Properties of Cryptographic Hash Functions

Michal Rjaško

Department of Computer Science, Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava. E-mail: rjasko@dcs.fmph.uniba.sk

Abstract. This paper extends the work of Rogaway and Shrimpton [6], where they formalized seven security properties: notions of preimage resistance (Pre, aPre, ePre), second-preimage resistance (Sec, aSec, eSec) and collision resistance (Coll). They also give all the implications and separations among the properties. In this paper we consider three additional security properties which are important in applications of hash functions: unforgeability (MAC), pseudo-random function (Prf) and pseudo-random oracle (Pro). We give a new type of the implication and separation between the security notions since the ones defined by Rogaway and Shrimpton were too constraining, and work out all the relationships among the ten security notions above. Some of the relations have been proven before, some of them appear to be new. We show that a property pseudo-random oracle (Pro) introduced by Coron, Dodis, Malinaud and Puniya [3] is (as expected) the strongest one, since it implies almost all of the other properties.

Keywords: cryptographic hash function, provable security, properties of hash functions.

1 Introduction

This paper studies the relationships among different security notions of hash function's security. It extends the work of Rogaway and Shrimpton [6], where they define seven basic notions of hash function's security (notions of preimage resistance, second-preimage resistance and collision resistance) and give all implications and separations among these notions. In this paper we consider three additional notions — unforgeability, pseudo-random function and pseudo-random oracle. Thus we have ten different security notions among which we provide all the implications and separations. Some of the relationships have been proven before, some of them appear to be new.

Implication and separation. Rogaway and Shrimpton in [6] defined two types of the implication – conventional and provisional. While the conventional implication carries usual semantics of the word implication, the strength of the provisional implication depends on a particular hash function. They also provide two types of separation – conventional and unconditional, but here is the difference in the fact, whether one can assume the existence of xxx secure hash function to prove the separation between xxx and yyy. However we find the definitions too constraining since they are not suitable for the situation, where we need to simulate some adversary multiple times (for more details see Section 4). Thus we need to somehow generalize the definition of implications and separations by Rogaway and Shrimpton. Our definition of implication (separation) between security notions arises from a definition, when is a hash function secure against attacks in some sense (e.g. collision resistance or preimage resistance). We consider a hash function to be secure in xxx sense if any *efficient* adversary has *negligible* advantage in that sense. By *efficient* adversary we mean an adversary running in a polynomial time. Thus the our definition of the implication informally says that a security notion xxx implies yyy if for a polynomial adversary A

attacking in yyy sense with non-negligible advantage there exists a polynomial adversary B attacking in xxx sense with non-negligible advantage.

Organization. We begin by presenting some basic notation and definitions. In the Section 3 we give ten formal definitions of hash function security — notions of preimage resistance, second-preimage resistance and collision resistance, the unforgeability notion (or MAC), pseudo-random function and pseudo-random oracle. For a more complete discussion about these notions we refer to the papers [6] (preimage, 2nd-preimage and collision resistance), [5] (unforgeability and pseudo-random function), [3] (pseudo-random oracle), [2] or [1] (unforgeability, pseudo-random function, pseudo-random oracle). In the Section 4 we formally define the implication and the separation between two security notions and finally in the Section 6 we give relationships among the definitions that do not appear in [6]. The summary over all relations we provide in the Table 1.

2 Preliminaries

In the formal definitions of hash function security a hash function family is used instead of a hash function. The hash function family is a hash function parametrized by a key. Its is more universal object than a hash function and it enables us to formally define notions, which are hard to define in the settings when using only hash functions. For example it is hard to define collision resistance when we consider only hash functions, since collisions exist in every hash function (its domains is bigger than its range) and trivial adversary can win against any hash function – it just need to have hardwired a colliding pair (however it can be difficult to find such adversary in practice). In hash function family settings, such adversary would need to have hardwired a colliding pair for every key. Note that one can consider a popular hash function SHA1 to be a member of a hash function family, which key is the initialization vector for the SHA1 algorithm.

Definition 1 (Hash function family). *A hash function family is a function $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$, where $\mathcal{K} = \{0,1\}^k$, $\mathcal{Y} = \{0,1\}^y$ for some integers $k, y > 0$ and $\mathcal{M} = \{0,1\}^*$. Set $\mathcal{K}$ is called key space, number y is called hash length of H.*

We write $M \stackrel{\$}{\leftarrow} \mathcal{S}$ for the experiment of choosing random element from the distribution $\mathcal{S}$. If $\mathcal{S}$ is a finite set, then M is chosen uniformly from $\mathcal{S}$. Concatenation of finite strings M_1 and M_2 we denote by $M_1||M_2$ or simply M_1M_2. Bitwise complement of string M we write as $\overline{M}$. Empty string is denoted by μ. If i is an integer, then $\langle i \rangle_r$ is r-bit string representation of i. Let $Func(D,R)$ represent the set of all functions $\rho : D \to R$ and let $RF_{D,R}$ be a function chosen randomly from the set $Func(D,R)$ (i.e. $RF_{D,R} \stackrel{\$}{\leftarrow} Func(D,R)$). We sometimes write $RF_{d,r}$ when $D = \{0,1\}^d$ and $R = \{0,1\}^r$. By $\text{Prefix}_n(M)$ we denote the n-bit prefix of message M, similarly by $\text{Suffix}_n(M)$ we denote the n-bit suffix of M.

Definition 2 (Adversary). *An adversary is a random access machine (RAM) with any number of inputs (i.e. it can access ith bit of input j in unit time) that can toss a coin in unit time (i.e. it can choose a sample from the set $\{0,1\}$ in a unit time). Running time of an adversary A on some input is the average time needed to compute an output (relative to some fixed RAM model) plus the description size of A (relative to some fixed coding of RAMs).*

Let $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash function family. We denote by $\text{Time}_{H,n}$ the running time of an algorithm P (i.e. some random access machine) computing H that has the best worst case running time over all inputs $(K, M); K \in \mathcal{K}; M \in \mathcal{M}; |M| = n$, that is, any other algorithm P' computing H has the worst case running time over all the inputs $(K, M); K \in \mathcal{K}; M \in \mathcal{M}; |M| = n$ greater or equal to P's. Informally speaking, $\mathrm{Time}_{H,n}$ is the time needed to compute H_K on any input of length n.

A function $f : \mathsf{N} \to \mathsf{R}^+$ is negligible, if it descends faster than any polynomial powered to -1. The formal definition is following.

Definition 3 (Negligible function). *A function $f : \mathsf{N} \to \mathsf{R}^+$ is negligible, if for every constant $c > 0$, there exists an integer $N_0 \in \mathsf{N}$, such that for all integers $n > N_0$ it holds*

$$f(n) < \frac{1}{n^c}.$$

The term negligible we mostly use when considering an advantage of an adversary attacking a hash function family $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$. We consider some advantage as negligible when it is a function of k or y and this function is negligible.

3 Definitions of the security notions

Standard notions. The seven security notions defined below are those from Rogaway-Shrimpton [6] – notions of collision resistance, second-preimage resistance and preimage resistance. We note, that the parameter $[\lambda]$ is used in the definitions to avoid random selection from the possibly infinite set $\mathcal{M}$ and also to bound the length of randomly selected messages.

Let $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash function family and let λ be a number such that $\{0,1\}^\lambda \subseteq \mathcal{M}$. Let A be an adversary. Then we define the following advantage measures:

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

We say that H is (t, L, ε)-xxx for xxx $\in \{\mathrm{Pre}, \mathrm{aPre}, \mathrm{Sec}, \mathrm{eSec}, \mathrm{aSec}\}$ if any adversary A running in time at most t and outputting messages of length less than or equal to L has advantage $\mathbf{Adv}_H^{\mathrm{xxx}[\lambda]}(A) \leq \varepsilon$ for all positive integers λ. We say that H is (t, L, ε)-yyy for yyy $\in$ {ePre, Coll}, if any adversary A running in time at most t and outputting messages of length less than or equal to L has advantage $\mathbf{Adv}_H^{\mathrm{yyy}}(A) \leq \varepsilon$.

Message authentication codes. Hash function families used for message authentication should be strong in the following sense. Let $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash function family. Let A be an adversary. Consider the following advantage measure:

$$\mathbf{Adv}_H^{\mathrm{MAC}}(A) = \Pr\left[K \stackrel{\$}{\leftarrow} \mathcal{K}; (M, Y) \leftarrow A^{H_K} : H_K(M) = Y \wedge M \text{ not queried}\right]$$

We say that a hash function family H is (t, q, L, ε)-MAC if any adversary A running in time at most t, outputting or querying messages of length less than or equal to L and making at most q queries to its oracle has advantage $\mathbf{Adv}_H^{\mathrm{MAC}}(A) \le \varepsilon$.

Note that the adversary A in the advantage measure above does not have access to the key K. It takes function $H_K : \mathcal{M} \to \mathcal{Y}$ as a black-box and can not output message, that was queried. Otherwise it would be easy to find such adversary for every function family H (it would query some message M and return a pair $(M, H_K(M))$).

Also note that it makes no sense to think about maximizing the MAC advantage over all K (i.e. defining always MAC), as for a given function $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ we can construct an adversary A always returning pair $(M, H_{K_0}(M))$ for some fixed K_0. The advantage of such adversary, if the key K_0 is chosen, is 1, thus if we maximize the advantage over all keys, it can not be smaller than 1.

Pseudo-random function and oracle. Hash functions are often used as a basic primitive, from which more complex cryptosystems are build. It can be very difficult, even impossible to prove the security of a cryptosystem that uses some complicated hash function. To simplify the task, one first proves that the cryptosystem is secure when we replace the hash function with some idealized object, i.e. random oracle. Then one proves that the hash function is "indistinguishable" from that idealized object and therefore the security of the system is not affected when the idealized object is replaced with the hash function.

Two concepts of the "indistinguishability" have been defined so far – *indistinguishability* and more general *indifferentiability* defined in [4]. We consider a hash function family H to be a pseudo-random function (Prf) when it is indistinguishable from the random oracle. Similarly, a hash function family is pseudo-random oracle (Pro), if it is indifferentiable from the random oracle.

The formal definitions of Prf and Pro arise from the definitions of indistinguishability and indifferentiability ([4]). The definition of Pro requires the hash function family to be built from some smaller "compression" function $f : \{0,1\}^{y+1} \to \{0,1\}^y$, i.e. we can consider a hash function family to be some domain extension over the compression function f.

Let $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash function family. Let A be an adversary, $f = RF_{y+d,y}$ for some integer $d > 0$ (f represents an ideal compression function) and let $\mathcal{S}$ be a simulator (the simulator $\mathcal{S}$ is an algorithm (i.e. a RAM), which simulates f to make distinguishing more difficult). Then we define the following advantage measures:

$$\mathbf{Adv}_H^{\mathrm{Prf}}(A) = \left| \Pr\left[K \xleftarrow{\$} \mathcal{K}; 1 \leftarrow A^{H_K(\cdot)}\right] - \Pr\left[\mathcal{F} \xleftarrow{\$} Func(\mathcal{M}, \mathcal{Y}); 1 \leftarrow A^{\mathcal{F}}\right] \right|$$

$$\begin{aligned}\mathbf{Adv}_{H,f,\mathcal{S}}^{\mathrm{Pro}}(A) = \Big| & \Pr\left[K \xleftarrow{\$} \mathcal{K}; 1 \leftarrow A^{H_K^f(\cdot), f(\cdot)}(K)\right] - \\ & - \Pr\left[K \xleftarrow{\$} \mathcal{K}; \mathcal{F} \xleftarrow{\$} Func(\mathcal{M}, \mathcal{Y}); 1 \leftarrow A^{\mathcal{F}(\cdot), \mathcal{S}^{\mathcal{F}}(K, \cdot)}(K)\right] \Big|\end{aligned}$$

We say that H is (t, q, L, ε)-Prf if any adversary A running in time at most t and making at most q queries to its oracle each of length less than or equal to L has advantage $\mathbf{Adv}_H^{\mathrm{Prf}}(A) \le \varepsilon$. We say that H is $(t_A, t_{\mathcal{S}}, q_1, q_2, L, \varepsilon)$-Pro if for any adversary A running in time at most t_A and making at most q_1 (q_2) queries to its first (second) oracle each of length less than or equal to L, there exists a simulator $\mathcal{S}$ running in time $t_{\mathcal{S}}$ such that the advantage $\mathbf{Adv}_{H,f,\mathcal{S}}^{\mathrm{Pro}}(A) \le \varepsilon$.

Again, it makes no sense to maximize the advantages A^{Prf} and A^{Pro} over all keys, since then a trivial adversary would prevail. Note that no adversary can have advantage equal to 1 in the senses above, since there is a probability, that the randomly selected function $\mathcal{F}$ equals to some function H_K from the family H.

4 Implication and separation

We consider a hash function family H to be secure in some sense (Pro, Prf, MAC, Sec,...), if any polynomial adversary has negligible advantage against $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ in that sense. Polynomial adversary runs in a time that is a polynomial of k, y and l, where l is the length of its input (if it has some).

Informally, we say that a security notion xxx implies security notion yyy, when for all hash function families H holds, that if H is secure in the xxx sense, then it is secure in yyy sense. Our formal definition of the implication comes straightly from this intuition. We note that in the following definition, and later, $[\cdot]$ is a placeholder which is either $[\lambda]$ (for Pre, aPre, Sec, aSec, eSec, CTFP, aCTFP) or empty (for ePre, Coll, Prf, Pro). We also write $\mathbf{Adv}_{H,\cdot,\cdot}^{\mathrm{xxx}[\cdot]}$, which is either $\mathbf{Adv}_{H,\mathcal{S},f}^{\mathrm{xxx}[\cdot]}$ (when xxx is Pro), or $\mathbf{Adv}_{H,f}^{\mathrm{xxx}[\cdot]}$ (when xxx is something else than Pro, but we are comparing it to Pro (e.g. yyy is Pro)), or $\mathbf{Adv}_{H}^{\mathrm{xxx}[\cdot]}$ (when both security notions xxx and yyy are different from Pro).

Definition 4 (xxx $\to$ yyy). *Let $\mathcal{K} = \{0,1\}^k$, $\mathcal{M} = \{0,1\}^*$ and $\mathcal{Y} = \{0,1\}^y$ for some fixed k and y, let λ be some fixed positive integer, $f = RF_{y+d,y}$ and suppose, that xxx, yyy $\in$ Atks. We say that the definition of security notion xxx implies security notion yyy (shortly xxx $\to$ yyy), if for any hash function family $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ and any adversary A running in polynomial time t, with non-negligible advantage (with respect to k, y or λ) in yyy sense (for all polynomial simulators $\mathcal{S}$ if yyy is Pro), there exists an adversary A' such that A' runs in polynomial time t' and has non-negligible advantage in xxx sense (for all polynomial simulators $\mathcal{S}$ if xxx is Pro).*

Note that the definition above is different from the Rogaway-Shrimpton's one [6]. We find their definition too constraining, since it does not allow to construct adversary A' by simulating A several times. For example consider the definition of provisional implication from [6] (we note that by $\mathbf{Adv}_{H}^{\mathrm{xxx}}(t)$ we denote the maximum advantage against a hash function family H in xxx sense over all adversaries running in time at most t):

> Fix $\mathcal{K}$, $\mathcal{M}$, m and n. Suppose that xxx and yyy are some notions for hash function security. We say that xxx implies yyy to ε, if $\mathbf{Adv}_{H}^{\mathrm{yyy}}(t) \leq c\,\mathbf{Adv}_{H}^{\mathrm{xxx}}(t') + \varepsilon$ for all hash function families $H : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$ where c is an absolute constant and $t' = t + c\,\mathrm{Time}_{H,m}$.

Now consider that we have two security notions xxx and yyy about which we prove that any adversary A attacking in yyy sense with advantage ε can be converted into an adversary A' attacking in xxx sense with advantage ε^2 (i.e. A' simulates A twice and wins when A wins in both simulations). Our intuition says, that such security notion xxx should imply yyy, however the definition above restricts the running time of A' to be only constantly greater than the running time of A (i.e. if we want to prove xxx implies yyy with respect to the definition above, A' does not have enough time to simulate A twice), and it also restricts the advantage of A' to be the constant multiple of the A's advantage, what is not the our case. Thus some generalization of the Rogaway-Shrimpton's definition was needed, what lead to the our definition above.

We note that the definition of the implication by Rogaway and Shrimpton [6] implies our definition. Thus all the proofs of implications from [6] holds also with the respect to the our definition of the implication.

Intuitively, a security notion xxx does not imply yyy, if there exists a hash function family H, which is secure in xxx sense, but insecure in yyy sense. We have two possibilities how to formalize this idea. The difference is in the condition, whether one can assume the existence of the xxx-secure hash function family or not. The first definition, *conventional separation* informally says, that xxx non-implies yyy if for any xxx-secure hash function family H we can construct a hash function family H', that is secure in xxx sense, but insecure in yyy sense. We note that the following definition generalizes the definition of conventional separation from [6] in the same way as in the case of implications.

Definition 5 (xxx $\not\to$ yyy). *Let $\mathcal{K} = \{0,1\}^k$, $\mathcal{M} = \{0,1\}^*$ and $\mathcal{Y} = \{0,1\}^y$ for some fixed k and y, let λ be some fixed positive integer, $f = RF_{y+d,y}$ and suppose that xxx, yyy $\in$ Atks. We say that the definition of security notion xxx non-implies security notion yyy (shortly xxx $\not\to$ yyy), if for any hash function family $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ there exists a hash function family $H' : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$, such that if $\mathbf{Adv}^{xxx[\cdot]}_{H',\cdot,\cdot}(t)$ is non-negligible (for all polynomial simulators $\mathcal{S}$ if xxx is Pro), then so is $\mathbf{Adv}^{xxx[\cdot]}_{H,\cdot,\cdot}(t')$ (for all polynomial simulators $\mathcal{S}$ if xxx is Pro), and $\mathbf{Adv}^{yyy[\cdot]}_{H',\cdot,\cdot}(t')$ is non-negligible too (for all polynomial simulators $\mathcal{S}$ if yyy is Pro), where t and t' are some polynomial running times.*

On the other hand, unconditional separation between security notions xxx and yyy does not need to assume the existence of a xxx secure hash function family. It says, that xxx non-implies yyy in unconditional sense, if there exists a hash function family H, which is secure in xxx sense, but insecure in yyy sense.

Definition 6 (xxx $\not\rightharpoonup$ yyy). *Let $\mathcal{K} = \{0,1\}^k$, $\mathcal{M} = \{0,1\}^*$ and $\mathcal{Y} = \{0,1\}^y$ for some fixed k and y, let λ be some fixed positive integer, $f = RF_{y+d,y}$ and suppose that xxx, yyy $\in$ Atks. We say that the definition of security notion xxx non-implies security notion yyy in the unconditional sense (shortly xxx $\not\rightharpoonup$ yyy), if there exists a hash function family $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$, such that $\mathbf{Adv}^{xxx[\cdot]}_{H',\cdot,\cdot}(t)$ is negligible for all polynomial t, and $\mathbf{Adv}^{yyy[\cdot]}_{H,\cdot,\cdot}(t')$ is non-negligible for some polynomial time t'.*

We note that in this paper, no unconditional separation is proved. An unconditional separation can be consequence of the matter, that for some domains and ranges secure hash functions trivially exist, for example identity function $H_K(M) = M$ is trivially collision resistant, but it is definitely not preimage resistant. However for standard domains and ranges (e.g. $\mathcal{M} = \{0,1\}^*$ and $\mathcal{Y} = \{0,1\}^y$), it is very hard to prove the unconditional separation xxx $\not\rightharpoonup$ yyy, since one has to find a hash function family secure in xxx sense (what would be a very important discovery).

5 Equivalent definitions with a two stage adversary

In the definitions of aPre, ePre, aSec, eSec, aCTFP we maximize over some quantity (over all keys or messages). However, there exist equivalent definitions to these already mentioned, where the specific value (key or message) is chosen by an adversary. That is, in the "first phase" the adversary choses that value, then a random choice is made by the environment and in the "second phase" the adversary continues, where it ended, but with given that randomly chosen value. The corresponding definitions are as follows.

Let $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash function family, and let λ be a number such that $\{0,1\}^\lambda \subseteq \mathcal{M}$. Let A be an adversary. Then we define:

$$\mathbf{Adv}^{\mathrm{aPre}[\lambda]}_{H}(A) = \Pr\left[(K,S) \leftarrow A; M \xleftarrow{\$} \{0,1\}^\lambda; Y \leftarrow H_K(M); M' \leftarrow A(Y,S) : H_K(M') = H_K(M)\right]$$

$$\mathbf{Adv}^{\mathrm{ePre}}_{H}(A) = \Pr\left[(Y,S) \leftarrow A; K \xleftarrow{\$} \mathcal{K}; M' \leftarrow A(K,S) : H_K(M') = Y\right]$$

$$\mathbf{Adv}^{\mathrm{aSec}[\lambda]}_{H}(A) = \Pr\left[(K,S) \leftarrow A; M \xleftarrow{\$} \{0,1\}^\lambda; M' \leftarrow A(M,S) : (M \neq M') \wedge (H_K(M) = H_K(M'))\right]$$

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

The proof of the equivalence between two-stage versions and normal versions of the definition is quite straightforward and its sketch can be found in [6].

6 Relationships among the definitions

Here we provide the relationships (implication or separation) among the definitions of security notions. Rogaway and Shrimpton [6] proved the relationships among notions of preimage resistance, second-preimage resistance and collision resistance, relationship between MAC and Prf were mentioned in [5]. Other relationships appear to be new.

We give an overview over all of the relations in the Table 1. In the Figure 1 we provide all the constructions used in the proofs of separations.

	Pre	aPre	ePre	Sec	aSec	eSec	Coll	MAC	Prf	Pro
Pre	x	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ 2	$\not\to$ 4	$\not\to$ 7
aPre	$\to$ [6]	x	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ 2	$\not\to$ 4	$\not\to$ 7
ePre	$\to$ [6]	$\not\to$ [6]	x	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ 2	$\not\to$ 4	$\not\to$ 7
Sec	$\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	x	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ 2	$\not\to$ 4	$\not\to$ 7
aSec	$\to$ [6]	$\to$ [6]	$\not\to$ [6]	$\to$ [6]	x	$\not\to$ [6]	$\not\to$ [6]	$\not\to$ 2	$\not\to$ 4	$\not\to$ 7
eSec	$\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\to$ [6]	$\not\to$ [6]	x	$\not\to$ [6]	$\not\to$ 2	$\not\to$ 4	$\not\to$ 7
Coll	$\to$ [6]	$\not\to$ [6]	$\not\to$ [6]	$\to$ [6]	$\not\to$ [6]	$\to$ [6]	x	$\not\to$ 2	$\not\to$ 4	$\not\to$ 7
Mac	$\not\to$ 1	$\not\to$ 1	$\not\to$ 1	$\not\to$ 1	$\not\to$ 1	$\not\to$ 1	$\not\to$ 1	x	$\not\to$ 4	$\not\to$ 7
Prf	$\not\to$ 3	$\not\to$ 3	$\not\to$ 3	$\not\to$ 3	$\not\to$ 3	$\not\to$ 3	$\not\to$ 3	$\to$ 3	x	$\not\to$ 7
Pro	$\to$ 5	$\not\to$ 6	$\to$ 5	$\to$ 5	$\not\to$ 6	$\to$ 5	$\to$ 5	$\to$ 5	$\to$ 5	x

Table 1. Relationships among the definitions. Numbers in brackets [·] are citations, other numbers are numbers of theorems, where the proof of the corresponding relation can be found.

$$H_K^{(1)}(M) = Y[0 \cdots y-1]||0 \qquad \text{if } H_K(M) = Y$$

$$H_K^{(2)}(M) = \begin{cases} 0^y & \text{if } M = 0 \\ H_K(M) & \text{if } M \neq 0 \text{ and } H_K(M) \neq 0^y \\ H_K(0) & \text{otherwise} \end{cases}$$

$$H_K^{(3)}(M) = \begin{cases} Y & \text{if } \mathrm{Prefix}_{(k+1+y)}(M) = K||b||Y \text{ for some } b \in \{0,1\} \\ H_K(M) & \text{otherwise} \end{cases}$$

$$H_K^{(4)}(M) = H_K(M[1 \ldots |M|-1]||0)$$

$$H_K^{(5)}(M) = \begin{cases} H_K(M) & \text{if } K \neq K_0 \\ 0^y & \text{if } K = K_0 \end{cases}$$

Fig. 1. Constructions of hash function families used in proofs of separations. The constructions $H^{(2)}$, $H^{(4)}$ and $H^{(5)}$ come from the work [6]

6.1 MAC

An adversary attacking in MAC sense does not have access to the key K, which is chosen randomly by the environment. However adversaries attacking in xPre, xSec, Coll senses have that access. Thus if we use the key K to bundle some information, which makes finding preimages or collisions easier, adversary in MAC sense cannot find that information, what is the idea the behind following separations.

Theorem 1 (MAC 1).

(1) MAC $\not\to$ Coll
(2) MAC $\not\to$ Sec, aSec, eSec
(3) MAC $\not\to$ Pre, aPre, ePre

Proof. Let $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ be a hash function family and consider the construction $H^{(3)}$ from the Figure 1. Let A be an adversary running in polynomial time t with non-negligible advantage $\mathbf{Adv}^{MAC}_{H^{(3)}}(A)$. We construct an adversary A', which simulates A and outputs the same as A. Since A has no access to the key K chosen randomly be the environment, the probability that it queries a message of length $(k+1+y)$ with prefix K is at most $\frac{q}{|\mathcal{K}|}$, what is the maximum probability that A'^{H} returns different output that $A^{H^{(3)}}$. Since A runs in a polynomial time, it can make at most polynomial number of queries. Therefore $\frac{q}{|\mathcal{K}|}$ is negligible and thus $\mathbf{Adv}^{MAC}_{H}(A')$ is non-negligible, what completes the first part of the proof.

One can easily see, that $H^{(3)}$ is not Coll secure, since $H^{(3)}_K(K||1||Y) = H^{(3)}_K(K||0||Y)$ for every key $K \in \mathcal{K}$ and arbitrary image Y. For the same reason it is not Sec, aSec and eSec secure. Finally, $H^{(3)}$ is not Pre, aPre, ePre secure, as for a given image Y we can construct a message $M = K||1||Y$ and $H_K(M) = Y$.

Theorem 2 (MAC 2).

(1) Coll $\not\to$ MAC
(2) Sec, aSec, eSec $\not\to$ MAC
(3) Pre, aPre, ePre $\not\to$ MAC

Proof.
(1),(2) We use the construction $H^{(2)}$. In Rogaway-Shrimpton [6] one can find the proof that if H is secure in Coll (Sec, aSec, eSec) sense, then so is $H^{(2)}$, what is the first part of the proof. The second part is easy — $H^{(2)}$ is clearly not MAC secure, since $H^{(2)}_K(0) = 0^y$ for all keys $K \in \mathcal{K}$.

(3) The separation comes from the construction $H^{(4)}$, which is clearly not MAC secure, since an adversary B, which first queries arbitrary message M, gets output $H_K(M)$ and then returns the pair $(M', H_K(M))$, where M' equals to M but with the last bit inverted, has advantage 1 in MAC sense against $H^{(4)}$. On the other hand, from a polynomial adversary A that attacks $H^{(4)}$ in Pre (aPre, ePre) sense with non-negligible advantage we can construct an adversary A' as follows:

Adversary $A'(Y, K)$
  $M \leftarrow A(Y, K)$
  if $H_K(M) = Y$ **then return** M
  else let $b := M[|M|]$; **return** $M[1 \ldots |M|-1]||\overline{b}$

It is clear that if A returns correct preimage, then so does A', thus if A has non-negligible probability of success, then so has A'.

6.2 Prf

Similarly to the MAC notion, adversaries attacking in Prf sense do not have access to the key. Therefore, to prove the separations between Prf and Coll, xSec or xPre, we can use the same idea as we did in the MAC case.

Theorem 3 **(Prf 1).**

(1) Prf $\not\to$ Coll
(2) Prf $\not\to$ Sec, aSec, eSec
(3) Prf $\not\to$ Pre, aPre, ePre
(4) Prf $\to$ MAC

Proof.
(1),(2),(3) We follow the steps of the proof of the Theorem 1. Only difference is in the part (a) where we need to note, that the second part $\Pr[\mathcal{F} \stackrel{\$}{\leftarrow} Func(\mathcal{M}, \mathcal{Y}); 1 \leftarrow A^{\mathcal{F}}]$ of the advantage in Prf sense is the same for both advantages $\mathbf{Adv}^{\text{Prf}}_{H^{(4)}}(A)$ and $\mathbf{Adv}^{\text{Prf}}_{H}(A')$. Therefore we need to consider only the first part $\Pr[K \stackrel{\$}{\leftarrow} \mathcal{K}; 1 \leftarrow A^{H_K(\cdot)}]$ of the advantage in Prf sense. However A'^{H_K} returns different output than $A^{H_K^{(4)}}$ only when A' queries a message of length $(k + 1 + y)$ with prefix K. The probability that such case happens is at most $\frac{q}{|\mathcal{K}|}$, therefore if A's advantage is non-negligible, then so is the advantage of A'. Other parts of the proof are nearly identical to those in the proof of the Theorem 1.

(4) This implication comes from the fact, that an adversary attacking in Prf sense can simulate an adversary attacking in MAC sense and verify, whether its output is correct. If so, then it returns 1, otherwise it returns 0. Therefore if we have an adversary A attacking an arbitrary hash function family H in MAC sense with non-negligible probability, we can construct (as described above) an adversary A', which has non-negligible probability against H in Prf sense. Note that the second part of Prf-advantage $\Pr[\mathcal{F} \stackrel{\$}{\leftarrow} Func(\mathcal{M}, \mathcal{Y}); 1 \leftarrow A'^{\mathcal{F}}]$ for the adversary A' is negligible, since the probability that A wins against a random function is negligible.

Theorem 4 **(Prf 2).**

(1) Coll $\not\to$ Prf
(2) Sec, aSec, eSec $\not\to$ Prf
(3) Pre, aPre, ePre $\not\to$ Prf
(4) MAC $\not\to$ Prf

Proof.
(1),(2),(3) These separations comes from the fact proven before in the Theorem 3 part (4), that if a hash function family is insecure in MAC sense, then it is insecure in Prf sense too. Therefore if the separations from the Theorem 2 hold, then also these separations hold.

(4) Consider the construction $H^{(1)}$ from the Figure 1 and an adversary A that has non-negligible advantage against $H^{(1)}$ in MAC sense. Let A' be the following adversary:

Adversary A'^f
  run $(M, Y) \leftarrow A^f$
  let $b \stackrel{\$}{\leftarrow} \{0, 1\}$
  return $(M, Y[1 \ldots y] || b)$

One can see, that if A returns the pair (M, Y) such that $H_K^{(1)}(M) = Y$, then A' returns a pair (M, Y'), where $H_K(M) = Y'$ with probability $\frac{1}{2}$. Therefore the advantage in MAC sense of the adversary A' against H is equal to the half of the advantage of A against $H^{(1)}$ in MAC sense. Thus A' has non-negligible advantage in MAC sense against H. On the other hand, $H^{(1)}$ is clearly not Prf secure, since adversary attacking in Prf sense can verify by querying some number of messages, whether its oracle returns images with the last bit always equal to 0. If so, it returns 1, otherwise 0.

6.3 Pro

When a hash function family is Pro secure, then it is indifferentiable from a random oracle and it is hard (effectively unfeasible) to win in any kind of presented attacks against the random oracle. Therefore Pro is expected to be the strongest property.

In the following text we will assume, that a hash function family $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ is build from some ideal compression function $f : \{0,1\}^{y+d} \to \mathcal{Y}; d > 0$ and an algorithm computing H has oracle access to f. For that reason we need to give the oracle access to f also to adversaries attacking in non-Pro sense (i.e. in Pre, aPre, ePre, Sec, aSec, eSec, Coll, MAC and Prf), adversaries attacking in Pro sense already have such access. For example, the advantage in Pre sense of an adversary A would look like follows:

$$\mathbf{Adv}_{H,f}^{\mathrm{Pre}}(A) = \Pr\left[K \xleftarrow{\$} \mathcal{K}; M \xleftarrow{\$} \mathcal{M}; Y \leftarrow H_K(M); M' \leftarrow A^f(K, Y) : H_K^f(M') = Y\right]$$

Advantages in other senses are modified similarly. We omit writing H^f (we write only H), since all the hash functions used in the following text has oracle access to f.

Theorem 5 (Pro 1).

(1) Pro $\to$ Pre, ePre
(2) Pro $\to$ Sec, eSec
(3) Pro $\to$ Coll
(4) Pro $\to$ MAC
(5) Pro $\to$ Prf

Proof. All these implications above are based on the fact, that adversaries attacking in Pro sense have access to the key K chosen randomly by the environment and therefore they can simulate any other adversary attacking in xPre, xSec, Coll, MAC or Prf sense and return 1 if and only if the simulated adversary returns correct output. Let xxx represent one of the security notions Pre, ePre, Sec, eSec, Coll, MAC or Prf. Let $f = RF_{y+d,y}$ be a random function and let A be a polynomial adversary attacking a hash function family H in xxx sense and let B be the following adversary performing attack in Pro sense:

Adversary $B^{f_1,f_2}(K)$
generate random input for A
simulate A^{f_2} with that input
by querying oracle f_1 verify, whether A^{f_2} returned correct output
return 1 if the output is correct
return 0 otherwise

Clearly B runs in polynomial time, if A runs in polynomial time. Suppose, that $\mathbf{Adv}_{H,f}^{xxx}(A) = \varepsilon$ is non-negligible. From the description of B it is clear that the first part of B's Pro advantage $\Pr[K \xleftarrow{\$} \mathcal{K}; 1 \leftarrow B^{H_K^f(\cdot), f(\cdot)}(K)] = \varepsilon$. The second part of the advantage

$$\Pr[K \xleftarrow{\$} \mathcal{K}; \mathcal{F} \xleftarrow{\$} Func(\mathcal{M}, \mathcal{Y}); 1 \leftarrow B^{\mathcal{F}(\cdot), \mathcal{S}^{\mathcal{F}}(K, \cdot)}(K)] \tag{1}$$

utilizes a random function $\mathcal{F}$ and a simulator with oracle access to that function $\mathcal{S}^{\mathcal{F}}$. However any polynomial adversary performing attack against a random function is doomed to fail thus this second part of the B's advantage seems to be negligible, only problem can be the simulator $\mathcal{S}^{\mathcal{F}}$, which can somehow assist in attacking $\mathcal{F}$ (and therefore raise the probability, that $B^{\mathcal{F},\mathcal{S}^{\mathcal{F}}}$ returns 1). On the other hand, both A and $\mathcal{S}$ run in a polynomial time, therefore at most polynomial number of queries can be made by $\mathcal{S}$ to $\mathcal{F}$ during the attack and therefore $A^{\mathcal{S}^{\mathcal{F}}}$ can succeed only with negligible probability (since $\mathcal{F}$ is a random function). Therefore B has non-negligible advantage in Pro sense, what completes the proof.

Theorem 6 (Pro 2).

(1) Pro $\not\to$ aPre
(2) Pro $\not\to$ aSec

Proof. Consider the construction $H^{(5)}$ and an adversary A with non-negligible advantage $\mathbf{Adv}^{\text{Pro}}_{H^{(5)}}(A)$. Since $\frac{1}{|\mathcal{K}|}$ is negligible, adversary A has also non-negligible advantage in Pro sense against H ($\frac{1}{|\mathcal{K}|}$ is the probability, that the key K_0 is chosen by the environment, what is the only case when A attacking H can notice some difference from the case it attacks $H^{(5)}$). On the other hand $H^{(5)}$ is clearly not aPre or aSec secure.

Theorem 7 (Pro 3).

(1) Pre, aPre, ePre $\not\to$ Pro
(2) Sec, aSec, eSec $\not\to$ Pro
(3) Coll $\not\to$ Pro
(4) MAC $\not\to$ Pro
(5) Prf $\not\to$ Pro

Proof. The separations (1),(2),(3),(4) comes from the fact that the same separations hold for Prf on the right side (see the Theorem 4) and if some hash function family is not Prf secure, then it is definitely not Pro secure (as Pro implies Prf). So the only separation left is (5). For this separation we use the construction $H^{(3)}$, about which we have already proved, that it is Prf secure, if H is. However it is easy to find an adversary, which attacks $H^{(3)}$ with non-negligible advantage in Pro sense:

Adversary $C^{f_1,f_2}(K)$
    if $f_1(K\|\|0\|\|0^y) = f_1(K\|\|1\|\|0^y) = 0^y$ **then return** 1
    otherwise return 0

7 Conclusion

In this paper we analyzed the relationships among different notions of hash function security. We extended the work of Rogaway and Shrimpton [6] by adding three more notions (MAC, Prf and Pro) and proving all the implication and separations among MAC, Prf, Pro and the notions of preimage resistance, second-preimage resistance and collision resistance. We also defined the new type of the implication and separation between two security notions, since the ones from [6] were too constraining. The summary of the relationships can be found in the Table 1.

Our results indicate that the Pseudo-random oracle (Pro) notion is the strongest one (as was expected), since it implies almost all of the other notions. However Pro requires a hash function family to be built from some kind of compression function, therefore we rather speak of Pseudo-random oracle as a domain extension transform, which extends the domain of a "small" compression function to the "big" hash function family. One can also speak about Pseudo-random oracle preserving domain extension transform (Pro-Pr), which transforms an ideal compression function to a hash function family, which is secure in Pro sense. As Bellare and Ristenpart [2] showed, if a Pro-Pr domain extension transform is applied to a non-ideal compression function f, it can actually "weaken" the security properties of a resulting hash function family, that is if f is a collision-free compression function, then the resulting hash function family need not to be collision resistant (i.e. Coll secure). On the other hand, we showed that when and ideal compression function is used, then the resulting hash function family must be Coll secure. Thus Pro-Pr domain extension transforms gives us beliefs, that there are no structural flaws in the construction and when used with well selected compression function, it can be sufficiently "secure" (especially, when the domain extension is besides Pro-Pr also Coll-Pr, Pre-Pr etc.).

Acknowledgments

This paper was supported by VEGA grant no. 1/3106/06.

References

1. M. Bellare and T. Ristenpart. Hash Functions in the Dedicated-Key Setting: Design Choices and MPP Transforms. In *International Colloquim on Automata, Languages, and Progamming, LNCS vol. 4596*, pages 399–410. Springer, 2006.
2. M. Bellare and T. Ristenpart. Multi-Property-Preserving Hash Domain Extension and the EMD Transform. In *Advances in Cryptology - ASIACRYPT 2006, LNCS vol. 4284*, pages 299–314. Springer, 2006.
3. J.S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Damgård Revisited: How to Construct a Hash Function. In *Advances in Cryptology – CRYPTO 2005, LNCS vol. 3621*, pages 430–448. Springer, 2005.
4. U. Maurer, R. Renner, and C. Holenstein. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In *Theory of Cryptography, LNCS vol. 2951*, pages 21–39. Springer, 2004.
5. M. Naor and O. Reingold. From Unpredictability to Indistinguishability: A Simple Construction of PseudoRandom Functions from MACs. In *Advances in Cryptology – CRYPTO '98, LNCS vol. 1462*, pages 267–281. Springer, 1998.
6. P. Rogaway and T. Shrimpton. Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In *Fast Software Encryption, LNCS vol. 3017*, pages 371–388. Springer, 2004.