Appears in *Fast Software Encryption(FSE 2004)*, Lecture Notes in Computer Science, Vol. ????, Springer-Verlag.
This is the full version.

Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance

P. Rogaway [*] T. Shrimpton [†]

February 10, 2004

Abstract

We consider basic notions of security for cryptographic hash functions: collision resistance, preimage resistance, and second-preimage resistance. We give seven different definitions that correspond to these three underlying ideas, and then we work out all of the implications and separations among these seven definitions within the concrete-security, provable-security framework. Because our results are concrete, we can show two types of implications, *conventional* and *provisional*, where the strength of the latter depends on the amount of compression achieved by the hash function. We also distinguish two types of separations, *conditional* and *unconditional*. When constructing counterexamples for our separations, we are careful to preserve specified hash-function domains and ranges; this rules out some pathological counterexamples and makes the separations more meaningful in practice. Four of our definitions are standard while three appear to be new; some of our relations and separations have appeared, others have not. Here we give a modern treatment that acts to catalog, in one place and with carefully-considered nomenclature, the most basic security notions for cryptographic hash functions.

Key words: collision resistance, cryptographic hash functions, preimage resistance, provable security, second-preimage resistance.

[*] Dept. of Computer Science, University of California, Davis, California 95616, USA; and Dept. of Computer Science, Faculty of Science, Chiang Mai University, 50200 Thailand. E-mail: `rogaway@cs.ucdavis.edu` WWW: `www.cs.ucdavis.edu/~rogaway/`

[†] Dept. of Electrical and Computer Engineering, University of California, Davis, California 95616, USA. E-mail: `teshrim@ucdavis.edu` WWW: `www.ece.ucdavis.edu/~teshrim/`

Contents

1 Introduction **1**

2 Preliminaries **3**

3 Definitions of Hash-Function Security **4**
3.1 Preimage resistance 4
3.2 Second-preimage resistance 6
3.3 Collision resistance 6

4 Equivalent Formalizations with a Two-Stage Adversary **7**

5 Implications **8**

6 Separations **9**

Acknowledgments **11**

References **11**

A Brief History **13**

B Proofs **14**
B.1 Proof of Theorem 7 14
B.2 Proof of Proposition 9 15
B.3 Proof of Proposition 10 16
B.4 Proof of Theorem 11 16
B.5 Proof of Proposition 12 18
B.6 Proof of Theorem 13 18
B.7 Proof of Theorem 14 19

1 Introduction

This paper casts some new light on an old topic: the basic security properties of cryptographic hash functions. We provide definitions for various notions of collision-resistance, preimage resistance, and second-preimage resistance, and then we work out all of the relationships among the definitions. We adopt a concrete-security, provable-security viewpoint, using reductions and definitions as the basic currency of our investigation.

Informal treatments of hash functions. Informal treatments of cryptographic hash functions can lead to a lot of ambiguity, with informal notions that might be formalized in very different ways and claims that might correspondingly be true or false. Consider, for example, the following quotes, taken from our favorite reference on cryptography [9, pp. 323–330]:

> *preimage-resistance* — for essentially all pre-specified outputs, it is computationally infeasible to find any input which hashes to that output, i.e., to find any preimage x' such that $h(x') = y$ when given any y for which a corresponding input is not known.
>
> *2nd-preimage resistance* — it is computationally infeasible to find any second input which has the same output as any specified input, i.e., given x, to find a 2nd-preimage $x' \neq x$ such that $h(x) = h(x')$.
>
> *collision resistance* — it is computationally infeasible to find any two distinct inputs x, x' which hash to the same output, i.e., such that $h(x) = h(x')$.
>
> **Fact** Collision resistance implies 2nd-preimage resistance of hash functions.
>
> **Note** (*collision resistance does not guarantee preimage resistance*)

In trying to formalize and verify such statements, certain aspects of the English are problematic and other aspects aren't. Consider the first statement above. Our community understands quite well how to deal with the term *computationally infeasible*. But how is it meant to specify the output y? (What, exactly, do "essentially all" and "pre-specified outputs" mean?) Is hash function h to be a fixed function or a random element from a set of functions? Similarly, for the second quote, is it really meant that the specified point x can be *any* domain point (e.g., it is not chosen at random)? As for the bottom two claims, we shall see that the first is true under two formalizations we give for 2nd-preimage resistance and false under a third, while the second statement is true only if one insists on allowing the degenerate case of hash functions that do not actually compress.[1]

Scope. In this paper we are going to examine seven different notions of security for a hash function family $H: \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$. For a more complete discussion of nomenclature, see Appendix A and reference [9].

[1] We emphasize that it is most definitely *not* our intent here to criticize one of the most useful books on cryptography; we only use it to help illustrate that there are many ways to go when formalizing notions of hash-function security, and how one chooses to formalize things matters for making even the most basic of claims.

Name	Find	Experiment	Some Aliases
Pre	Find a preimage	random key, random challenge	OWF
ePre	Find a preimage	random key, fixed challenge	
aPre	Find a preimage	fixed key, random challenge	
Sec	Find a second-preimage	random key, random challenge	weak collision resistance
eSec	Find a second-preimage	random key, fixed challenge	UOWHF
aSec	Find a second-preimage	fixed key, random challenge	
Coll	Find a collision	random key (no challenge)	strong collision resistance, collision-free

How did we arrive at exactly these seven notions? We set out to be exhaustive. For *two* of our goals—finding a preimage and finding a second preimage—it makes sense to think of *three* different settings: the key and the challenge being random; the key being random and the challenge being fixed; or the key being fixed and the challenge being random. It makes no sense to think of the key and the challenge as both being fixed, for a trivial adversary would then succeed. For the final goal—finding a collision—there is no challenge and one is compelled to think of the key as being random, for a trivial adversary would prevail if the key were fixed. We thus have $2 \cdot 3 + 1 = 7$ sensible notions, which we name Pre, ePre, aPre, Sec, eSec, aSec, and Coll. The leading "a" in the name of a notion is meant to suggest *always*: if a hash function is secure for any fixed key, then it is "always" secure. The leading "e" in the name of a notion is meant to suggest *everywhere*: if a hash function is secure for any fixed challenge, then it is "everywhere" secure. Notions Coll, Pre, Sec, eSec are standard; variants ePre, aPre, and aSec would seem to be new.

Comments. The aPre and aSec notions may be useful for designing higher-level protocols that employ hash functions that are to be instantiated with SHA1-like objects. Consider a protocol that uses an object like SHA1 but says it is using a collision-resistant hash function, and proves security under such an assumption. There is a problem here, because there is no natural way to think of SHA1 as being a random element drawn from some family of hash functions. If the protocol could instead have used an aSec-secure hash-function family, doing the proof from that assumption, then instantiating with SHA1 would seem to raise no analogous, foundational issues. In short, assuming that your hash function is aSec- or aPre-secure serves to eliminate the mismatch of using a standard cryptographic hash function after having done proofs that depend on using a random element from a hash-function family.

Contributions. Despite the numerous papers that construct, attack, and use cryptographic hash functions, and despite a couple of investigations of cryptographic hash functions whose purpose was close to ours [15, 16], the area seems to have more than its share of conflicting terminology, informal notions, and assertions of implications and separations that are not supported by convincing proofs or counterexamples. Our goal has been to help straighten out some of the basics. See Appendix A for an abbreviated exposition of related work.

We begin by giving formal definitions for our seven notions of hash-function security. Our definitions are concrete (no asymptotics) and treat a hash function H as a family of functions, $H\colon \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$.

After defining the different notions of security we work out all of the relationships among them. Between each pair of notions xxx and yyy we provide either an implication or a separation. Informally, saying that xxx *implies* yyy means that if H is secure in the xxx-sense then it is also secure in the yyy-sense. To separate notions, we say, informally, that xxx *nonimplies* yyy if H can be secure in the xxx-sense without being secure in the yyy-sense.[2] Our implications and separations are quantitative, so we provide both an implication *and* a separation for the cases where this makes sense. Since we are providing implications and separations, we adopt the strongest feasible notions of each, in order to strengthen our results.

We actually give two kinds of implications. We do this because, in some cases, the strength of an implication crucially depends on the amount of compression achieved by the hash function. For these *provisional* implications, if the hash function is substantially compressing (e.g., mapping 256 bits to 128 bits) then the implication is a strong one, but if the hash function compresses little or not at all, then the implication effectively vanishes. It is a matter of interpretation whether such a provisional implication is an implication with a minor "technical" condition, or if a provisional implication is fundamentally not an implication at all. A *conventional* implication is an ordinary one; the strength of the implication does not depend on how much the hash function compresses.

We will also use two kinds of separations, but here the distinction is less dramatic, as both flavors of separations are strong. The difference between a *conventional* separation and an *unconditional* separation lies in whether or not one must effectively assume the existence of an xxx-secure hash function in order to show that xxx nonimplies yyy.

When we give separations, we are careful to impose the hash-function domain and range first; we don't allow these to be chosen so as to make for convenient counterexamples. This makes the problem of constructing counterexamples harder, but it also make the results more meaningful. For example, if a protocol designer wants to know if collision-resistance implies preimage-resistance for a 160-bit hash function H, what good is a counterexample that uses H to make a 161-bit hash function H' that is collision resistant but not preimage-resistant? It would not engender any confidence that collision-resistance fails to imply preimage-resistance when all hash functions of interest have 160-bit outputs.

Some of the counterexamples we use may appear to be unnatural, or to exhibit behavior unlike "real world" hash functions. This is not a concern; our goal is to demonstrate when one notion does not imply another by constructing counterexamples that respect imposed domain and range lengths; there is no need for the examples to look natural.

Our findings are summarized in Figure 1, which shows when one notion implies the other (drawn with a solid arrow), when one notion provisionally implies the other (drawn with a dotted arrow), and when one notion nonimplies the other (we use the absence of an arrow and do not bother to distinguish between the two types of nonimplications). In Figure 2 we give a more detailed summary of the results of this paper.

2 Preliminaries

We write $M \xleftarrow{\$} \mathcal{S}$ for the experiment of choosing a random element from the distribution $\mathcal{S}$ and calling it M. When $\mathcal{S}$ is a finite set it is given the uniform distribution. The concatenation of strings M and M' is denoted by $M \parallel M'$ or MM'. When $M = M_1 \cdots M_m \in \{0,1\}^m$ is an m-bit string and $1 \le a \le b \le m$ we write $M[a..b]$ for $M_a \cdots M_b$. The bitwise complement of a string M is written $\overline{M}$. The empty string is denoted by ε. When a is an integer we write $\langle a \rangle_r$ for the r-bit string that represents a.

A *hash-function family* is a function $H\colon \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ where $\mathcal{K}$ and $\mathcal{Y}$ are finite nonempty sets and $\mathcal{M}$ and $\mathcal{Y}$ are sets of strings. We insist that $\mathcal{Y} = \{0,1\}^n$ for some $n > 0$. The number n is

[2] We say "nonimplies" rather than "does not imply" because a separation is not the negation of an implication; a separation is effectively stronger and more constructive than that.

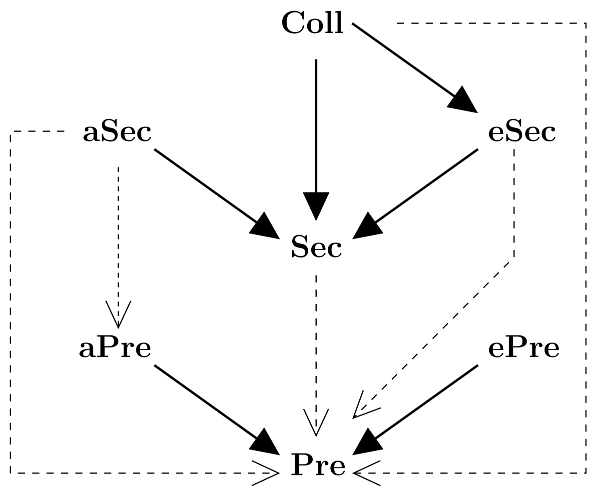

Figure 1: Summary of the relationships among seven notions of hash-function security. Solid arrows represent conventional implications, dotted arrows represent provisional implications (their strength depends on the relative size of the domain and range), and the lack of an arrow represents a separation.

called the *hash length* of H. We also insist that if $M \in \mathcal{M}$ then $\{0,1\}^{|M|} \subseteq \mathcal{M}$ (the assumption is convenient and any reasonable hash function would certainly have this property). Often we will write the first argument to H as a subscript, so that $H_K(M) = H(K, M)$ for all $M \in \mathcal{M}$.

When $H\colon \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ and $\{0,1\}^m \subseteq \mathcal{M}$ we denote by $\mathrm{Time}_{H,m}$ the minimum, over all programs P_H that compute H, of the length of P_H plus the worst-case running time of P_H over all inputs (K, M) where $K \in \mathcal{K}$ and $M \in \{0,1\}^m$; plus the the minimum, over all programs P_K that sample from $\mathcal{K}$, of the time to compute the sample plus the size of P_K. We insist that P_H read its input, so that $\mathrm{Time}_{H,m}$ will always be at least m. Some underlying RAM model of computation must be fixed.

An *adversary* is an algorithm that takes any number of inputs. Some of these inputs may be long strings and so we establish the convention that the adversary can read the ith bit of argument j by writing (i, j), in binary, on distinguished query tape. The resulting bit is returned to the adversary in unit time. If A is an adversary and $\mathbf{Adv}_H^{\mathrm{xxx}}(A)$ is a measure of adversarial advantage already defined then we write $\mathbf{Adv}_H^{\mathrm{xxx}}(\mathcal{R})$ to mean the maximal value of $\mathbf{Adv}_H^{\mathrm{xxx}}(A)$ over all adversaries A that use resources bounded by $\mathcal{R}$. In this paper it is sufficient to consider only the resource t, the running time of the adversary. By convention, the running time is the actual worst case running time of A (relative to some fixed RAM model) plus the description size of A (relative to some fixed encoding of algorithms).

3 Definitions of Hash-Function Security

Here we give formal definitions for seven notions of hash-function security. The definitions fall under the general categories of *preimage-resistance*, *second-preimage resistance*, and *collision-resistance*.

3.1 Preimage resistance

One would like to speak of the difficulty with which an adversary is able to find a preimage for a point in the range of a hash function. Several definitions make sense for this intuition of inverting.

	Pre	ePre	aPre	Sec	eSec	aSec	Coll
Pre	$\to$	$\not\to$ to δ_3 (d)	$\not\to$ to δ_4 (e)	$\not\to$ (h)	$\not\to$ (h)	$\not\to$ (h)	$\not\to$ (h)
ePre	$\to$ (l)	$\to$	$\not\to$ to δ_4 (e)	$\not\to$ (h)	$\not\to$ (h)	$\not\to$ (h)	$\not\to$ (h)
aPre	$\to$ (l)	$\not\to$ to δ_3 (d)	$\to$	$\not\to$ (h)	$\not\to$ (h)	$\not\to$ (h)	$\not\to$ (h)
Sec	$\to$ to δ_1 (a) $\not\to$ to δ_2 (b)	$\not\to$ to δ_3 (d)	$\not\to$ to δ_4 (e)	$\to$	$\not\to$ to δ_5 (i)	$\not\to$ to δ_4 (e)	$\not\to$ to δ_5 (i)
eSec	$\to$ to δ_1 (a) $\not\to$ to δ_2 (c)	$\not\to$ (f)	$\not\to$ to δ_4 (e)	$\to$ (l)	$\to$	$\not\to$ to δ_4 (e)	$\not\to$ to δ_5 (j) $\not\to$ (k)
aSec	$\to$ to δ_1 (a) $\not\to$ to δ_2 (b)	$\not\to$ to δ_3 (d)	$\to$ to δ_1 (a) $\not\to$ to δ_2 (b)	$\to$ (l)	$\not\to$ to δ_5 (i)	$\to$	$\not\to$ to δ_5 (i)
Coll	$\to$ to δ_1 (a)	$\not\to$ (g)	$\not\to$ to δ_4 (e)	$\to$ (l)	$\to$ (l)	$\not\to$ to δ_4 (e)	$\to$

Figure 2: Summary of results. The entry at row xxx and column yyy gives the relationships we establish between notions xxx and yyy. Here $\delta_1 = 2^{n-m}$, $\delta_2 = 1 - 2^{n-m-1}$, $\delta_3 = 2^{-m}$, $\delta_4 = 1/|\mathcal{K}|$, and $\delta_5 = 2^{1-m}$. The hash functions $H1, \ldots, H6$ and $G1, G2, G3$ are specified in Figure 3. The annotations (a)-(j) mean: (a) see Theorem 7; (b) by $G1$, see Proposition 9; (c) by $G3$, see Proposition 10; (d) by $H1$, see Theorem 15; (e) by $H2$, see Theorem 15 (f) by $H6$, see Theorem 14; (g) by $H6$, see Theorem 13; (h) by $H3$, see Theorem 15; (i) by $H4$, see Theorem 15; (j) by $G2$, see Theorem 11; (k) by $H5$, see Theorem 11; (l) see Proposition 6

$$
\begin{aligned}
H1_K(M) &= \begin{cases} 0^n \text{ if } M = 0^m \\ H_K(M) \text{ otherwise} \end{cases} \\
H2_K(M) &= \begin{cases} 0^n \text{ if } K = K_0 \\ H_K(M) \text{ otherwise} \end{cases} \\
H3^b_K(M) &= H_K(M[1..m-1] \parallel b) \\
H4_K(M) &= \begin{cases} 0^n \text{ if } M = 0^m \text{ or } M = 1^m \\ H_K(M) \text{ otherwise} \end{cases} \\
H5^c_K(M) &= \begin{cases} H_K(0^{m-n} \parallel H_K(c)) \text{ if } M = 1^{m-n} \parallel H_K(c) & (1) \\ H_K(M) \text{ otherwise} & (2) \end{cases} \\
H6_K(M) &= \begin{cases} 0^n \text{ if } M = 0^m & (1) \\ H_K(M) \text{ if } M \neq 0^m \text{ and } H_K(M) \neq 0^n & (2) \\ H_K(0^m) \text{ otherwise} & (3) \end{cases} \\
G1_K(M) &= \begin{cases} M[1..n] \text{ if } M[n+1..m] = 0^{m-n} \\ 0^n \text{ otherwise} \end{cases} \\
G2_K(M) &= \begin{cases} 1^{n-m} \parallel K \text{ if } M \in \{K, \overline{K}\} \\ 0^{n-m} \parallel M \text{ otherwise} \end{cases} \\
G3_K(M) &= \begin{cases} \langle i \rangle_n \text{ if } M = \langle (K+i) \bmod 2^m \rangle_m \text{ for some } i \in [1..2^n - 1] \\ 0^n \text{ otherwise} \end{cases}
\end{aligned}
$$

Figure 3: Given a hash function $H\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ we construct hash functions $H1, \ldots, H6\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ for our conditional separations. The value $K_0 \in \mathcal{K}$ is fixed and arbitrary. The hash functions $G1\colon \{\varepsilon\} \times \{0,1\}^m \to \{0,1\}^n$, $G2\colon \{0,1\}^m \times \{0,1\}^m \to \{0,1\}^n$, $G3\colon \{1, \ldots, 2^m - 1\} \times \{0,1\}^m \to \{0,1\}^n$, are used in our unconditional separations.

Definition 1 **[Types of preimage resistance]** Let $H = \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash-function family and let m be a number such that $\{0,1\}^m \subseteq \mathcal{M}$. Let A be an adversary. Then define:

$$\begin{aligned}
\mathbf{Adv}_H^{\mathrm{Pre}\,[m]}(A) &= \Pr\left[K \xleftarrow{\$} \mathcal{K};\, M \xleftarrow{\$} \{0,1\}^m;\, Y \leftarrow H_K(M);\, M' \xleftarrow{\$} A(K,Y):\ H_K(M') = Y\right] \\
\mathbf{Adv}_H^{\mathrm{ePre}}(A) &= \max_{Y \in \mathcal{Y}} \left\{\Pr\left[K \xleftarrow{\$} \mathcal{K};\, M \xleftarrow{\$} A(K):\ H_K(M) = Y\right]\right\} \\
\mathbf{Adv}_H^{\mathrm{aPre}\,[m]}(A) &= \max_{K \in \mathcal{K}} \left\{\Pr\left[M \xleftarrow{\$} \{0,1\}^m;\, Y \leftarrow H_K(M);\, M' \xleftarrow{\$} A(Y):\ H_K(M') = Y\right]\right\}
\end{aligned}$$

The first definition, *preimage resistance* (Pre), is the usual way to define when a hash-function family is a *one-way function*. (Of course the notion is different from a function $f\colon \mathcal{M} \to \mathcal{Y}$ being a one-way function, as these are syntactically different objects.) The second definition, *everywhere preimage-resistance* (ePre), most directly captures the intuition that it is infeasible to find the preimage of range points: for *whatever* range point is selected, it is computationally hard to find its preimage. The final definition, *always preimage-resistance* (aPre), strengthens the first definition in the way needed to say that a function like SHA1 is one-way: one regards SHA1 as one function from a family of hash functions (keyed, for example, by the initial chaining value) and we wish to say that for this particular function from the family it remains hard to find a preimage of a random point.

3.2 Second-preimage resistance

It is likewise possible to formalize multiple definitions that might be understood as technical meaning for second-preimage resistance. In all cases a domain point M and a description of a hash function H_K are known to the adversary, whose job it is to find an M' different from M such that $H(K,M) = H(K,M')$. Such an M and M' are called *partners*.

Definition 2 **[Types of second-preimage resistance]** Let $H\colon\ \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash-function family and let m be a number such that $\{0,1\}^m \subseteq \mathcal{M}$. Let A be an adversary. Then define:

$$\begin{aligned}
\mathbf{Adv}_H^{\mathrm{Sec}\,[m]}(A) &= \Pr\left[K \xleftarrow{\$} \mathcal{K};\, M \xleftarrow{\$} \{0,1\}^m;\, M' \xleftarrow{\$} A(K,M):\ (M \neq M') \wedge (H_K(M) = H_K(M'))\right] \\
\mathbf{Adv}_H^{\mathrm{eSec}\,[m]}(A) &= \max_{M \in \{0,1\}^m} \left\{\Pr\left[K \xleftarrow{\$} \mathcal{K};\, M' \xleftarrow{\$} A(K):\ (M \neq M') \wedge (H_K(M) = H_K(M'))\right]\right\} \\
\mathbf{Adv}_H^{\mathrm{aSec}\,[m]}(A) &= \max_{K \in \mathcal{K}} \left\{\Pr\left[M \xleftarrow{\$} \{0,1\}^m;\, M' \xleftarrow{\$} A(M):\ (M \neq M') \wedge (H_K(M) = H_K(M'))\right]\right\}
\end{aligned}$$

The first definition, *second-preimage resistance* (Sec), is the standard one. The second definition, *everywhere second-preimage resistance* (eSec), most directly formalizes that it is hard to find a partner for any particular domain point. This notion is also called a *universal one-way hash-function family* (UOWHF) and it was first defined by Naor and Yung [12]. The final definition, *always second-preimage resistance* (aSec), strengthens the first in the way needed to say that a function like SHA1 is second-preimage resistant: one regards SHA1 as one function from a family of hash functions and we wish to say that for this particular function it is remains hard to find a partner for a random point.

3.3 Collision resistance

Finally, we would like to speak of the difficulty with which an adversary is able to find two distinct points in the domain of a hash function that hash to the same range point.

Definition 3 [Collision resistance] Let $H\colon \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash-function family and let A be an adversary. Then we define:

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

It does not make sense to think of strengthening this definition by maximizing over all $K \in \mathcal{K}$: for any fixed function $h\colon \mathcal{M} \to \mathcal{Y}$ with $|\mathcal{M}| > |\mathcal{Y}|$ there is is an efficient algorithm that outputs an M and M' that collide under h. While this program might be hard to find in practice, there is no known sense in which this can be formalized.

4 Equivalent Formalizations with a Two-Stage Adversary

Four of our definitions (ePre, aPre, eSec, aSec) maximize over some quantity that one may imagine the adversary to know. In each of these cases it possible to modify the definition so as to have the adversary itself choose this value. That is, in a "first phase" of the adversary's execution it chooses the quantity in question, and then a random choice is made by the environment, and then the adversary continues from where it left off, but now given this randomly chosen value. The corresponding definitions are then as follows:

Definition 4 [Equivalent versions of ePre, aPre, eSec, aSec] Let $H = \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash-function family and let m be a number such that $\{0,1\}^m \subseteq \mathcal{M}$. Let A be an adversary. Then define:

$$\begin{aligned}
\mathbf{Adv}_H^{\mathrm{ePre}}(A) &= \Pr\left[(Y, S) \xleftarrow{\$} A();\, K \xleftarrow{\$} \mathcal{K};\, M \xleftarrow{\$} A(K, S):\ H_K(M) = Y\right] \\
\mathbf{Adv}_H^{\mathrm{aPre}\,[m]}(A) &= \Pr\left[(K, S) \xleftarrow{\$} A();\, M \xleftarrow{\$} \{0,1\}^m;\, Y \leftarrow H_K(M);\, M' \xleftarrow{\$} A(Y, S):\ H_K(M') = Y\right] \\
\mathbf{Adv}_H^{\mathrm{eSec}\,[m]}(A) &= \Pr\left[(M, S) \xleftarrow{\$} A();\, K \xleftarrow{\$} \mathcal{K};\, M' \xleftarrow{\$} A(K, S):\ (M \neq M') \wedge (H_K(M) = H_K(M'))\right] \\
\mathbf{Adv}_H^{\mathrm{aSec}\,[m]}(A) &= \Pr\left[(K, S) \xleftarrow{\$} A();\, M \xleftarrow{\$} \{0,1\}^m;\, M' \xleftarrow{\$} A(M, S):\ (M \neq M') \wedge (H_K(M) = H_K(M'))\right]
\end{aligned}$$

In the two-stage definition of $\mathbf{Adv}_H^{\mathrm{eSec}\,[m]}(A)$ we insist that the message M output by A is of length m bits, that is $M \in \{0,1\}^m$. Each of these four definitions are extended to their resource-parameterized version in the usual way.

The two-stage definitions above are easily seen to be equivalent to their one-stage counterparts. Saying here that definitions xxx and yyy are *equivalent* means that there is a constant C such that $\mathbf{Adv}_H^{\mathrm{xxx}\,[m]}(t) \leq \mathbf{Adv}_H^{\mathrm{yyy}\,[m]}(C(t+m+n))$ and $\mathbf{Adv}_H^{\mathrm{yyy}\,[m]}(t) \leq \mathbf{Adv}_H^{\mathrm{xxx}\,[m]}(C(t+m+n))$. Omit mention of $+m$ and $[m]$ in the definition for everywhere preimage resistance since this does not depend on m. Since the exact interpretation of time t was model-dependent anyway, two measures of adversarial advantage that are equivalent need not be distinguished.

We give an example of the equivalence of one-stage and two-stage adversaries, explaining why eSec and eSec2 are equivalent, where eSec2 temporarily denotes the version of eSec defined in Definition 4 (and eSec refers to what is given in Definition 2). Let A attack hash function H in the eSec sense. For every fixed M there is a two-stage adversary $A2$ that does as well as A at finding a partner for M. Specifically, let $A2$ be an adversary with the value M "hardwired in" to it. Adversary $A2$ prints out M and when it resumes it behaves like A. Similarly, let $A2$ be a two-stage adversary attacking H in the eSec2 sense. Consider the random coins used by $A2$ during its first stage and choose specific coins that maximize the probability that $A2$ will subsequently succeed. For these coins there is a specific pair (M, S) that $A2$ returns. Let A be a (one-stage) adversary that on input (K, M) runs exactly as $A2$ would on input (K, S).

5 Implications

Definitions of implications. In this section we investigate which of our notions of security (Pre, aPre, ePre, Sec, aSec, eSec, and Coll) imply which others. First we explain our notion of an implication.

Definition 5 [Implications] Fix $\mathcal{K}$, $\mathcal{M}$, m, and n where $\{0,1\}^m \subseteq \mathcal{M}$. Suppose that xxx and yyy are labels for which $\mathbf{Adv}_H^{\mathrm{xxx}\,\cdot}$ and $\mathbf{Adv}_H^{\mathrm{yyy}\,\cdot}$ have been defined for any $H\colon \mathcal{K}\times\mathcal{M}\to\{0,1\}^n$.

- *Conventional implication.* We say that xxx **implies** yyy, written xxx $\to$ yyy, if $\mathbf{Adv}_H^{\mathrm{yyy}\,\cdot}(t) \le c\,\mathbf{Adv}_H^{\mathrm{xxx}\,\cdot}(t')$ for all hash functions $H\colon \mathcal{K}\times\mathcal{M}\to\{0,1\}^n$ where c is an absolute constant and $t' = t + c\,\mathrm{Time}_{H,m}$.
- *Provisional implication.* We say that xxx **implies** yyy **to** ϵ, written xxx $\to$ yyy to ϵ, if $\mathbf{Adv}_H^{\mathrm{yyy}\,\cdot}(t) \le c\,\mathbf{Adv}_H^{\mathrm{xxx}\,\cdot}(t') + \epsilon$ for all hash functions $H\colon \mathcal{K}\times\mathcal{M}\to\{0,1\}^n$ where c is an absolute constant and $t' = t + c\,\mathrm{Time}_{H,m}$. ▮

In the definition above, and later, the $\cdot$ is a placeholder which is either $[m]$ (for Pre, aPre, Sec, aSec, eSec) or empty (for ePre, Coll).

Conventional implications are what one expects: xxx $\to$ yyy means that if a hash function is secure in the xxx-sense, then it is secure in the yyy-sense. Whether or not a provisional implication carries the usual semantics of the word *implication* depends on the value of ϵ. Below we will demonstrate provisional implications with a value of $\epsilon = 2^{n-m}$ and so the interpretation of such a result is that we have demonstrated a "real" implication for hash functions that are substantially compressing (e.g., if the hash function maps 256 bits to 128 bits) while we have given a *non-result* if the hash function is length-preserving, length-increasing, or it compresses just a little.

Conventional implications. The conventional implications among our notions are straightforward, so we quickly dispense with those, omitting the proofs. In particular, the following are easily verified.

Proposition 6 [Conventional implications] Fix $\mathcal{K}$, $\mathcal{M}$, m, such that $\{0,1\}^m \subseteq \mathcal{M}$, and $n > 0$. Let Coll, Pre, aPre, ePre, Sec, aSec, eSec be the corresponding security notions. Then:

(1) Coll $\to$ Sec
(2) Coll $\to$ eSec
(3) aSec $\to$ Sec
(4) eSec $\to$ Sec
(5) aPre $\to$ Pre
(6) ePre $\to$ Pre ▮

In addition to the above, of course xxx $\to$ xxx for each notion xxx that we have given.

Provisional implications. We now give five provisional implications. The value of ϵ implicit in these claims depends on the relative difference of the domain length m and the hash length n. Intuitively, one can follow paths through the graph in Figure 1, composing implications to produce the five provisional implications. The formal proof of these five results appears in Appendix B.1.

Theorem 7 **[Provisional implications]** Fix $\mathcal{K}$, $\mathcal{M}$, m, such that $\{0,1\}^m \subseteq \mathcal{M}$, and $n > 0$. Let Coll, Pre, aPre, Sec, aSec, eSec be the corresponding security notions. Then:

(1) Sec $\rightarrow$ Pre to 2^{n-m}
(2) aSec $\rightarrow$ Pre to 2^{n-m}
(3) eSec $\rightarrow$ Pre to 2^{n-m}
(4) Coll $\rightarrow$ Pre to 2^{n-m}
(5) aSec $\rightarrow$ aPre to 2^{n-m}

▮

6 Separations

Definitions. We now investigate separations among our seven security notions. We emphasize that asserting a separation—which we will also call a *nonimplication*—is *not* the assertion of a lack of an implication (though it does effectively imply this for any practical hash function). In fact, we will show that both a separation and an implication can exist between two notions, the relative strength of the separation/implication being determined by the amount of compression performed by the hash function. Intuitively, xxx nonimplies yyy if it is possible for something to be xxx-secure but not yyy-secure. We provide two variants of this idea. The first notion, a *conventional* nonimplication, says that if H is a hash function that is secure in the xxx-sense then H can be converted into a hash function H' having the same domain and range that is still secure in the xxx-sense but that is now completely *insecure* in the yyy-sense. The second notion, an *unconditional* nonimplication, says that there is a hash function H that is secure in the xxx-sense but completely insecure in the yyy-sense. Thus the first kind of separation effectively assumes an xxx-secure hash function in order to separate xxx from yyy, while the second kind of separation does not need to do this.[3]

Definition 8 **[Separations]** Fix $\mathcal{K}$, $\mathcal{M}$, m, and n where $\{0,1\}^m \subseteq \mathcal{M}$. Suppose that xxx and yyy be labels for which $\mathbf{Adv}_H^{\mathrm{xxx}}$ and $\mathbf{Adv}_H^{\mathrm{yyy}}$ have been defined for any $H\colon \mathcal{K} \times \mathcal{M} \rightarrow \{0,1\}^n$.

- *Conventional separation.* We say that xxx **nonimplies** yyy **to** ϵ, in the conventional sense, written xxx $\not\rightarrow$ yyy to ϵ, if for any $H\colon \mathcal{K}\times\mathcal{M} \rightarrow \{0,1\}^n$ there exists an $H'\colon \mathcal{K}\times\mathcal{M} \rightarrow \{0,1\}^n$ such that $\mathbf{Adv}_{H'}^{\mathrm{xxx}}(t) \le c\,\mathbf{Adv}_H^{\mathrm{xxx}}(t') + \epsilon$ and yet $\mathbf{Adv}_{H'}^{\mathrm{yyy}}(t') = 1$ where c is an absolute constant and $t' = t + c\,\mathrm{Time}_{H,m}$.

- *Unconditional separation.* We say that xxx **nonimplies** yyy **to** ϵ, in the unconditional sense, written xxx $\not\rightharpoonup$ yyy to ϵ, if there exists an $H\colon \mathcal{K}\times\mathcal{M} \rightarrow \{0,1\}^n$ such that $\mathbf{Adv}_H^{\mathrm{xxx}}(t) \le \epsilon$ for all t and yet $\mathbf{Adv}_H^{\mathrm{yyy}}(t') = 1$ where $t' = c\,\mathrm{Time}_{H,m}$ for some absolute constant c. ▮

When $\epsilon = 0$ above we say that we have a *strong* separation and we omit saying "to ϵ" in speaking of it. When $\epsilon > 0$ above we say that we have a *provisional* separation. The degree to which a provisional separation should be regarded as a "real" separation depends on the value ϵ.

Some provisional separations. The following separations depend on the relative values of the domain size m and the range size n. As an example, if the hash-function family H is length-preserving, meaning $H\colon \mathcal{K}\times\{0,1\}^n \rightarrow \{0,1\}^n$, then it being second preimage resistant won't imply it being preimage resistant: just consider the identify function, which is perfectly second preimage

[3]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).

resistant (no domain point has a partner) but trivially breakable in the sense of finding preimages. This counterexample is well-known. We now generalize and extend this counterexample, giving a "gap" of $1 - 2^{n-m-1}$ for three of our pairs of notions. Thus we have a strong separation when $m = n$ and a rapidly weakening separation as m exceeds n by more and more. Taken together with Proposition 7 we see that this behavior is not an artifact of the proof: as m exceeds n, the 2^{n-m}-implication we have given effectively takes over.

Proposition 9 [Separations, part 1a] Fix $m \geq n > 0$ and let Sec, Pre, aSec, aPre be the corresponding security notions. Then:

(1) Sec $\not\rightharpoonup$ Pre to $1 - 2^{n-m-1}$
(2) aSec $\not\rightharpoonup$ Pre to $1 - 2^{n-m-1}$
(3) aSec $\not\rightharpoonup$ aPre to $1 - 2^{n-m-1}$ ▮

The proof is given in Appendix B.2.

Proposition 10 [Separations, part 1b] Fix $m \geq n > 0$, and let Pre and eSec be the corresponding security notions. Then eSec $\not\rightharpoonup$ Pre to $1 - 2^{n-m-1}$.

The proof is given in Appendix B.3.

Additional Separations. We now give some further nonimplications. Unlike those just given, these nonimplications do not have a corresponding provisional implication. Here, the separation is the whole story of the relationship between the notions, and the strength of the separation is not dependent on the amount of compression performed by the hash function.

Theorem 11 [Separations, part 2A] Fix $m > n > 0$ and let eSec and Coll be the corresponding security notions. Then eSec $\not\to$ Coll. ▮

The proof is in Appendix B.4. Because of the structure of the counterexample used in Theorem 11, we give the following proposition for completeness.

Proposition 12 Fix $n > 0$ and $m \leq n$, and let eSec and Coll be the corresponding security notions. Then eSec $\not\rightharpoonup$ Coll to $2^{-(m+1)}$. ▮

The proof appears in Appendix B.5

Theorem 13 [Separations, part 2B] Fix m, n such that $n > 0$, and let Coll and ePre be the corresponding security notions. Then Coll $\not\to$ ePre. ▮

The proof of the theorem above is in Appendix B.6.

Theorem 14 [Separations, part 2C] Fix m, n such that $n > 0$, and let eSec and ePre be the corresponding security notions. Then eSec $\not\to$ ePre. ▮

The proof of the theorem above is in Appendix B.7.

The remaining 28 separations are not as hard to show those given so far, so we present them as one theorem and without proof. The specific constructions $H1, H2, H3, H4$ are those given in Figure 3.

Theorem 15 **[Separations, part 3]** Fix m, n such that $n > 0$, and let Coll, Pre, aPre, ePre, Sec, aSec, eSec be the corresponding security notions. Let $H\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ be a hash function and define $H1, \ldots, H6$ from it according to Figure 3. Then:

(1) Pre $\not\to$ ePre to 2^{-m} : $\mathbf{Adv}_{H1}^{\mathrm{Pre}}(t) \le 1/2^m + \mathbf{Adv}_{H}^{\mathrm{Pre}}(t)$ and $\mathbf{Adv}_{H1}^{\mathrm{ePre}}(t') = 1$

(2) Pre $\not\to$ aPre to $1/|\mathcal{K}|$: $\mathbf{Adv}_{H2}^{\mathrm{Pre}}(t) \le 1/|\mathcal{K}| + \mathbf{Adv}_{H}^{\mathrm{Pre}}(t)$ and $\mathbf{Adv}_{H2}^{\mathrm{aPre}}(t') = 1$

(3) Pre $\not\to$ Sec: $\mathbf{Adv}_{H3}^{\mathrm{Pre}}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{Pre}}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{Sec}}(t') = 1$

(4) Pre $\not\to$ eSec: $\mathbf{Adv}_{H3}^{\mathrm{Pre}}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{Pre}}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{eSec}}(t') = 1$

(5) Pre $\not\to$ aSec: $\mathbf{Adv}_{H3}^{\mathrm{Pre}}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{Pre}}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{aSec}}(t') = 1$

(6) Pre $\not\to$ Coll: $\mathbf{Adv}_{H3}^{\mathrm{Pre}}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{Pre}}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{Coll}}(t') = 1$

(7) ePre $\not\to$ aPre to $1/|\mathcal{K}|$: $\mathbf{Adv}_{H2}^{\mathrm{ePre}}(t) \le 1/|\mathcal{K}| + \mathbf{Adv}_{H}^{\mathrm{ePre}}(t)$ and $\mathbf{Adv}_{H2}^{\mathrm{aPre}\,[m]}(t') = 1$

(8) ePre $\not\to$ Sec: $\mathbf{Adv}_{H3}^{\mathrm{ePre}}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{ePre}}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{Sec}\,[m]}(t') = 1$

(9) ePre $\not\to$ eSec: $\mathbf{Adv}_{H3}^{\mathrm{ePre}}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{ePre}}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{eSec}\,[m]}(t') = 1$

(10) ePre $\not\to$ aSec: $\mathbf{Adv}_{H3}^{\mathrm{ePre}}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{ePre}}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{aSec}\,[m]}(t') = 1$

(11) ePre $\not\to$ Coll: $\mathbf{Adv}_{H3}^{\mathrm{ePre}}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{ePre}}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{Coll}}(t') = 1$

(12) aPre $\not\to$ ePre to 2^{-m}: $\mathbf{Adv}_{H1}^{\mathrm{aPre}\,[m]}(t) \le 1/2^m + \mathbf{Adv}_{H}^{\mathrm{aPre}\,[m]}(t)$ and $\mathbf{Adv}_{H1}^{\mathrm{ePre}}(t') = 1$

(13) aPre $\not\to$ Sec: $\mathbf{Adv}_{H3}^{\mathrm{aPre}\,[m]}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{aPre}\,[m]}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{Sec}\,[m]}(t') = 1$

(14) aPre $\not\to$ eSec: $\mathbf{Adv}_{H3}^{\mathrm{aPre}\,[m]}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{aPre}\,[m]}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{eSec}\,[m]}(t') = 1$

(15) aPre $\not\to$ aSec: $\mathbf{Adv}_{H3}^{\mathrm{aPre}\,[m]}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{aPre}\,[m]}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{aSec}\,[m]}(t') = 1$

(16) aPre $\not\to$ Coll: $\mathbf{Adv}_{H3}^{\mathrm{aPre}\,[m]}(t) \le 2 \cdot \mathbf{Adv}_{H}^{\mathrm{aPre}\,[m]}(t)$ and $\mathbf{Adv}_{H3}^{\mathrm{Coll}}(t') = 1$

(17) Sec $\not\to$ ePre to 2^{-m}: $\mathbf{Adv}_{H1}^{\mathrm{Sec}\,[m]}(t) \le 1/2^m + \mathbf{Adv}_{H}^{\mathrm{Sec}\,[m]}(t)$ and $\mathbf{Adv}_{H1}^{\mathrm{ePre}}(t') = 1$

(18) Sec $\not\to$ aPre to $1/|\mathcal{K}|$: $\mathbf{Adv}_{H2}^{\mathrm{Sec}\,[m]}(t) \le 1/|\mathcal{K}| + \mathbf{Adv}_{H}^{\mathrm{Sec}\,[m]}(t)$ and $\mathbf{Adv}_{H2}^{\mathrm{aPre}\,[m]}(t') = 1$

(19) Sec $\not\to$ eSec to 2^{-m+1}: $\mathbf{Adv}_{H4}^{\mathrm{Sec}\,[m]}(t) \le 1/2^{m-1} + \mathbf{Adv}_{H}^{\mathrm{Sec}\,[m]}(t)$ and $\mathbf{Adv}_{H4}^{\mathrm{eSec}\,[m]}(t') = 1$

(20) Sec $\not\to$ aSec to 2^{-m}: $\mathbf{Adv}_{H2}^{\mathrm{Sec}\,[m]}(t) \le 1/|\mathcal{K}| + \mathbf{Adv}_{H}^{\mathrm{Sec}\,[m]}(t)$ and $\mathbf{Adv}_{H2}^{\mathrm{aSec}\,[m]}(t') = 1$

(21) Sec $\not\to$ Coll to 2^{-m+1}: $\mathbf{Adv}_{H4}^{\mathrm{Sec}\,[m]}(t) \le 1/2^{m-1} + \mathbf{Adv}_{H}^{\mathrm{Sec}\,[m]}(t)$ and $\mathbf{Adv}_{H4}^{\mathrm{Coll}}(t') = 1$

(22) eSec $\not\to$ aPre to $1/|\mathcal{K}|$: $\mathbf{Adv}_{H2}^{\mathrm{eSec}\,[m]}(t) \le 1/|\mathcal{K}| + \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(t)$ and $\mathbf{Adv}_{H2}^{\mathrm{aPre}\,[m]}(t') = 1$

(23) eSec $\not\to$ aSec to $1/|\mathcal{K}|$: $\mathbf{Adv}_{H2}^{\mathrm{eSec}\,[m]}(t) \le 1/|\mathcal{K}| + \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(t)$ and $\mathbf{Adv}_{H2}^{\mathrm{aSec}\,[m]}(t') = 1$

(24) aSec $\not\to$ ePre to 2^{-m}: $\mathbf{Adv}_{H1}^{\mathrm{aSec}\,[m]}(t) \le 1/2^m + \mathbf{Adv}_{H}^{\mathrm{aSec}\,[m]}(t)$ and $\mathbf{Adv}_{H1}^{\mathrm{ePre}}(t') = 1$

(25) aSec $\not\to$ eSec to 2^{-m}: $\mathbf{Adv}_{H4}^{\mathrm{aSec}\,[m]}(t) \le 1/2^{m-1} + \mathbf{Adv}_{H}^{\mathrm{aSec}\,[m]}(t)$ and $\mathbf{Adv}_{H4}^{\mathrm{eSec}\,[m]}(t') = 1$

(26) aSec $\not\to$ Coll to 2^{-m+1}: $\mathbf{Adv}_{H4}^{\mathrm{aSec}\,[m]}(t) \le 1/2^{m-1} + \mathbf{Adv}_{H}^{\mathrm{aSec}\,[m]}(t)$ and $\mathbf{Adv}_{H4}^{\mathrm{Coll}}(t') = 1$

(27) Coll $\not\to$ aPre to $1/|\mathcal{K}|$: $\mathbf{Adv}_{H2}^{\mathrm{Coll}}(t) \le 1/|\mathcal{K}| + \mathbf{Adv}_{H}^{\mathrm{Coll}}(t)$ and $\mathbf{Adv}_{H2}^{\mathrm{aPre}}(t') = 1$

(28) Coll $\not\to$ aSec to $1/|\mathcal{K}|$: $\mathbf{Adv}_{H2}^{\mathrm{Coll}}(t) \le 1/|\mathcal{K}| + \mathbf{Adv}_{H}^{\mathrm{Coll}}(t)$ and $\mathbf{Adv}_{H2}^{\mathrm{aSec}}(t') = 1$

where $t' = c\,\mathrm{Time}_{H,m}$ for some absolute constant c. ▮

Acknowledgments

Thanks to Mihir Bellare and to various anonymous reviewers, who provided useful comments on an earlier draft of this paper.

This work was supported by NSF 0085961, NSF 0208842, and a gift from Cisco Systems. Many thanks to the NSF and Cisco for their support. Work on this paper was carried out while the authors were at Chiang Mai University, Chulalongkorn University, and UC Davis.

References

[1] R. Anderson. The classification of hash functions. In *IMA Conference in Cryptography and Coding IV*, pages 83–94, December 1993.

[2] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In H. Krawczyk, editor, *Advances in Cryptology – CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 232–249. Springer-Verlag, 1998.

[3] M. Bellare and P. Rogaway. Collision-resistant hashing: Towards making UOWHFs practical. In *Advances in Cryptology – CRYPTO 97*, volume 1294 of *Lecture Notes in Computer Science*, pages 470–484, 1997.

[4] J. Black, P. Rogaway, and T. Shrimpton. Black-box analysis of the block-cipher-based hash-function constructions from PGV. In *Advances in Cryptology – CRYPTO '02*, volume 2442 of *Lecture Notes in Computer Science*. Springer-Verlag, 2002.

[5] D. Brown and D. Johnson. Formal security proofs for a signature scheme with partial message recovery. *Lecture Notes in Computer Science*, 2020:126–144, 2001.

[6] I. Damgård. Collision free hash fucntions and public key signature schemes. In *Advances in Cryptology – EUROCRYPT '87*, volume 304 of *Lecture Notes in Computer Science*. Springer-Verlag, 1988.

[7] I. Damgård. A design principle for hash functions. In G. Brassard, editor, *Advances in Cryptology – CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990.

[8] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, April 1984.

[9] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.

[10] R. Merkle. One way hash functions and DES. In G. Brassard, editor, *Advances in Cryptology – CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990.

[11] I. Mironov. Hash functions: From Merkle-Damgård to Shoup. In *Advances in Cryptology – EUROCRYPT '01*, Lecture Notes in Computer Science. Springer-Verlag, 2001.

[12] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the Twenty-first ACM Symposium on Theory of Computing*, pages 33–43, 1989.

[13] B. Preneel. *Cryptographic hash functions.* Katholieke Universiteit Leuven (Belgium), 1993.

[14] P. Rogaway and T. Shrimpton. Cryptographic hash-function basics: Definitions, implications and separations for preimage resistance, second-preimage resistance, and collision resistance. Full version of this paper,www.cs.ucdavis.edu/˜rogaway, 2004.

[15] D. Stinson. Some observations on the theory of cryptographic hash functions. Technical Report 2001/020, University of Waterloo, 2001.

[16] Y. Zheng, T. Matsumoto, and H. Imai. Connections among several versions of one-way hash functions. In *Special Issue on Cryptography and Information Security, Proceedings of IEICE of Japan*, 1990.

A Brief History

It is beyond the scope of the current work to give a full survey of the many hash-function security-notions in the literature, formal an informal, and the many relationships that have (and have not) been shown among them. We touch upon some of the more prominent work that we know.

The term *universal one-way hash function*(UOWHF) was introduced by Naor and Yung [12] to name their asymptotic definition of second-preimage resistance. Along with Damgård [6, 7], who introduced the notion of *collision freeness*, these papers were the first to put notions of hash-function security on a solid formal footing by suggesting to study keyed family of hash functions. This was a necessary step for developing a meaningful formalization of collision-resistance. Contemporaneously, Merkle [10] describes notions of hash-function security: *weak collision resistance* and *strong collision resistance*, which refer to second-preimage and collision resistance, respectively. Damgård also notes that a compressing collision-free hash function has one-wayness properties (our pre notion), and points out some subtleties in this implication.

Merkle and Damgård [7, 10] each show that if one properly iterates a collision-resistant function with a fixed domain, then one can construct a collision-resistant hash-function with an enlarged domain. This iterative method is now called the Merkle-Damgård construction.

Preneel [13] describes *one-way hash functions* (those which are both preimage-resistant and second-preimage resistant) and *collision-resistant hash functions* (those which are preimage, second-preimage and collision resistant). He identifies four types of attacks and studies hash functions constructed from block ciphers.

Bellare and Rogaway [3] give concrete-security definitions for hash-function security and study second-preimage resistance and collision resistance. Their *target collision-resistance*(TCR) coincides with a UOWHF (eSec) and their *any collision-resistance*(ACR) coincides with Coll-security.

Brown and Johnson [5] define a *strong hash* that, if properly formalized in the concrete setting, would include our ePre notion.

Mironov [11] investigates a class of asymptotic definitions that bridge between conventional collision resistance and UOWHF. He also looks at which members of that class are preserved by the Merkle-Damgård constructions.

Anderson [1] discusses some unconventional notions of security for hash functions that might arise when one considers how hash functions might interact with higher-level protocols.

Black, Rogaway, and Shrimpton [4] use a concrete definition of preimage resistance that requires inversion of a uniformly selected range point.

Two papers set out on a program somewhat similar to ours [15] and [16]. Stinson [15] considers hash function security from the perspective that the notions of primary interest are those related to producing digital signatures. He considers four problems (zero-preimage, preimage, second-preimage, collision) and describes notions of security based on them. He considers in some depth the relationship between the preimage problem and the collision problem.

Zheng, Matsumoto and Imai [16] examine some asymptotic formalizations of the notions of second-preimage resistance and collision resistance. In particular, they suggest five classes of second-preimage resistant hash functions and three classes of collision resistant hash functions, and then consider the relationships among these classes.

Our focus on provable security follows a line that begins with Goldwasser and Micali [8]. In defining several related notions of security and then working out all relations between them, we follow work like that of Bellare, Desai, Pointcheval, and Rogaway [2].

B Proofs

B.1 Proof of Theorem 7

We prove the first statement from the theorem; the other proof the others follows from this one. Let $H\colon \mathcal{K}\times\mathcal{M}\to\{0,1\}^n$ be a hash-function family. We will show that

$$\mathbf{Adv}_H^{\mathrm{Pre}\,[m]}(t) \;\le\; 2\,\mathbf{Adv}_H^{\mathrm{Sec}\,[m]}(t') + 2^{n-m}$$

where $t' = t + c\,\mathrm{Time}_{H,m}$ for some absolute constant c.

Let B be an adversary attacking H in the Pre-sense and let $\delta_m = \mathbf{Adv}_H^{\mathrm{Pre}\,[m]}(B)$ be its advantage and let t be its running time. We construct as follows an adversary A for attacking H in the Sec-sense: let A, on input (K,M), compute $Y \leftarrow H_K(M)$, run $B(K,Y)$, and return the value M' that B outputs. We now analyze the probability that A finds a partner for a random point M and a random hash function H_K.

Let $\mathsf{I}_K(M)$ be the event that a point $M\in\{0,1\}^m$ has no partner under H_K—that is, the event that there exists no $M'\neq M$ such that $H_K(M)=H_K(M')$. Let $\Pr_{K,M}[\cdot]$ denote the probability of an event in an experiment which begins by choosing $M \xleftarrow{\$} \{0,1\}^m$ and $K \xleftarrow{\$} \mathcal{K}$. Now

$$\begin{aligned}
\delta_m &= \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\xleftarrow{\$} B(K,Y):\; H_K(M')=Y\Big]\\
&= \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\xleftarrow{\$} B(K,Y):\; \mathsf{I}_K(M)\wedge(H_K(M')=Y)\Big]\\
&\quad + \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\xleftarrow{\$} B(K,Y):\; \overline{\mathsf{I}_K(M)}\wedge(M\neq M')\wedge(H_K(M')=Y)\Big]\\
&\quad + \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\xleftarrow{\$} B(K,Y):\; \overline{\mathsf{I}_K(M)}\wedge(M= M')\wedge(H_K(M')=Y)\Big]\\
&\le \Pr_{K,M}[\mathsf{I}_K(M)] + \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\xleftarrow{\$} B(K,Y):\; \overline{\mathsf{I}_K(M)}\wedge(M\neq M')\wedge(H_K(M')=Y)\Big]\\
&\quad + \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\xleftarrow{\$} B(K,Y):\; \overline{\mathsf{I}_K(M)}\wedge(M= M')\wedge(H_K(M')=Y)\Big]\\
&\le \frac{2^n}{2^m} + \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\xleftarrow{\$} B(K,Y):\; \overline{\mathsf{I}_K(M)}\wedge(M\neq M')\wedge(H_K(M')=Y)\Big]\\
&\quad + \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\xleftarrow{\$} B(K,Y):\; \overline{\mathsf{I}_K(M)}\wedge(M= M')\wedge(H_K(M')=Y)\Big]
\end{aligned}$$

That $\Pr_{K,M}[\mathsf{I}_K(M)] \le 2^{n-m}$ can be seen as follows. For any key $K\in\mathcal{K}$ there are at most 2^n points M such that $\mathsf{I}_K(M)$ occurs. The domain of H_K has $2^m \ge 2^n$ points so for any $K\in\mathcal{K}$ we have that $\Pr_x[\mathsf{I}_K(M)]\le 2^n/2^m$. Therefore $\Pr_{K,M}[\mathsf{I}_K(M)]\le 2^n/2^m$ as well. Continuing,

$$\begin{aligned}
\delta_m - \frac{2^n}{2^m} \;&\le\; \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\leftarrow B(K,Y):\; \overline{\mathsf{I}_K(M)}\wedge(M\neq M')\wedge(H_K(M')=Y)\Big]\\
&\quad + \Pr_{K,M}\Big[Y\leftarrow H_K(M);\, M'\leftarrow B(K,Y):\; \overline{\mathsf{I}_K(M)}\wedge(M= M')\wedge(H_K(M')=Y)\Big]
\end{aligned}$$

We claim that the first probability above is at least as large as the second. This is so because we choose M at random from $\{0,1\}^m$ and B has no information about M except its image under H_K. We know that $H_K(M)$ has at least two preimages so B's chance to name the one which is M is at most B's chance to name one that is not M. We conclude that

$$\begin{aligned}
\delta_m - \frac{2^n}{2^m} &\le 2\left(\Pr_{K,M}\left[Y \leftarrow H_K(M);\ M' \leftarrow B(K,Y):\ \overline{\mathsf{I}_K(M)} \wedge (M \neq M') \wedge (H_K(M') = Y)\right]\right) \\
&\le 2\left(\Pr_{K,M}\left[Y \leftarrow H_K(M);\ M' \leftarrow B(K,Y):\ (M \neq M') \wedge (H_K(M') = Y)\right]\right) \\
&= 2\left(\Pr_{K,M}\left[M' \leftarrow A(h,x):\ (M \neq M') \wedge (H_K(M') = H_K(M))\right]\right) \\
&= 2\ \mathbf{Adv}_H^{\mathrm{Sec}\,[m]}(A)
\end{aligned}$$

Thus $\mathbf{Adv}_H^{\mathrm{Pre}\,[m]}(A) \le 2\,\mathbf{Adv}_H^{\mathrm{Sec}\,[m]}(B) + 2^{n-m}$ and we are done.

B.2 Proof of Proposition 9

We prove the first statement, the next two statements being very similar. We show that there is a function $H\colon \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$ such that

$$\mathbf{Adv}_H^{\mathrm{Sec}\,[m]}(t) \ \le\ 1 - 2^{n-m-1} \quad \text{and} \quad \mathbf{Adv}_H^{\mathrm{Pre}\,[m]}(cm) = 1$$

for some absolute constant c. Let $H\colon \mathcal{K}\times\mathcal{M} \to \{0,1\}^n$ be the function $G1\colon\ \{\varepsilon\}\times\{0,1\}^m \to \{0,1\}^n$ given in Figure 3. For convenience, we write H for H_ε. We begin by exhibiting an adversary B that runs in time cm and achieves advantage $\mathbf{Adv}_H^{\mathrm{Pre}\,[m]}(B) = 1$. Adversary B takes input (K,Y). If $Y = 0^n$ then it returns 1^m; otherwise, it returns $Y \parallel 0^{m-n}$.

We now consider an arbitrary partner-finding adversary A and bound its maximal advantage. Let $\Pr_M[\cdot]$ denote the probability of an event in an experiment which begins by choosing $M \xleftarrow{\$} \{0,1\}^m$. Let $\mathsf{Z}(M)$ be shorthand for $M[n+1..m] = 0^{m-n}$. Then

$$\begin{aligned}
\mathbf{Adv}_H^{\mathrm{Sec}\,[m]}(A) &= \Pr_M[M' \xleftarrow{\$} A(\varepsilon,M) : (M \neq M') \ \wedge\ (H(M) = H(M'))] \\
&= \Pr_M[M' \xleftarrow{\$} A(\varepsilon,M) : (M \neq M') \ \wedge\ (H(M) = H(M')) \mid \mathsf{Z}(M) \ \wedge\ M \neq 0^m] \\
&\qquad \cdot \Pr_M[\mathsf{Z}(M) \ \wedge\ M \neq 0^m] \\
&\quad + \Pr_M[M' \xleftarrow{\$} A(\varepsilon,M) : (M \neq M') \ \wedge\ (H(M) = H(M')) \mid \overline{\mathsf{Z}(M)} \ \vee\ M = 0^m] \\
&\qquad \cdot \Pr_M[\overline{\mathsf{Z}(M)} \ \vee\ M = 0^m] \\
&= \Pr_M[M' \xleftarrow{\$} A(\varepsilon,M) : (M \neq M') \ \wedge\ (H(M) = H(M')) \mid \overline{\mathsf{Z}(M)} \ \vee\ M = 0^m] \\
&\qquad \cdot \Pr_M[\overline{\mathsf{Z}(M)} \ \vee\ M = 0^m]
\end{aligned}$$

where the last equality is true because if $M[n+1..m] = 0^{m-n}$ and $M \neq 0^m$ then A has no chance to find a partner for M. Continuing we have that $\mathbf{Adv}_H^{\mathrm{Sec}\,[m]}(A) \le (1)(1 - (2^n/2^m) + 1/2^m) = 1 - (2^n-1)/2^m \le 1 - 2^n/2^{m+1}$ and we are done.

B.3 Proof of Proposition 10

We show that there is a hash function $H\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ such that

$$\mathbf{Adv}_H^{\mathrm{eSec}\,[m]}(t) \;\leq\; 1 - 2^{n-m-1} \quad \text{and} \quad \mathbf{Adv}_H^{\mathrm{Pre}\,[m]}(cm) = 1$$

for some absolute constant c.

Let $H\colon \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$ be the function $G3\colon \{1, \ldots, 2^m - 1\} \times \{0,1\}^m \to \{0,1\}^n$ in Figure 3. Notice that the key K defines a set of $(2^n - 1)$ domain points that are bijectively mapped under H_K, and all other domain points are mapped to 0^n.

First we show that there exists an adversary B that runs in time cm for some absolute constant c and achieves advantage $\mathbf{Adv}_H^{\mathrm{Pre}\,[m]}(B) = 1$. Adversary B takes as input (K, Y) and returns $\langle K + i \bmod 2^m \rangle_m$ where $Y = \langle i \rangle_n$.

We now consider an arbitrary partner-finding adversary A and bound its maximal advantage.

$$\begin{aligned}
\mathbf{Adv}_H^{\mathrm{eSec}\,[m]}(A) &= \Pr\left[(M,S) \xleftarrow{\$} A();\, K \xleftarrow{\$} \mathcal{K};\, M' \xleftarrow{\$} A(K,S):\; (M \neq M') \wedge (H_K(M) = H_K(M'))\right] \\
&\leq \Pr\left[(M,S) \xleftarrow{\$} A();\, K \xleftarrow{\$} \mathcal{K};\, M' \xleftarrow{\$} A(K,S):\; (M \neq M') \wedge (H_K(M) = 0^n)\right] \\
&\leq \Pr\left[(M,S) \xleftarrow{\$} A();\, K \xleftarrow{\$} \mathcal{K};\, M' \xleftarrow{\$} A(K,S):\; H_K(M) = 0^n\right] \\
&\leq 1 - \frac{2^n - 1}{2^m} \;\leq\; 1 - \frac{2^n}{2^{m+1}}
\end{aligned}$$

where the first inequality holds because if $H_K(M) \neq M$ then the adversary has no chance to find a partner M' for M.

B.4 Proof of Theorem 11

Let $H\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ be a hash function family and let $H5\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ be the function defined in Figure 3. We show that

$$\mathbf{Adv}_{H5}^{\mathrm{eSec}\,[m]}(t) \leq 2\,\mathbf{Adv}_H^{\mathrm{eSec}\,[m]}(t') \quad \text{and} \quad \mathbf{Adv}_{H5}^{\mathrm{Coll}}(t') = 1$$

where $t' \leq t + \ell \mathrm{Time}_{H,m}$ for some absolute constant ℓ.

Let $\Pr_K$ denote probability taken over $K \in \mathcal{K}$. Given H we define for every $c \in \{0,1\}^m$ an n-bit string Y_c and a real number δ_c as follows. Let Y_c be the lexicographically first string that maximizes $\delta_c = \Pr_K[H_K(c) = Y_c]$. Over all pairs c, c' we select the lexicographically first pair c, c' (when considered as the $2n$-bit string $c \parallel c'$) such that $c \neq c'$ and $Y_c = Y_{c'}$ and δ_c is maximized (ie, $\Pr_K[H_K(c) = H_K(c')]$ is maximized). Now let $H5 = H5^c$ be defined according to Figure 3.

We begin by exhibiting an adversary T that gains $\mathbf{Adv}_{H5}^{\mathrm{Coll}}(T) = 1$ and runs in time ℓm for some absolute constant ℓ. On input $K \in \mathcal{K}$, let T output $M = 1^{m-n} \parallel H_K(c)$ and $M' = 0^{m-n} \parallel H_K(c)$.

Now we show that if H is strong in the eSec-sense then so is $H5$. Let A be a two-stage adversary that gains advantage $\delta_m = \mathbf{Adv}_{H5}^{\mathrm{eSec}\,[m]}(A)$ and runs in time t. Let second-preimage-finding adversaries B and C be constructed as follows:

Algorithm B

[Stage 1] On input ():

Run $(M, S) \leftarrow A()$

return (M, S)

[Stage 2] On input (K, S):

Run $M' \leftarrow A(K, S)$

if $M \neq M'$ **and** $M \neq 1^{m-n} \parallel H_K(c)$

then return M'

else return $0^{m-n} \parallel H_K(c)$

Algorithm C

[Stage 1] On input ():

return (c, ε)

[Stage 2] On input (K, S)

return c'

The central claim of the proof is as follows:

Claim: $\mathbf{Adv}_{H5}^{\mathrm{eSec}\,[m]}(A) \leq \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(B) + \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(C)$

Let us prove this claim. Recall that the job of A is to find an M and an M' such that $M \neq M'$ and $H5(M) = H5(M')$. Referring to the line numbers in Figure 3, we say that u-v is a collision if M caused $H5$ to output on line $u \in \{1, 2\}$ and $M' \neq M$ caused $H5$ to output on line $v \in \{1, 2\}$, and $H5(M) = H5(M')$. We analyze the four possible u-v collisions that A can create.

[Case 1-1] Adversary A does not win by creating a 1-1 collision because in this case $M = M'$.

[Case 2-2] Assume A wins by causing a 2-2 collision. In this case $M \neq M'$ and $M \neq 1^{m-n} \parallel H_K(c)$ and $M' \neq 1^{m-n} \parallel H_K(c)$. Thus $H_K(M) = H_K(M')$ and so B finds a collision under H. We have then that $\Pr_K[A \text{ wins by a 2-2 collision}] \leq \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(B)$.

[Case 1-2] Assume that A wins by creating a 1-2 collision. Then $M \neq M'$ and $M = 1^{m-n} \parallel H_K(c)$. We claim that in this case adversary C wins. To see this, note that $\Pr[M \stackrel{\$}{\leftarrow} A();\, K \stackrel{\$}{\leftarrow} \mathcal{K} :\ M = 1^{m-n} \parallel H_K(c)] = \Pr_K[H_K(c) = Y]$ for some fixed $Y \in \{0,1\}^n$. By the way we chose c and c' we have $\Pr_K[H_K(c) = Y] \leq \Pr_K[H_K(c) = Y_c] = \Pr_K[H_K(c) = Y_{c'}] = \Pr_K[H_K(c) = H_K(c')]$; hence $\Pr[M \stackrel{\$}{\leftarrow} A();\, K \stackrel{\$}{\leftarrow} \mathcal{K} :\ M = 1^{m-n} \parallel H_K(c)] \leq \Pr_K[H_K(c) = H_K(c')]$. The conclusion is that $\Pr_K[A \text{ wins by a 1-2 collision}] \leq \Pr[M \stackrel{\$}{\leftarrow} A();\, K \stackrel{\$}{\leftarrow} \mathcal{K} :\ M = 1^{m-n} \parallel H_K(c)] \leq \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(C)$.

[Case 2-1] Assume that A wins by creating a 2-1 collision. Then $M \neq M'$ and $M' = 1^{m-n} \parallel H_K(c)$, and so $H_K(M) = H_K(0^{m-n} \parallel H_K(c))$. We claim that in this case either adversary B wins, or C does. Let BAD be the event that $M = 0^{m-n} \parallel H_K(c)$. If $M \neq 0^{m-n} \parallel H_K(c)$ then clearly B wins, so $\Pr_K[A \text{ wins by a 2-1 collision } \wedge\ \overline{\mathsf{BAD}}] \leq \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(B)$. If $M = 0^{m-n} \parallel H_K(c)$ then we have that $\Pr_K[A \text{ wins by a 2-1 collision } \wedge\ \mathsf{BAD}] \leq \Pr[M \stackrel{\$}{\leftarrow} A();\, K \stackrel{\$}{\leftarrow} \mathcal{K} :\ M = 0^{m-n} \parallel H_K(c)] \leq \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(C)$ by an argument nearly identical to that given for Case 1-2,.

Pulling together all of the cases yields the following:

$$\begin{aligned}
\mathbf{Adv}_{H5}^{\mathrm{eSec}\,[m]}(A) &= \Pr_K[A \text{ wins by a 1-1 collision}] \Pr_K[\text{1-1 collision}] \\
&\quad + \Pr_K[A \text{ wins by a 2-2 collision}] \Pr_K[\text{2-2 collision}] \\
&\quad + \Pr_K[A \text{ wins by a 1-2 collision}] \Pr_K[\text{1-2 collision}] \\
&\quad + \Pr_K[A \text{ wins by a 2-1 collision} \ \wedge\ \overline{\mathsf{BAD}}] \Pr_K[\text{2-1 collision} \ \wedge\ \overline{\mathsf{BAD}}] \\
&\quad + \Pr_K[A \text{ wins by a 2-1 collision} \ \wedge\ \mathsf{BAD}] \Pr_K[\text{2-1 collision} \ \wedge\ \mathsf{BAD}] \\
&\le 0 + \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(B) \Pr_K[\text{2-2 collision}] + \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(C) \Pr_K[\text{1-2 collision}] \\
&\quad + \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(B) \Pr_K[\text{2-1 collision} \ \wedge\ \overline{\mathsf{BAD}}] \\
&\quad + \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(C) \Pr_K[\text{2-1 collision} \ \wedge\ \mathsf{BAD}] \\
&\le \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(B) + \mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(C)
\end{aligned}$$

where the last inequality is because of convexity. This completes the proof of the claim.

Finally, since the running time of B is $t + \mathrm{Time}_{H,m} + \ell m$ for some absolute constant ℓ, and this is greater than the running time of C, we are done.

B.5 Proof of Proposition 12

Let $H\colon \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$ be the function $G2\colon \{0,1\}^m \times \{0,1\}^m \to \{0,1\}^n$ in Figure 3.

Let T be a collision-finding adversary that on input $K \in \mathcal{K}$ returns the strings $M = K$ and $M = \overline{K}$. Clearly $\mathbf{Adv}_H^{\mathrm{Coll}}(T) = 1$ and T runs in time ℓm for some absolute constant ℓ. It remains to show that $\mathbf{Adv}_H^{\mathrm{eSec}\,[m]}(t) \le 1/2^{m-1}$. Let A be an adversary that runs in time t and gains $\delta = \mathbf{Adv}_H^{\mathrm{eSec}\,[m]}(A)$. Then

$$\begin{aligned}
\delta &= \Pr\left[(M,S) \xleftarrow{\$} A();\, K \xleftarrow{\$} \mathcal{K};\, M' \xleftarrow{\$} A(K,S) :\ (M \ne M') \wedge (H_K(M) = H_K(M'))\right] \\
&\le \Pr\left[(M,S) \xleftarrow{\$} A();\, K \xleftarrow{\$} \mathcal{K} :\ (M = K) \vee (M = \overline{K})\right] \\
&\le 2/2^m
\end{aligned}$$

The first inequality is true because if the adversary does not name a first point M that is either K or $\overline{K}$, then $H_K(M') \ne H_K(M)$ for every $M' \in \{0,1\}^m$. This completes the proof.

B.6 Proof of Theorem 13

Let $H\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ be a hash-function family. Consider $H6\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ defined in Figure 3. We will show that

$$\mathbf{Adv}_{H6}^{\mathrm{Coll}}(t) \le \mathbf{Adv}_{H}^{\mathrm{Coll}}(t') \quad \text{and} \quad \mathbf{Adv}_{H6}^{\mathrm{ePre}}(t') = 1$$

where $t' = t + c\mathrm{Time}_{H,m}$ for some absolute constant c. We begin by showing that $H6$ is trivially breakable in the ePre-sense. Let T be an adversary that on input $K \in \mathcal{K}$ returns 0^m.

Now we show that if H is strong in the Coll-sense, then so is $H6$. Let A be an adversary that gains advantage $\delta = \mathbf{Adv}_{H6}^{\mathrm{Coll}}(A)$ and that runs in time t. We construct an adversary B for finding collisions under H as follows:

Algorithm $B(K)$
  Run $(M, M') \leftarrow A(K)$
  if $M = 0^m$ and $H_K(M') = 0^n$ **then return** (M, M')
  if $M \neq 0^m$ and $H_K(M) \neq 0^n$ and $M' \neq 0^m$ and $H_K(M') = 0^n$ **then return** $(M, 0^m)$
  if $M \neq 0^m$ and $H_K(M) = 0^n$ and $M' = 0^m$ **then return** (M, M')
  if $M \neq 0^m$ and $H_K(M) = 0^n$ and $M' \neq 0^m$ and $H_K(M') \neq 0^n$ **then return** $(0^m, M')$
  else return (M, M')

Note that the running time of B is at most $t + c\mathrm{Time}_{H,m}$ for some absolute constant c.

Let us verify that B returns a collision for H whenever A returns a collision for $H6$ and so $\mathbf{Adv}_{H6}^{\mathrm{Coll}}(A) \leq \mathbf{Adv}_{H}^{\mathrm{Coll}}(B)$. Referring to the line numbers in Figure 3, we say that u-v is a collision if M caused $H6$ to output on line $u \in \{1,2,3\}$ and $M' \neq M$ caused $H6$ to output on line $v \in \{1,2,3\}$ and $H6(M) = H6(M')$. A 1-1 collision is impossible because then $M = M'$, and both a 1-2 collision and a 2-1 collision are impossible because line 2 always returns something different from 0^n. This leaves six cases to consider.

[Case 1-3] Assume A wins by making 1-3 collision. Then we have $M = 0^m$ and $H_K(M') = 0^n$ and so $H_K(0^m) = 0^n$; in this case M' and $0^m = M$ collide under H, and B wins by returning (M, M').

[Case 3-1] Symmetric to case 1-3.

[Case 2-3] Assume A wins by making a 2-3 collision. Then $M \neq 0^m$, $H_K(M) \neq 0^n$, $M' \neq 0^m$, $H_K(M') = 0^n$ and so $H_K(0^m) = H_K(M)$. Hence B wins by returning $(M, 0^m)$.

[Case 3-2] Assume A wins by making a 3-2 collision. Then $M \neq 0^m$, $H_K(M) = 0^n$, $M' \neq 0^m$, $H_K(M') \neq 0^n$ and so $H_K(0^m) = H_K(M')$. Hence B wins by returning $(0^m, M')$.

[Case 2-2] Assume A wins by returning a 2-2 collision. Then $H_K(M) = H_K(M')$ and B wins by returning (M, M').

[Case 3-3] Assume A wins by returning a 3-3 collision. Then $H_K(M) = H_K(M')$ and B wins by returning (M, M').

This completes the proof.

B.7 Proof of Theorem 14

Let $H\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ be a hash-function family. Consider the hash-function family $H6\colon \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ defined in Figure 3. We claim that

$$\mathbf{Adv}_{H6}^{\mathrm{eSec}\,[m]}(t) \leq 2\,\mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(t') \quad \text{and} \quad \mathbf{Adv}_{H6}^{\mathrm{ePre}}(t') = 1$$

where $t' \leq t + c\,\mathrm{Time}_{H,m}$ for some absolute constant c. We begin by showing that $H6$ is trivially breakable in the ePre-sense. Let T be an adversary that on input $K \in \mathcal{K}$ returns 0^m.

Now we show that if H is strong in the eSec-sense then so is $H6$. Let A be an adversary that gains advantage $\delta = \mathbf{Adv}_{H6}^{\mathrm{eSec}\,[m]}(A)$ and runs in time t. We construct an adversary $B0$ as follows:

Algorithm $B0$

[Stage 1] On input ():

Run $(M, S) \leftarrow A()$

(*) **return** (M, S)

[Stage 2] On input (K, S):

Run $M' \leftarrow A(K, S)$

if $M = 0^m$ and $H_K(M') = 0^n$ **then return** M'

if $M \neq 0^m$ and $H_K(M) \neq 0^n$ and $M' \neq 0^m$ and $H_K(M') = 0^n$ **then return** 0^m

if $M \neq 0^m$ and $H_K(M) = 0^n$ and $M' = 0^m$ **then return** 0^m

if $M \neq 0^m$ and $H_K(M) = 0^n$ and $M' \neq 0^m$ and $H_K(M') \neq 0^n$ **then return** M'

else return M'

Let $B1$ be an adversary that is constructed identically to $B0$ except that line (*) is replaced by "**return** $(0^m, S)$".

We claim that whenever A breaks $H6$ in the eSec-sense, then either $B0$ or $B1$ breaks H in the eSec-sense. Referring to the line numbers in Figure 3, we say that u-v is a collision if $M \neq M'$ caused $H6$ to output on line $u \in \{1, 2, 3\}$ and M' caused $H6$ to output on line $v \in \{1, 2, 3\}$ and $H6(M) = H6(M')$. There are six cases to consider, since collisions 1-1, 1-2, and 2-1 are impossible.

[Case 1-3] Assume A wins by making a 1-3 collision. Then $M = 0^m$ and $H_K(M') = 0^n$ and so $H_K(0^m) = 0^n$; in this case M' is a partner for $0^m = M$ under H, and so $B0$ wins

[Case 2-3] Assume A wins by making a 2-3 collision. Then $M \neq 0^m$, $H_K(M) \neq 0^n$, $M' \neq 0^m$ and $H_K(M') = 0^n$. In this case $H_K(M) = H_K(0^m)$, and so $B0$ wins.

[Case 3-1] Assume A wins by making a 3-1 collision. Then $M \neq 0^m$, $H_K(M) = 0^n$ and $M' = 0^n$, and so $H_K(0^m) = 0^n$. In this case $H_K(M) = H_K(0^m)$, and so $B0$ wins.

[Case 3-2] Assume A wins by making a 3-2 collision. Then $M \neq 0^m$, $H_K(M) = 0^n$, $M' \neq 0^m$ and $H_K(M') \neq 0^n$. In this case $H_K(0^m) = H_K(M')$, and so $B1$ wins.

[Case 2-2] Assume A wins by making a 2-2 collision. Then $H_K(M) = H_K(M')$, and so $B0$ wins.

[Case 3-3] Assume A wins by making a 3-3 collision. Then $H_K(M) = H_K(M')$, and so $B0$ wins.

Let $\delta = \delta_0 + \delta_1$ where δ_1 is the probability that A wins (ie, finds a partner for M) by creating a 3-2 collision, and δ_0 is the probability that A wins by creating a 1-3,2-3,3-1,2-2,or 3-3 collision. In the case that $\delta_0 \geq \delta/2$ let $B = B0$; otherwise let $B = B1$. We conclude that $\mathbf{Adv}_{H6}^{\mathrm{eSec}\,[m]}(A) \leq 2\,\mathbf{Adv}_{H}^{\mathrm{eSec}\,[m]}(B)$ and the claim follows.