

upper-bound $\mathrm{Adv}_H^{\mathsf{yyy}}(t')$ as a function of $\mathrm{Adv}_H^{\mathsf{xxx}}(t)$, but the inherited security guarantee for the notion yyy using such a bound is provisioned on the exact security degradation characteristics of the reduction, which usually depends on the hash function parameters, e.g. the lengths of input (δ) and output (n). Therefore, these provisional implications should be interpreted carefully, as for some values of the parameters (e.g. when there is little or no compression, *i.e.* $\delta \approx n$) these reductions may effectively vanish.

Theorem 2. *For any dedicated-key hash function $H : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$, and for any fixed value of parameter δ such that $\{0,1\}^\delta \subseteq \mathcal{M}$, we have:*

1. ***s-Coll*** ⇢ ***ePre***: $Adv_H^{ePre}(t') \leq \sqrt{Adv_H^{s-Coll}(t)} + 2^{-k}$
2. *s-Coll* ⇢ *s-Pre*$[\delta]$: $Adv_H^{s-Pre[\delta]}(t') \leq 3\ Adv_H^{s-Coll}(t) + 2^{n-\delta}$
3. *s-Coll* ⇢ *Pre*$[\delta]$: $Adv_H^{Pre[\delta]}(t') \leq 3\ Adv_H^{s-Coll}(t) + 2^{n-\delta}$
4. ***s-Sec***$[\delta]$ ⇢ ***s-Pre***$[\delta]$: $Adv_H^{s-Pre[\delta]}(t') \leq 3\ Adv_H^{s-Sec[\delta]}(t) + 2^{n-\delta}$
5. *s-Sec*$[\delta]$ ⇢ *Pre*$[\delta]$: $Adv_H^{Pre[\delta]}(t') \leq 3\ Adv_H^{s-Sec[\delta]}(t) + 2^{n-\delta}$
6. *s-aSec*$[\delta]$ ⇢ *s-Pre*$[\delta]$: $Adv_H^{s-Pre[\delta]}(t') \leq 3\ Adv_H^{s-aSec[\delta]}(t) + 2^{n-\delta}$
7. *s-aSec*$[\delta]$ ⇢ *Pre*$[\delta]$: $Adv_H^{Pre[\delta]}(t') \leq 3\ Adv_H^{s-aSec[\delta]}(t) + 2^{n-\delta}$
8. *s-aSec*$[\delta]$ ⇢ *aPre*$[\delta]$: $Adv_H^{aPre[\delta]}(t') \leq 3\ Adv_H^{s-aSec[\delta]}(t) + 2^{n-\delta}$
9. ***s-aSec***$[\delta]$ ⇢ ***s-aPre***$[\delta]$: $Adv_H^{s-aPre[\delta]}(t') \leq 3\ Adv_H^{s-aSec[\delta]}(t) + 2^{n-\delta}$
10. *s-eSec*$[\delta]$ ⇢ *s-Pre*$[\delta]$: $Adv_H^{s-Pre[\delta]}(t') \leq 3\ Adv_H^{s-eSec[\delta]}(t) + 2^{n-\delta}$
11. *s-eSec*$[\delta]$ ⇢ *Pre*$[\delta]$: $Adv_H^{Pre[\delta]}(t') \leq 3\ Adv_H^{s-eSec[\delta]}(t) + 2^{n-\delta}$

where $t' = t - cT_{H,\delta}$, for some small non-negative constant c, and $T_{H,\delta}$ denotes the time for one computation of H.

Proof. We prove the first, fourth, and ninth cases, *i.e.* 's-Coll ⇢ ePre', 's-Sec$[\delta]$ ⇢ s-Pre$[\delta]$', and 's-aSec$[\delta]$ ⇢ s-aPre$[\delta]$'. Note that these are the three *new* essential provisional implications as depicted in Fig. 2. (We remind that the other two essential provisional implications, namely 'Sec ⇢ Pre' and 'aSec ⇢ aPre', in Fig. 2, are already known from [32].) All other provisional implications can be straightforwardly obtained by combining these essential provisional implications with the security-preserving implications.

Proof of 's-Coll ⇢ ePre'. We employ the Reset Lemma from [2] for our purpose. The first and main step is to express our problem in a format which can be considered as a special case of the Reset Lemma, and then we can apply the probabilistic analysis of the Reset Lemma. To simplify the representation of our proof, we denote an adversary as a single probabilistic algorithm A which uses a *State* variable to keep track of its several attack steps, rather than viewing A as consisting of two sub-algorithms $A = (A_1, A_2)$.

Let $\mathrm{Verify}(M, K, Y)$ be a deterministic predicate to compute a boolean decision as follows:

$$\mathrm{Verify}(M,K,Y) = \begin{cases} 1 & \text{if } H_K(M) = Y \\ 0 & \text{otherwise} \end{cases}$$

Let $R \in \{0,1\}^r$ denote the random tape (*i.e.* coins) used by the probabilistic algorithm A. Using the above predicate, we can rewrite the experiment defining the ePre attack by A against H (in a format which is appropriate for our analysis) as below; where $\varnothing$ means an empty string:

ePre Experiment
$R \xleftarrow{\$} \{0,1\}^r$; $(Y, State) = A(\varnothing; R)$;
$K \xleftarrow{\$} \{0,1\}^k$; $M = A(K, State; R)$; $d = \mathrm{Verify}(M, K, Y)$;
Return d

Clearly probability that the above 'ePre Experiment' returns 1 is equal to $\mathrm{Adv}_H^{ePre}(A)$. Now consider the following Reset Experiment:

Reset Experiment:
$R \xleftarrow{\$} \{0,1\}^r$; $(Y, State) = A(\varnothing; R)$;
$K1 \xleftarrow{\$} \{0,1\}^k$; $M1 = A(K1, State; R)$; $d_1 = \mathrm{Verify}(M1, K1, Y)$;
$K2 \xleftarrow{\$} \{0,1\}^k$; $M2 = A(K2, State; R)$; $d_2 = \mathrm{Verify}(M2, K2, Y)$;
If $(d_1 = 1 \ \wedge \ d_2 = 1 \ \wedge \ K1 \neq K2)$ then **return 1** else **return 0**

The proof of the following proposition can be deduced as a special case of that of the Reset Lemma in [2]. We provide a proof of this probabilistic claim here for completeness, as it will also surface in several other cases in the following.

Proposition 1. *Let p denote the probability that the ePre Experiment returns 1 (i.e. $p = Adv_H^{ePre}(A)$), and q be the probability that the Reset Experiment returns 1; we have $p \leq \sqrt{q} + 2^{-k}$.*

Proof (Proof of the Proposition). For any $R \in \{0,1\}^r$, let Y_R and M_R denote the target hash value and the message, output by ePre adversary A having a random tape R. Define two functions $X : \{0,1\}^r \rightarrow [0,1]$ and $Y : \{0,1\}^r \rightarrow [0,1]$ as follows:

$$X(R) \triangleq \Pr[\mathrm{Verify}(M_R, K, Y_R) = 1] \tag{1}$$

where the probability is taken over random selection of K from the key space $\{0,1\}^k$, and

$$Y(R) \triangleq \Pr[\mathrm{Verify}(M1_R, K1, Y_R) = 1 \wedge \mathrm{Verify}(M2_R, K2, Y_R) = 1 \wedge K1 \neq K2] \tag{2}$$

where the probability is taken over random and *independent* selection of $K1$ and $K2$ from the key space $\{0,1\}^k$. By a simple argument, noting that $K1$ and $K2$ are chosen independently and using the fact that $\Pr(E \ \wedge \ \overline{F}) \geq \Pr(E) - \Pr(F)$ for any two events E and F, we have:

$$\begin{aligned} Y(R) &= \Pr[\mathrm{Verify}(M1_R, K1, Y_R) = 1] \,.\, \Pr[\mathrm{Verify}(M2_R, K2, Y_R) = 1 \wedge K1 \neq K2] \\ &\geq X(R)[X(R) - 2^{-k}] \end{aligned} \tag{3}$$

We can view functions X and Y as random variables over sample space $\{0,1\}^r$ of random tape used by probabilistic algorithm A. Now, note that the probabilities that the 'ePre Experiment' and the 'Reset Experiment' return 1 are, respectively, the expected values of the random variables X and Y with respect to R, i.e. $p = \mathbf{E}[X]$ and $q = \mathbf{E}[Y]$. Using the inequality (3) and letting $c = 2^{-k}$ we have:

$$q = \mathbf{E}[Y] \geq \mathbf{E}[X(X - c)] = \mathbf{E}[X^2] - c\mathbf{E}[X] \geq \mathbf{E}[X]^2 - c\mathbf{E}[X] = p^2 - cp$$

Using the above relation we have:

$$(p - \frac{c}{2})^2 = p^2 - cp + \frac{c^2}{4} \leq q + \frac{c^2}{4}$$

and using the fact that $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ for $a, b \geq 0$ we have:

$$p - \frac{c}{2} \leq \sqrt{q} + \frac{c}{2}$$

that is, (remembering $c = 2^{-k}$) we get the final result as $p \leq \sqrt{q} + 2^{-k}$.

To complete our proof for 's-Coll ⇢ ePre', we construct an adversary B against s-Coll property of H; such that $\mathrm{Adv}_H^{s-Coll}(B) = q$ as follows. Adversary B, on receiving the first random key K, chooses another random key K' and employs A as shown in the 'Reset Experiment', by putting $K1 = K$ and $K2 = K'$. B returns $(M1, K1)$ and $(M2, K2)$ as colliding pair in its own s-Coll game. Advantage of B in s-Coll game will be the same as the probability that the 'Reset Experiment' returns 1. This can be easily verified by considering the condition that the 'Reset Experiment' returns 1; noticing the defining game of s-Coll property in Fig. 4, and the definition of predicate Verify(., ., .). Note that the Reset Experiment returns 1 if $(\mathrm{Verify}(M1, K1, Y) = 1 \ \wedge \ \mathrm{Verify}(M2, K2, Y) = 1 \ \wedge \ K1 \neq K2)$, and from the definition of Verify(., ., .) this means that $(H_{K1}(M1) = H_{K2}(M2) = Y \ \wedge \ K1 \neq K2)$. Hence, whenever the Reset Experiment returns 1 the pair $(K1, M1) \neq (K2, M2)$ and $H_{K1}(M1) = H_{K2}(M2)$, *i.e.* B succeeds in s-Coll attack against H. This ends the proof of 's-Coll ⇢ ePre'.

Proof of 's-Sec$[\delta]$ **⇢ s-Pre**$[\delta]$**'.** Let A be any adversary that succeeds in s-Pre$[\delta]$ attack against $H : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$, with probability $\epsilon' = \mathrm{Adv}_H^{s-Pre[\delta]}(A)$ and having time complexity at most t'. We can construct the following adversary B against s-Sec$[\delta]$ property of H, which uses A as a subroutine: Adversary B, on receiving (K, M), computes $Y \leftarrow H_K(M)$, runs A on input (K, Y) to get (K', M') and outputs (K', M') as its own output in s-Sec game.

Clearly the time complexity of B is that of A plus the time to compute $Y = H_K(M)$; *i.e.* $t = t' + T_{H,\delta}$. Let $\epsilon = \mathrm{Adv}_H^{s-Sec[\delta]}(B)$. It remains to prove that $\epsilon' \leq 3\epsilon + 2^{n-\delta}$ as stated in the theorem.

Consider the experiment defining the s-Pre$[\delta]$ attack by A against H as follows (see Fig. 4):

$$K \xleftarrow{\$} \{0,1\}^k ; \ M \xleftarrow{\$} \{0,1\}^\delta ; \ Y \leftarrow H_K(M); \ K', M' \xleftarrow{\$} A(K, Y) \tag{4}$$

, where $\epsilon' = \mathrm{Adv}_H^{s-Pre[\delta]}(A)$ is defined as the probability that, after running the above experiment in (4), the success event, $H_{K'}(M') = Y$, happens.

Let $\mathrm{SF}_K(M)$ denote the event that (in the above experiment in (4)), the message M is sibling-free over the input space $\{0,1\}^\delta$ under the function $H_K(.)$; that is, there is no different message $M' \neq M$ such that $M' \in \{0,1\}^\delta$ and $H_K(M') = H_K(M)$. Clearly $\Pr[\mathrm{SF}_K(M)] \leq \frac{2^n}{2^\delta} = 2^{n-\delta}$ because the number of sibling-free messages cannot be more than 2^n, considering the fact that the range of the hash function H has 2^n elements.

Let E1, E2, E3 and E4 be the following events, under the experiment in (4):

- E1: $H_{K'}(M') = Y$ **AND** $\mathrm{SF}_K(M)$

- E2: $H_{K'}(M') = Y$ **AND** $\overline{\mathrm{SF}_K(M)}$ **AND** $K' = K$ **AND** $M' = M$

- E3: $H_{K'}(M') = Y$ **AND** $\overline{\mathrm{SF}_K(M)}$ **AND** $K' = K$ **AND** $M' \neq M$

- E4: $H_{K'}(M') = Y$ **AND** $\overline{\mathrm{SF}_K(M)}$ **AND** $K' \neq K$

Let $p_1 = \Pr[\mathrm{E1}]$, $p_2 = \Pr[\mathrm{E2}]$, $p_3 = \Pr[\mathrm{E3}]$, and $p_4 = \Pr[\mathrm{E4}]$; where the probabilities are under the same experiment as defined in (4). Clearly E1, E2, E3, and E4 are disjoint events, and their union is the event that A succeeds in the s-Pre attack against H; *i.e.* $\epsilon' = \Pr[H_{K'}(M') = Y] = p_1 + p_2 + p_3 + p_4$. We can bound p_1, p_2, p_3, and p_3 as follows:

- $p_1 = \Pr[H_{K'}(M') = Y \ \wedge \ \mathrm{SF}_K(M)] \leq \Pr[\mathrm{SF}_K(M)] \leq 2^{n-\delta}$.

- It is easy to verify that $p_3 \leq \epsilon$ and $p_4 \leq \epsilon$ as follows. Note that adversary B computes $Y = H_K(M)$, and if E3 happens, we have $H_{K'}(M') = Y \ \wedge \ (K', M') \neq (K, M)$ (as $M' \neq M$), and so B succeeds in s-Sec attack. Therefore, $p_3 = \Pr[\mathrm{E3}] \leq \mathrm{Adv}_H^{s-Sec[\delta]}(B) = \epsilon$. Similarly, if E4 happens we

have $H_{K'}(M') = Y \wedge (K', M') \neq (K, M)$ (as $K' \neq K$), and B succeeds in s-Sec attack. Hence, $p_4 = \Pr[\text{E4}] \leq \text{Adv}_H^{s-Sec[\delta]}(B) = \epsilon$.

– It remains to bound p_2. We claim that $\Pr[\text{E2}] \leq \Pr[\text{E3}] \leq \epsilon$. Let $H_K^{-1}(Y, \delta) \triangleq \left\{ M \in \{0,1\}^\delta \,|H_K(M) = Y \right\}$ and $H_K^{-1}(Y) \triangleq \{M \in \mathcal{M} | H_K(M) = Y\}$. The event $\overline{\text{SF}_K(M)}$ means that the message M is not sibling-free over $\{0,1\}^\delta$; that is, $|H_K^{-1}(Y, \delta)| \geq 2$. (Clearly if M is *not* sibling-free over $\{0,1\}^\delta \subseteq \mathcal{M}$, then it will not be sibling-free over $\mathcal{M}$ either; as $|H_K^{-1}(Y)| \geq |H_K^{-1}(Y, \delta)| \geq 2$.) Now, referring to the s-Pre attack experiment by A in (4), it can be seen that adversary A is *not* given any information about preimage M except its hash value $Y = H_K(M)$. Noting that the message M is selected uniformly at random from $\{0,1\}^\delta$ and $|H_K^{-1}(Y, \delta)| \geq 2$, the probability that A outputs $M' \in H_K^{-1}(Y, \delta)$ such that $M' = M$ is at most the probability that A outputs any arbitrary preimage in $H_K^{-1}(Y) - \{M\}$.

This ends the proof of the case for 's-Sec[δ] $\dashrightarrow$ s-Pre[δ]'.

Proof of 's-aSec[δ] $\dashrightarrow$ s-aPre[δ]'. The proof of this case is very similar to the previous case with some minor differences in the reduction. Let $A = (A_1, A_2)$ be any adversary that succeeds in s-aPre[δ] attack against $H : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$, with probability $\epsilon' = \text{Adv}_H^{s-Pre[\delta]}(A)$ and having time complexity at most t'. We construct an adversary $B = (B_1, B_2)$ against s-aSec[δ] property of H, which uses A as a subroutine as follows: B_1 just runs A_1 and forwards whatever it outputs (*i.e.* $(K, State)$) as its own output. B_2 on receiving $(M, State)$, computes $Y \leftarrow H_K(M)$, runs A_2 on input $(Y, State)$ to get (K', M') and forwards this as its own output in the s-aSec attack.

The time complexity of B is that of A plus the time to compute $Y = H_K(M)$; *i.e.* $t = t' + T_{H,\delta}$. Let $\epsilon = \text{Adv}_H^{s-Sec[\delta]}(B)$. It remains to prove that $\epsilon' \leq 3\epsilon + 2^{n-\delta}$. The argument is quite similar to that of the previous case, *i.e.* 's-Sec[δ] $\dashrightarrow$ s-Pre[δ]', and hence omitted here. (The only difference is that the s-Pre attack experiment in (4) should be replaced by the s-aPre attack experiment.) □

3.3 Separations

We use $\mathsf{xxx} \nrightarrow \mathsf{yyy}$ to show that notion xxx does not imply notion yyy. These separation results are shown by providing counterexamples. Namely, assuming that there exists a dedicated-key hash function $H : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^n$ that is $(t, \epsilon) - \mathsf{xxx}$ secure, we construct (as a counterexample) a dedicated-key hash function $G : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^n$ which is $(t', \epsilon') - \mathsf{xxx}$ secure, but completely *insecure* in yyy sense; *i.e.* $\text{Adv}_G^{\mathsf{yyy}}(c) \approx 1$, where c is a small constant. The concrete relations between adversarial advantages (*i.e.* $\epsilon = \text{Adv}_H^{\mathsf{xxx}}(t)$ and $\epsilon' = \text{Adv}_G^{\mathsf{xxx}}(t')$) and the resource parameters (t and t') are given explicitly for each case.

The following simple lemma will be quite useful in stating the separation results compactly.

Lemma 2. *Let* xxx, yyy, *and* zzz *be any three security properties defined for a hash function* $H : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$. *If* $\mathsf{zzz} \to \mathsf{yyy}$, *then from* $\mathsf{xxx} \nrightarrow \mathsf{yyy}$ *we can conclude that* $\mathsf{xxx} \nrightarrow \mathsf{zzz}$.

Proof. Note that $\mathsf{zzz} \to \mathsf{yyy}$ (in this paper) means that $\text{Adv}_H^{\mathsf{yyy}}(t') \leq \text{Adv}_H^{\mathsf{zzz}}(t)$, where $t' = t - c$ for a small constant c. Hence, if one constructs a counterexample hash function $G : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^n$ that has the property xxx, but is insecure in the yyy sense (*i.e.* $\text{Adv}_G^{\mathsf{yyy}}(c) \approx 1$, for a small constant c), then clearly $\text{Adv}_G^{\mathsf{zzz}}(c') \approx 1$ for a small constant c'; that is, G will also be insecure in the zzz sense.

Remark 2. We should mention that for extreme ranges of the parameter values, when the provisional implications vanish (e.g. when there is no compression; $\delta = n$), Rogaway and Shrimpton [31] considered the possibility of showing some "unconditional separation" results, but as they stated in [31]: "That unconditional separations are (sometimes) possible in this domain is a consequence of the fact that, for some values of the domain and range, secure hash functions trivially exist (e.g. the identity function $H_K(M) = M$ is

collision-free [but not preimage resistant]). " In this paper, we do not consider such unconditional separations and instead we emphasize that provisional implications must be interpreted carefully according to the exact bounds shown by related reductions.

Figure 5 lists the counterexamples that we use to prove the separation results. Construction of some of these counterexamples are inspired from those of [31, 28, 27], where they were utilized to show other separation results.

$$G1_K(M) = \begin{cases} C^* & \text{if } K = K^* \\ H_K(M) & \text{otherwise} \end{cases}$$

$$G2_K(M) = \begin{cases} K_{1\ldots n} & \text{if } val(M) = val(K) \\ H_K(M) & \text{otherwise} \end{cases}$$

$$G3_K(M) = \begin{cases} H_K(0^{m-k}||K) & \text{if } M = 1^{m-k}||K \\ H_K(M) & \text{otherwise} \end{cases}$$

$$G4_K(M) = \begin{cases} C^* & \text{if } M = 0^m \vee M = 1^m \\ H_K(M) & \text{otherwise} \end{cases}$$

$$G5_K(M) = H_K(M_{1\cdots m-1}||0)$$

$$G6_K(M) = \begin{cases} C^* & \text{if } M = M^* \\ H_K(M) & \text{otherwise} \end{cases}$$

$$G7_K(M) = \begin{cases} K_{1\ldots n} & \text{if } val(M) = val(K) & (1) \\ H_K(\langle val(K) \rangle_m) & \text{if } val(M) \neq val(K) \wedge H_K(M) = K_{1\ldots n} & (2) \\ H_K(M) & \text{otherwise} & (3) \end{cases}$$

Fig. 5. Construction of counterexample hash functions $Gi : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^n$, for $1 \leq i \leq 7$, from a given hash function $H : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^n$. All counterexamples have the same key space, domain and range as the underlying hash function H. For the cases of $G2, G3$ and $G7$, it is assumed that $m > k \geq n$. The parameters $K^* \in \{0,1\}^k$; $M^* \in \{0,1\}^m$; and $C^* \in \{0,1\}^n$ have *arbitrary* and *fixed* values; e.g. $K^* = 0^k, M^* = 0^m, C^* = 0^n$. The function $val(.)$ on input a binary string $S = S_1 \cdots S_s$, considered as an unsigned binary number with S_1 as the most significant bit, returns its decimal value.

Referring to (the three tables in) Fig. 2, it can be seen that there are 87 separations among the properties, of which 11 separations are already known from [28, 27]. In the sequel, we complete the study of all the remaining 76 new separations. The proofs are organized as follows:

- Theorem 3 (showing 2 separations) and Theorem 4 (showing 22 separations) together with Lemma 2 and the security-preserving implications (see Fig. 2) provide details of the 41 new separations shown in the top two tables in Fig. 2.
- Theorem 5 (showing 7 separations) and Theorem 6 (showing 7 separations) together with Lemma 2 provide the remaining 35 new separations shown in the bottom table in Fig. 2.

Theorem 3. *s-Coll $\nrightarrow$ aSec and s-Coll $\nrightarrow$ aPre*

Proof. We use counterexample $G1$, defined in Fig. 5, to prove these separations. Let's first demonstrate that $G1$ is completely insecure in both the aSec sense and the aPre sense.

- $\mathrm{Adv}^{aSec}_{G1}(c') = 1$: Consider the following simple adversary $A = (A_1, A_2)$ playing aSec game against $G1$. A_1 chooses the key as $K = K^*$, and A_2 after receiving the first randomly selected message M,

outputs any different message $M' \neq M$. It can be easily seen that this adversary, spending a small constant c', always wins the aSec game because $M' \neq M$, and by the construction of $G1$ we have $G1_{K^*}(M') = G1_{K^*}(M) = C^*$.

– $\mathrm{Adv}^{aPre}_{G1}(c') = 1$: Consider the following simple adversary $A = (A_1, A_2)$ playing aPre game against $G1$. A_1 chooses the key as $K = K^*$, and A_2 after receiving the hash value $Y = G1_{K^*}(M) = C^*$, outputs any arbitrary message $M' \in \{0,1\}^m$. Adversary $A = (A_1, A_2)$ always wins the aPre game because, according to the construction of $G1$, we have $G1_{K^*}(M') = C^*$ for any $M^* \in \{0,1\}^m$.

To complete the proof, we show that $G1$ inherits the s-Coll property of H by demonstrating that $\mathrm{Adv}^{s-Coll}_{G1}(t') \leq \mathrm{Adv}^{s-Coll}_{H}(t) + \sqrt{\mathrm{Adv}^{s-Coll}_{H}(t)} + 2^{-k+1}$.

Let A be any adversary that can win s-Coll game against $G1$ with success probability $\epsilon' = \mathrm{Adv}^{s-Coll}_{G1}(A)$ and having time complexity at most t'. Consider the following adversary B against s-Coll property of H which uses A as a subroutine (and simply forwards whatever it returns):

Algorithm $B(K)$
10: if $K = K^*$ then $bad \leftarrow true$
20: $(M, M', K') \xleftarrow{\$} A(K)$;
30: if $H_K(M) = C^*$ then $bad \leftarrow true$
40: **return** (M, M', K')

We note that the use of a flag *bad* (whose initial value is assumed to be *false*) in the description of B is only aimed to make the proof easier to follow; otherwise, the lines 10 and 30 in the description of B are dummy and can be omitted from B without affecting its operation.

Let **Bad** be the event that the flag *bad* is set to *true* by B, *i.e.* either $K = K^*$ or $H_K(M) = C^*$. We show that if **Bad** does not happen then B will succeeds in the s-Coll attack against H whenever A succeeds in the s-Coll attack against $G1$.

Note that A succeeds in the s-Coll attack against $G1$ whenever $(M, K) \neq (M', K')$ and $G1_K(M) = G1_{K'}(M')$. Assuming that the event **Bad** does not happen; that is, $K \neq K^* \wedge H_K(M) \neq C^*$, and referring to the construction of $G1$, it can be observed that in this case $G1_K(M) = G1_{K'}(M')$ will imply that $H_K(M) = H_{K'}(M')$; that is, B also succeeds in the s-Coll attack against H. As it is assumed that H is $(t, \epsilon)-$s-Coll, we have: $\epsilon \geq \Pr[B \text{ succeeds}] = \Pr[A \text{ succeeds} \wedge \overline{\mathbf{Bad}}] \geq \Pr[A \text{ succeeds}] - \Pr[\mathbf{Bad}] = \epsilon' - \Pr[\mathbf{Bad}]$. Rearranging the terms we have:

$$\epsilon' \leq \epsilon + \Pr[\mathbf{Bad}] \tag{5}$$

Now we need to upperbound $\Pr[\mathbf{Bad}] = \Pr[K = K^* \vee H_K(M) = C^*]$. Using the union bound we have:

$$\Pr[\mathbf{Bad}] \leq \Pr[K = K^*] + \Pr[H_K(M) = C^*] = 2^{-k} + \Pr[H_K(M) = C^*] \tag{6}$$

It remains to upper-bound $p = \Pr[H_K(M) = C^*]$. We claim that:

Claim. $p = \Pr[H_K(M) = C^*] \leq 2^{-k} + \sqrt{\epsilon}$.

Before continuing to prove this claim, note that the inequalities (5), (6) and the above claim complete the proof of the Theorem 3, *i.e.* we get the target upper-bound as $\epsilon' \leq \epsilon + \sqrt{\epsilon} + 2^{-k+1}$. Clearly, the time complexity of B (denote by t) is that of A (denote by t') plus a small constant time c, i.e. $t = t' + c$.

Proof of the Claim: Let $\mathrm{Verify}(M, K)$ be a deterministic boolean predicate which is defined as follows:

$$\mathrm{Verify}(M, K) = \begin{cases} 1 & \text{if } H_K(M) = C^* \\ 0 & \text{otherwise} \end{cases}$$

According to the description of B, the probability $p = \Pr[H_K(M) = C^*]$ is taken over the random coins used by A and the random selection of the first key K. Let $R \in \{0,1\}^r$ denote the random tape used by A. Referring to the description of B it can be seen that p equals to the probability that the following experiment returns 1:

Experiment I
$R \xleftarrow{\$} \{0,1\}^r$;
$K \xleftarrow{\$} \{0,1\}^k$; $(M, M', K') = A(K; R)$; $d = \text{Verify}(M, K)$;
return d

Let q be the probability that the following reset experiment returns 1:

Experiment II
$R \xleftarrow{\$} \{0,1\}^r$;
$K1 \xleftarrow{\$} \{0,1\}^k$; $(M1, M1', K1') = A(K1; R)$; $d_1 = \text{Verify}(M1, K1)$;
$K2 \xleftarrow{\$} \{0,1\}^k$; $(M2, M2', K2') = A(K2; R)$; $d_2 = \text{Verify}(M2, K2)$;
If $(d_1 = 1 \ \wedge \ d_2 = 1 \ \wedge \ K1 \neq K2)$ then **return 1** else **return 0**

The proof of the following proposition is similar to that of Proposition 1.

Proposition 2. $p \leq \sqrt{q} + 2^{-k}$.

To complete the proof of the Claim, we show that $q \leq \epsilon$. We construct an adversary C against s-Coll property of H, such that $\text{Adv}_H^{s-Coll}(C) = q$, as follows: The adversary C, on receiving a random key $K1$, chooses another random key $K2$, and uses A by reseting it as shown in the Experiment II. C returns $(K2, M1, M2)$ in its s-Coll game. Advantage of C in s-Coll game will be the same as the probability that the Experiment II returns 1. This can be easily verified by considering the condition that the Experiment II returns 1; noticing the defining game of s-Coll property in Fig. 4, and the definition of predicate Verify(., .). Note that Experiment II returns 1 if $\text{Verify}(M1, K1) = 1 \ \wedge \ \text{Verify}(M2, K2) = 1 \ \wedge \ K1 \neq K2$, and from the definition of Verify(., .) this means that $H_{K1}(M1) = H_{K2}(M2) = C^* \ \wedge \ K1 \neq K2$. Hence whenever the Experiment II returns 1, the pair $(K1, M1) \neq (K2, M2)$ and $H_{K1}(M1) = H_{K2}(M2)$, *i.e.* C succeeds in s-Coll attack against H. □

Theorem 4. *Fix the values of the parameters for hash functions as indicated in Fig. 5. The following separations hold (where c and c′ are small constant values and $t' = t - c$):*

1. *s-Sec ↛ Coll:* $Adv_{G3}^{s-Sec}(t') \leq Adv_H^{s-Sec}(t) + 2^{-m+1}$, *and* $Adv_{G3}^{Coll}(c') = 1$.
2. *s-Sec ↛ aSec:* $Adv_{G1}^{s-Sec}(t') \leq Adv_H^{s-Sec}(t) + \sqrt{Adv_H^{s-Sec}(t)} + 2^{-k-m} + 2^{-k}$, *and* $Adv_{G1}^{aSec}(c') = 1$.
3. *s-Sec ↛ aPre:* $Adv_{G1}^{s-Sec}(t') \leq Adv_H^{s-Sec}(t) + \sqrt{Adv_H^{s-Sec}(t)} + 2^{-k-m} + 2^{-k}$, *and* $Adv_{G1}^{aPre}(c') = 1$.
4. *s-Sec ↛ eSec:* $Adv_{G4}^{s-Sec}(t') \leq Adv_H^{s-Sec}(t) + \sqrt{Adv_H^{s-Sec}(t)} + 2^{-k-m} + 2^{-m+1}$, *and* $Adv_{G4}^{eSec}(c') = 1$.
5. *s-Sec ↛ ePre:* $Adv_{G4}^{s-Sec}(t') \leq Adv_H^{s-Sec}(t) + \sqrt{Adv_H^{s-Sec}(t)} + 2^{-k-m} + 2^{-m+1}$, *and* $Adv_{G4}^{ePre}(c') = 1$.
6. *s-aSec ↛ Coll:* $Adv_{G3}^{s-aSec}(t') \leq Adv_H^{s-aSec}(t) + 2^{-m+1}$, *and* $Adv_{G3}^{Coll}(c') = 1$.
7. *s-aSec ↛ eSec:* $Adv_{G4}^{s-aSec}(t') \leq Adv_H^{s-aSec}(t) + \sqrt{Adv_H^{s-aSec}(t)} + 3 \times 2^{-m}$, *and* $Adv_{G4}^{eSec}(c') = 1$.
8. *s-aSec ↛ ePre:* $Adv_{G4}^{s-aSec}(t') \leq Adv_H^{s-aSec}(t) + \sqrt{Adv_H^{s-aSec}(t)} + 3 \times 2^{-m}$, *and* $Adv_{G4}^{ePre}(c') = 1$.

9. *s-eSec* $\nrightarrow$ *s-Coll:* $Adv_{G3}^{s-eSec}(t') \leq Adv_{H}^{s-eSec}(t) + 2^{-k+1}$, *and* $Adv_{G3}^{s-Coll}(c') = 1$.
10. *s-eSec* $\nrightarrow$ *s-aSec:* $Adv_{G1}^{s-eSec}(t') \leq Adv_{H}^{s-eSec}(t) + \sqrt{Adv_{H}^{s-eSec}(t)} + 2^{-k+1}$, *and* $Adv_{G1}^{s-aSec}(c') = 1$.
11. *s-eSec* $\nrightarrow$ *s-aPre:* $Adv_{G1}^{s-eSec}(t') \leq Adv_{H}^{s-eSec}(t) + \sqrt{Adv_{H}^{s-eSec}(t)} + 2^{-k+1}$, *and* $Adv_{G1}^{s-aPre}(c') = 1$.
12. *s-Pre* $\nrightarrow$ *Coll:* $Adv_{G5}^{s-Pre}(t') \leq 2Adv_{H}^{s-Pre}(t)$, *and* $Adv_{G5}^{Coll}(c') = 1$.
13. *s-Pre* $\nrightarrow$ *Sec:* $Adv_{G5}^{s-Pre}(t') \leq 2Adv_{H}^{s-Pre}(t)$, *and* $Adv_{G5}^{Sec}(c') = 1$.
14. *s-Pre* $\nrightarrow$ *aSec:* $Adv_{G5}^{s-Pre}(t') \leq 2Adv_{H}^{s-Pre}(t)$, *and* $Adv_{G5}^{aSec}(c') = 1$.
15. *s-Pre* $\nrightarrow$ *eSec:* $Adv_{G5}^{s-Pre}(t') \leq 2Adv_{H}^{s-Pre}(t)$, *and* $Adv_{G5}^{eSec}(c') = 1$.
16. *s-Pre* $\nrightarrow$ *aPre:* $Adv_{G1}^{s-Pre}(t') \leq Adv_{H}^{s-Pre}(t) + \sqrt{Adv_{H}^{s-Pre}(t)} + 2^{-k}$, *and* $Adv_{G1}^{aPre}(c') = 1$.
17. *s-Pre* $\nrightarrow$ *ePre:* $Adv_{G6}^{s-Pre}(t') \leq Adv_{H}^{s-Pre}(t) + \sqrt{Adv_{H}^{s-Pre}(t)} + 2^{-m}$, *and* $Adv_{G6}^{ePre}(c') = 1$.
18. *s-aPre* $\nrightarrow$ *Coll:* $Adv_{G5}^{s-aPre}(t') \leq 2Adv_{H}^{s-aPre}(t)$, *and* $Adv_{G5}^{Coll}(c') = 1$.
19. *s-aPre* $\nrightarrow$ *Sec:* $Adv_{G5}^{s-aPre}(t') \leq 2Adv_{H}^{s-aPre}(t)$, *and* $Adv_{G5}^{Sec}(c') = 1$.
20. *s-aPre* $\nrightarrow$ *aSec:* $Adv_{G5}^{s-aPre}(t') \leq 2Adv_{H}^{s-aPre}(t)$, *and* $Adv_{G5}^{aSec}(c') = 1$.
21. *s-aPre* $\nrightarrow$ *eSec:* $Adv_{G5}^{s-aPre}(t') \leq 2Adv_{H}^{s-aPre}(t)$, *and* $Adv_{G5}^{eSec}(c') = 1$.
22. *s-aPre* $\nrightarrow$ *ePre:* $Adv_{G6}^{s-aPre}(t') \leq Adv_{H}^{s-aPre}(t) + \sqrt{Adv_{H}^{s-aPre}(t)} + 2^{-m}$, *and* $Adv_{G6}^{ePre}(c') = 1$.

The proof of the cases 2–5, 7–8, 10–11, 16–17, and 22 in this Theorem are quite similar in main parts to that of Theorem 3, where we adapt the Reset Lemma to obtain the square root terms in our upper-bounds. The reductions for the other cases are also straightforward, and hence the proofs are omitted.

Theorem 5. *For any property* xxx $\in \{$*Coll, Sec, aSec, eSec, Pre, aPre, ePre*$\}$*, we have* xxx $\nrightarrow$ *s-Pre.*

The proof is divided into two parts: Lemma 3 shows that xxx $\nrightarrow$ s-Pre, for any xxx $\in \{$Coll, Sec, aSec, eSec$\}$, and Lemma 4 shows that xxx $\nrightarrow$ s-Pre, for any xxx $\in \{$Pre, aPre, ePre$\}$.

Lemma 3. xxx $\nrightarrow$ *s-Pre, for any* xxx $\in \{$*Coll, Sec, aSec, eSec*$\}$

Proof. We use $G7$ as a counterexample. It is easy to verify that $G7$ is completely insecure in s-Pre sense using the following simple adversary A. On receiving the first randomly selected key K and the hash value Y, A outputs (K', M'), where $K' = Y||0^{k-n}$ and $M' = \langle val(K') \rangle_m$. A always wins, as $G7_{K'}(M') = Y$ (noting that $val(M') = val(K')$ and $K'_{1...n} = Y$). So, we have $\mathrm{Adv}_{G7}^{s-Pre}(c') = 1$, where the small constant c' is the time complexity of A.

To complete the proof, it remains to show that $G7$ inherits the xxx property of H, for any xxx $\in$ {Coll, Sec, aSec, eSec}. Here we provide the proof for the case of xxx=Coll property (proof of other cases are quite similar and omitted). This is done by reducing Coll security of $G7$ to that of H. Let A be an adversary that can win Coll game against $G7$ with probability ϵ' using time complexity t'. We construct an adversary B against Coll property of H with the same success probability, *i.e.* $\epsilon = \epsilon'$, and time $t = t' + c$ as stated in the lemma. The construction of B is as follows:

Algorithm $B(K)$
10: $(M, M') \xleftarrow{\$} A(K)$;
20: if $val(M) = val(K) \wedge H_K(M') = K_{1...n}$ then **return** (M, M')
30: if $val(M') = val(K) \wedge H_K(M) = K_{1...n}$ then **return** (M, M')
40: if $val(M) \neq val(K) \wedge H_K(M) = K_{1...n} \wedge$
$val(M') \neq val(K) \wedge H_K(M') \neq K_{1...n}$ then **return** $(\langle val(K) \rangle_m, M')$

50: if $val(M') \neq val(K) \wedge H_K(M') = K_{1...n} \wedge$
$val(M) \neq val(K) \wedge H_K(M) \neq K_{1...n}$ then **return** $(M, \langle val(K) \rangle_m)$
60: **return** (M, M')

We claim that B will return a valid collision for H whenever A returns a valid collision (M, M') for $G7$. To prove this claim first note that B will return a message pair depending on which of the conditions specified in lines 20-60 of its code are satisfied. Referring to the definition of counterexample hash function $G7$, if A returns a valid collision (M, M') under $G7_K$, we can analyze all possible cases that this can happen and show that in each case algorithm B also returns a collision for H_K. Let **(i)-(j) Coll** mean that the colliding messages M and M' output by A for $G7_K$, respectively, satisfy conditions in line (i) and line (j) in definition of the function $G7$. Then we have the following cases:

1. **(1)-(1) Coll**, **(1)-(3) Coll** and **(3)-(1) Coll** are impossible. A **(1)-(1) Coll** implies that $M = M'$ which is not possible as it is assumed that (M, M') is a valid collision for $G7_K$. Notice that the condition in line (3) of the definition of $G7$ (implicitly denoted as "*otherwise*") actually can be explicitly shown as:
[if $val(M) \neq val(K) \ \wedge \ H_K(M) \neq K_{1...n}$]; and hence, the hash value computed on line (3) is always different from K and therefore **(1)-(3) Coll** and **(3)-(1) Coll** are not possible for $G7$.
2. **(1)-(2) Coll**: If A outputs a valid **(1)-(2) Coll** for $G7$ then, referring to the definition of $G7$, it can be seen that $val(M) = val(K)$ and $H_K(M') = K_{1...n}$, and from $G7_K(M') = G7_K(M)$ we have that $H_K(\langle val(K) \rangle_m) = K_{1...n}$. In this case, the adversary B returns (M, M') in line 20 of its code as colliding messages for H_K and wins because $H_K(M) = H_K(\langle val(K) \rangle_m) = K_{1...n} = H_K(M')$ (Note that from $val(M) = val(K)$ and $m = |M| > |K| = k$ we get that $M = \langle val(K) \rangle_m$.)
3. **(2)-(1) Coll**: The proof for this case is symmetric to the case of **(1)-(2) Coll** and this time adversary B returns (M, M') in line 30 of its code as collision for H_K.
4. **(2)-(3) Coll**: When A outputs a valid **(2)-(3) Coll** for $G7$ then (by referring to the definition of $G7$, and considering the condition for line (3) of $G7$ explicitly), we have: $val(M) \neq val(K) \wedge H_K(M) = K_{1...n}$, and $val(M') \neq val(K) \wedge H_K(M') \neq K_{1...n}$. Hence, as (M, M') output by A is a valid collision for $G7$, *i.e.* $G7_K(M') = G7_K(M)$, we have $H_K(M') = H_K(\langle val(K) \rangle_m)$ and therefor $(\langle val(K) \rangle_m, M')$ returned by B in line 40, will be a valid collision for H_K.
5. **(3)-(2) Coll**: The proof for this case is symmetric to the case of **(2)-(3) Coll** and this time the adversary B returns $(M, \langle val(K) \rangle_m)$ in line 50 of its code as collision for H_K.
6. **(2)-(2) Coll** and **(3)-(3) Coll**: It can be seen that in these two cases B returns (M, M') as a collision for H_K in line 60 of its code. Referring to the definition of function $G7$, it is seen that, when A outputs a valid collision (M, M') for $G7_K$ as either a **(2)-(2) Coll** or **(3)-(3) Coll** (that is, $M \neq M' \wedge G7_K(M) = G7_K(M')$ and both M and M' belong to the same sub-domain of $G7$) then (M, M') will also be a valid collision for H_K. Note that $G7_K(M) = G7_K(M')$ implies that, in **(2)-(2) Coll** case we have $H_K(M) = H_K(M') = K_{1...n}$, and in **(3)-(3) Coll** case we also have $H_K(M) = H_K(M')$.

Lemma 4. $\mathsf{xxx} \nrightarrow$ *s-Pre, for any* $\mathsf{xxx} \in \{$*Pre, aPre, ePre*$\}$.

Proof. $G2$ is used as a counterexample. We first show that $G2$ is completely insecure in s-Pre sense, using the following simple adversary A. On receiving the first randomly selected key K and the hash value Y, adversary A outputs (K', M'), where $K' = Y||0^{k-n}$ and $M' = \langle val(K') \rangle_m$. A wins the game as $G2_{K'}(M') = Y$ (noting that $val(M') = val(K')$ and $K'_{1...n} = Y$). That is, $\mathrm{Adv}^{s-Pre}_{G2}(c') = 1$, where c' is the (small constant) time complexity of A.

It remains to show that $G2$ inherits the xxx property from H, for any xxx $\in$ {Pre, aPre, ePre}. Here we provide the proof of the case of xxx=Pre (other cases are quite similar and omitted).

Let A be any adversary that breaks Pre property of $G2$, using time complexity t and with advantage ϵ. From the construction of $G2$ it can be seen that, if $val(M) \neq val(K)$ then we have $G2_K(M) = H_K(M)$, and hence the same algorithm A will also succeed in attacking Pre property of H. Let **Bad** denote the event that $val(M) = val(K)$. Clearly $\Pr[\mathbf{Bad}] = 2^{-k}$, where the probability is taken under the experiment defining the Pre property (where K and M are chosen independently at random and $m = |M| > |K| = k$ according to the construction of $G2$). Hence, we have: $\mathrm{Adv}^{Pre}_{H}(A) = \mathrm{Adv}^{Pre}_{G2}(A) - 2^{-k}$. That is, if H is (t, ϵ)-Pre secure then $G2$ is $(t, \epsilon + 2^{-k})$-Pre secure.

Theorem 6. *For any property* xxx $\in$ {*Coll, Sec, aSec, eSec, Pre, aPre, ePre*}*, we have* xxx $\nrightarrow$ *s-Sec.*

The counterexample functions $G7$ and $G2$ are used for the proof. The proof is very similar to that of Theorem 5, and is omitted.

4 Conclusion

We have extended the set of security notions for dedicated-key hash functions by providing new set of enhanced (strengthened) properties, which includes the well-known enhanced target collision resistance property. The latter property has been proven to be useful to enrich the notions of hash functions, in particular with its application to construct the Randomized Hashing mode for strengthening digital signatures. We have also provided a full picture of relationships among the (thirteen) security properties including the (six) enhanced properties and the previously considered seven properties. It is expected that by future research the new enhanced properties introduced in this paper may also find interesting applications in practice. Meanwhile, we note that these new enhanced properties can be considered by cryptanalysts as easier targets for attacking dedicated-key hash functions (e.g. some of the NIST SHA-3 candidates); for example, it might be the case that a hash function H is hard to break in (conventional) Coll sense but it is vulnerable to attacks against the strengthened Coll (*i.e.* s-Coll) property.

References

[1] Bellare, M., Rogaway, P.: Introduction to Modern Cryptography: Chapter 5, Hash Functions. Available at Bellare's homepage via : http://cseweb.ucsd.edu/users/mihir/cse207/index.html (20 September 2009)
[2] Bellare, M., Palacio, A.: GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks. In M. Yung (ed.): CRYPTO 2002. LNCS, vol. 2442, pp. 162–177. Springer (2002)
[3] Bellare, M., Kilian, J., Rogaway, P.: The Security of the Cipher Block Chaining Message Authentication Code. *Journal of Computer and System Sciences*, Vol. 61, No. 3, pp. 362–399, December 2000.
[4] Bellare, M., Rogaway, P.: Collision-Resistant Hashing: Towards Making UOWHFs Practical. In B.S. Kaliski Jr. (ed.): CRYPTO 1997. LNCS, vol. 1294, pp. 470–484. Springer (1997)
[5] Bellare, M., Desai, A., Jokipii, E., Rogawayy, P.: A Concrete Security Treatment of Symmetric Encryption. *Proceedings of the 38th Symposium on Foundations of Computer Science*, IEEE, 1997.
[6] Bellare, M., Canetti, R., Krawczyk, H.: Psuedorandom functions revisited: The cascade construction and its concrete security. Proceedings of the 37th Symposium on Foundations of Computer Science, pp. 514–523, IEEE, 1996.
[7] Bellare, M., Guerin, R., Rogaway, P.: XOR MACs: New Methods for Message Authentication using Finite Pseudorandom Functions. In D. Coppersmith (ed.): CRYPTO 1995. LNCS, vol. 963, pp. 15–28. Springer-Verlag, 1995.
[8] Bellare, M., Kilian, J., Rogaway, P.: The Security of Cipher Block Chaining. In Y. Desmedt (ed.): CRYPTO 1994. LNCS, vol. 839, pp. 341–358. Springer (1994).

[9] Contini, S., Steinfeld, R., Pieprzyk, J., Matusiewicz, K.: A Critical Look at Cryptographic Hash Function Literature. In *ECRYPT Hash Workshop*, 2007.
[10] Damgård, I. B.: Collision Free Hash Functions and Public Key Signature Schemes. In *Advances in Cryptology–EUROCRYPT '87* (1988), Vol. 304 of *LNCS*, Springer-Verlag, 203–216.
[11] Damgård, I. B.: A Design Principle for Hash Functions. In G. Brassard (ed.): CRYPTO 1989. LNCS, vol. 435, pp. 416–427. Springer (1990)
[12] Diffie, W., Hellman, M. E.: New directions in cryptography. *IEEE Trans. on Information Theory*, Vol. IT22, No. 6, 1976, pp. 644–654.
[13] Goldreich, O.: Foundations of Cryptography: Volume 2, Basic Applications. *Cambridge University Press*, 2004.
[14] Goldwasser, S., Bellare, M.: Lecture Notes on Cryptography. Available at Bellare's homepage via: http://cseweb.ucsd.edu/users/mihir/papers/gb.html (20 September 2009)
[15] Halevi, S., Krawczyk, H.: Strengthening Digital Signatures Via Randomized Hashing. In C. Dwork (ed.): CRYPTO 2006. LNCS, vol. 4117, pp. 41–59. Springer (2006)
[16] Kelsey, J., Schneier, B.: Second Preimages on n-Bit Hash Functions for Much Less than 2^n Work. In *Advances in Cryptology–EUROCRYPT '05* (2005), Vol. 3494 of *LNCS*, Springer-Verlag, 474–490.
[17] Menezes, A. J., van Oorschot, P. C., Vanstone, S. A.: *Handbook of Applied Cryptography*. CRC Press, 1996.
[18] Merkle, R. C.: Secrecy, Authentication, and Public Key Systems. UMI Research Press, 1979.
[19] Merkle, R. C. : One Way Hash Functions and DES. In G. Brassard (ed.): CRYPTO 1989. LNCS, vol. 435, pp. 428–446. Springer (1990)
[20] Naor, M., M. Yung: Universal One-Way Hash Functions and Their Cryptographic Applications. In: *Proc. of 21st ACM Symposium on the Theory of Computing* (STOC 1989), pp. 33–43. ACM (1989)
[21] National Institute of Standards and Technology. Cryptographic Hash Algorithm Competition. http://csrc.nist.gov/groups/ST/hash/sha-3/index.html. (20 September 2009)
[22] National Institute of Standards and Technology. NIST SP 800-106: Randomized Hashing for Digital Signatures, February 2009. http://www.csrc.nist.gov/publications/PubsSPs.html#800-106. (20 September 2009)
[23] National Institute of Standards and Technology. FIPS PUB 180-2: Secure Hash Standard. August 2002.
[24] National Institute of Standards and Technology. FIPS PUB 180-3: Secure Hash Standard. June 2007.
[25] Preneel, B.: Analysis and Design of Cryptographic Hash Functions. Doctoral dissertation, K. U. Leuven, 1993.
[26] Rabin, M.O.: Digitalized Signatures. In: R. Lipton, R. DeMillo, Eds. *Foundations of Secure Computation*, Academic Press, New York, 1978, pp. 155–166.
[27] Reyhanitabar, M.R., Susilo, W., Mu, Y.: An Investigation of the Enhanced Target Collision Resistance Property for Hash Functions. Cryptology ePrint Archive, Report 2009/506, 2009.
[28] Reyhanitabar, M.R., Susilo, W., Mu, Y.: Enhanced Target Collision Resistant Hash Functions Revisited. In O. Dunkelman (ed.): FSE 2009. LNCS, vol. 5665, pp. 327–344. Springer (2009)
[29] Rivest, R.: The MD5 Message-Digest Algorithm. RFC 1321, April 1992. Available: http://www.ietf.org/rfc/rfc1321.txt (19 September 2009).
[30] Rogaway, P.: Formalizing Human Ignorance: Collision-Resistant Hashing without the Keys. In P.Q. Nguyen (ed.): VIETCRYPT 2006. LNCS, vol. 4341, pp. 211–228. Springer (2006)
[31] Rogaway, P., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In B.K. Roy, W. Meier (eds.): FSE 2004. LNCS, vol. 3017, pp. 371–388. Springer (2004)
[32] P. Rogaway, T. Shrimpton: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. Cryptology ePrint Archive: Report 2004/035 (Revised version of [31]: 9 Aug 2009).
[33] Yasuda, K.: How to Fill Up Merkle-Damgård Hash Functions. In: Pieprzyk, J. (ed.): ASIACRYPT 2008. LNCS, vol. 5350, pp. 272–289. Springer (2008).
[34] Zheng, Y., Matsumoto, T., Imai, H.: Connections among several versions of one-way hash functions. In *Special Issue on Cryptography and Information Security, Proceedings of IEICE of Japan*, 1990.