Non-Trivial Black-Box Combiners for Collision-Resistant Hash-Functions don't Exist

Krzysztof Pietrzak[⋆]

CWI Amsterdam
pietrzak@cwi.nl

Abstract. A (k,ℓ)-robust combiner for collision-resistant hash-functions is a construction which from ℓ hash-functions constructs a hash-function which is collision-resistant if at least k of the components are collision-resistant. One trivially gets a (k,ℓ)-robust combiner by concatenating the output of any $\ell-k+1$ of the components, unfortunately this is not very practical as the length of the output of the combiner is quite large. We show that this is unavoidable as no black-box (k,ℓ)-robust combiner whose output is significantly shorter than what can be achieved by concatenation exists. This answers a question of Boneh and Boyen (Crypto'06).

1 Introduction

A function $H:\{0,1\}^*\rightarrow\{0,1\}^v$ is a collision-resistant hash-function (CRHF), if no efficient algorithm can find two inputs $M\neq M'$ where $H(M)=H(M')$, such a pair (M,M') is called a collision for H.[1]

In the last few years we saw several attacks on popular CRHFs previously believed to be secure [18, 19]. Although provably secure[2] hash-functions exist (see e.g. [3] and references therein), they are rather inefficient and rarely used in practice. As we do not know which of the CRHFs used today will stay secure, it is natural to investigate combiners for CRHFs. In its simplest form the problem is the following: given two hash-functions

$$H_1,H_2:\{0,1\}^*\rightarrow\{0,1\}^v,$$

can we construct a new hash-function which is collision-resistant if either H_1 or H_2 is? The answer is that of course we can, just concatenate the outputs:

$$H(X)=H_1(X)\|H_2(X). \tag{1}$$

[⋆] Supported by DIAMANT, the Dutch national mathematics cluster for discrete interactive and algorithmic algebra and number theory. This work was partially done while the author was a postdoc at the Ecole Normale Supérieure, Paris.

[1] This definition is very informal as there are some issues which make it hard to have a definition for collision-resistant hash-functions which is theoretically and practically satisfying, see [15] for recent discussion on that topic.

[2] Provably secure means that finding a collision can be shown to be at least as hard as solving some concrete (usually number theoretic) problem.

As any collision M, M' for H is also a collision for H_1 *and* H_2, if *either* H_1 or H_2 is collision-resistant, so is H. Unfortunately the length of the output of H is the sum of the output lengths of H_1 and H_2, this makes the combiner quite unattractive for practical purposes.

1.1 The Boneh-Boyen and Our Result

Boneh and Boyen [2] ask whether one can combine CRHFs such that the output length is (significantly) less than what can be achieved by concatenation. They prove a first negative result in this direction, namely that there is no black-box construction for combining CRHFs in such a way that the output is shorter than what can be achieved by concatenation *under the additional assumption that this combiner queries each of the components exactly once.* They ask whether a similar impossibility result can be obtained in the general case where the combiner is allowed to query the components several times. We answer this question in the affirmative: any combiner for ℓ functions with range $\{0,1\}^v$ must have output length at least $(v - O(\log(q)))\ell$ bits[3], where q is the number of oracle calls made by the combiner. Stated in asymptotic terms, if $q \in 2^{o(v)}$ is subexponential, then the output length is in $(1 - o(1))v\ell$, and if q is constant the output length is in $v\ell - O(1)$, this must be compared to $v\ell$ which is trivially achieved by concatenation.

(k,ℓ)-Robust Combiner. In this paper we will consider the more general question whether secure and non-trivial (k,ℓ)-robust combiners for collision-resistant hash-functions exist. A (k,ℓ)-robust combiner is collision-resistant, if at least k (and not just one) of the components used are secure. We trivially get a (k,ℓ)-robust combiner by concatenating any $\ell - k + 1$ of the components,[4] which gives an output length of $v(\ell - k + 1)$. We show that this cannot be significantly improved as any (k,ℓ)-robust combiner must have output length at least $(v - O(\log(q)))(\ell - k + 1) - \ell$.

The main technical contribution of this paper is Lemma 2, which generalizes (and as a special case contains the statement of) Theorem 3 from [2]. Roughly, this lemma states that there exist hash-functions and a collision for any combiner with sufficiently short output, such that this collision does not trivially lead to collisions for all[5] of the hash-functions. The proof of this lemma follows from a simple application of the probabilistic method, and in particular is much simpler than the proof of Theorem 3 in [2].

An Information Theoretic Argument. There is a quite intuitive information theoretic argument why (k,ℓ)-robust combiners for CHRFs $\{0,1\}^* \to \{0,1\}^v$ whose output is significantly shorter than $v(\ell - k + 1)$ bits can't exist. We give this argument below, it will turn out that this simple approach gives an impossibility result which is much weaker than what we prove in this paper. This argument is shown only for motivational reasons and is not relevant for the

[3] In this paper all logarithms are to base 2.

[4] We'll look at this construction in more detail in the next section.

[5] Or for $\ell - k + 1$ of the hash-functions if we consider (k,ℓ)-robust combiners.

rest of the paper, the reader can skip the rest of this section if this does not seem to be of interest.

Basically, the argument uses the fact that one can encode a collision for any function with output length w using roughly w bits[6] and if the function is uniformly random, then w bits are also necessary. Now if a combiner with short output is instantiated with uniformly random functions, (the encoding of) a collision for the combiner will simply be too short to encode the information necessary to find collisions for the components. A bit more precisely, for $(1,2)$-robust combiners this argument goes as follows. Assume we are given a combiner C for two functions $\{0,1\}^* \to \{0,1\}^v$ whose output length is $2v-t$ (i.e. t bits less than concatenation). Now we simply sample two uniformly random functions $H_1, H_2 : \{0,1\}^* \to \{0,1\}^v$ and output a collision M, M' for C^{H_1,H_2}, such a collision can be encoded using $2v-t$ bits. To encode a collision for a random function $\{0,1\}^* \to \{0,1\}^v$, v bits are necessary and sufficient. Thus given the collision for the combiner, we still lack about $t/2$ bits of information (i.e. we have that much min-entropy) about a collision for one of the H_i's, and would have to make about $2^{t/2}$ more queries to this H_i in order to find a collision.This argument only rules out very strong combiners, where from any collision on the combiner we expect to get a collision for both components very efficiently. For example it does not rule out the possibility of $(1,2)$-robust combiners with range $3v/2$ (which we can consider significantly less than $2v$), where each collision for the combiner gives collisions for both components if we are ready to invest an additional $O(2^{v/4})$ queries. Such a combiner would still be sufficient if we are willing to assume that at least one of the components we combine has security (slightly more than) $2^{v/4}$. This assumption is very mild, as usually v is something like 160 or 256, such that the birthday bound $2^{v/2}$ is infeasible, but if a collision can be found after $2^{v/4}$ queries, the CRHF would be considered completely broken. More generally, the above argument does not rule out $(1,2)$-robust combiners with output length $2v-t$ for a t where $2^{t/2}$ queries are considered feasible (for an attacker). In contrast, the theorem proven in this paper rules out $(1,2)$-robust combiners with output length $2v-t$, unless the combiner itself makes $2^{t/2}$ invocations to the components.

[6] The following is a possible encoding. To define the encoding choose values $X_1, X_2, \ldots$ in $\{0,1\}^{w+1}$ uniformly at random. Now given a function $f : \{0,1\}^* \to \{0,1\}^w$ (which can be chosen adversarialy, but independent of the X_i's) let i be minimal such that $f(X_i)$ has at least 2 preimages in $\{0,1\}^{w+1}$ and output any $X \in \{0,1\}^{w+1}$ where $X \neq X_i$ and $f(X) = f(X_i)$. The expectation of i is at most 2 (as the probability that $f(Z)$ has only one preimage in $\{0,1\}^{w+1}$ for a random Z is at most $1/2$). Thus given X, we must make an expected number of at most 3 queries to f to find a collision (i.e. first compute $f(X)$ and then try $f(X_1), f(X_2), \ldots$ until $f(X) = f(X_i)$). If we only have $w+1-c$ (not $w+1$) bits for the encoding, we can simply omit the last c bits in the encoding just described, and when decoding trying all 2^c possibilities for this bits, thus we need an expected number of $2+2^c$ evaluations of f to find a collision given $w+1-c$ bits of X, which is better than no information at all if c is less than $w/2$.

1.2 Related Work

Combiners. The idea of combining two or more cryptographic components in order to get a system which is secure whenever at least one of the underlying primitives is secure is quite old.[7] The early results are on symmetric encryption schemes [1, 6, 11]. Combiners for asymmetric primitives were constructed by Dodis and Katz [5] (for CCA secure encryption schemes) and Harnik et al. [7] (for key-agreement). The general notion of a combiner was put forward by Herzberg [8] who calls them "tolerant combiners". In recent works one often calls them "robust combiners", a term introduced in [7]. Combiners have been generalized in several ways:

(k, ℓ)-Robust Combiners: [7] put forward the notion of (k, ℓ)-robust combiners as discussed in the last section. Such combiners are only guaranteed to be secure if at least k (and not just one) of the ℓ components used is secure. Interestingly, for natural primitives as statistically hiding commitments [8] and oblivious transfer [7, 13] only 2-3 but no 1-2 combiners are known.

Cross-Primitive Combiners: In a cross-primitive combiner the combined primitive is different from the components used, one can think of this as simultaneously being a reduction and a combiner. This notion was introduced by Meier and Przydatek [12] who construct a 1-2 private information retrieval to oblivious transfer cross-primitive combiner, which is interesting as normal 1-2 combiners for oblivious transfer might not exist [7].

Efficiency and Other Parameters: In practice the mere existence of a combiner is not enough, as the parameters of a combiner are important. Efficiency is always of concern, fortunately for most primitives where combiners are known to exist, also efficient realizations are known [7, 8], with bit-commitments being a notable exception [8] to that rule. Besides efficiency, for different primitives also other parameters are important, in particular this paper is about the output-length of combiners for CRHFs.

Collision Resistance. collision-resistant hash-functions are very important and subtle [15] cryptographic primitives which have attracted a lot of research, even more in the recent years as widely used (presumably) collision-resistant hash-functions as MD5 or SHA-1 have been broken [18, 19]. Here we only mention some of the generic results on CRHFs.

Simon shows that collision-resistant hash-functions cannot be constructed from one-way functions via a black-box reduction [17]. On the positive side, Naor and Yung [14] show that for some applications (in particular for signature schemes) collision resistance is not necessary, as universal one-way hash-functions are enough. Those can be constructed from one-way functions [10, 16].

Merkle and Damgård show that by iterating a CRHF with fixed input length, one gets a CRHF for inputs of arbitrary length. Most CRHFs used today follow

[7] We also see many combiners in the physical world, for example one often has several *different* locks on a door. This does not to simply increase the time a burglar needs to break the k locks by a factor of k, but there's hope that some particular lock might turn out to be much harder to come by than the others.

this approach. Coron et al. [4] show that the Merkle-Damgård construction does not give a random function if instantiated with a random function (which was not the design goal of this construction), but that this can be achieved with some small modifications. Joux [9] shows that for iterated hash-functions (like the Merkle-Damgård construction) finding many values which hash to the same value is not much harder than finding an ordinary collision. As a consequence concatenating the output of such hash-functions does not increase the security: let H_1, H_2 be iterated hash-functions with v bits output, then one can find a collision for $H(X) = H_1(X) \| H_2(X)$ in time $O(v2^{v/2})$.

2 Combiners For CRHFs

Informally, a (k, ℓ)-robust combiner for CRHFs is a construction (modeled as an oracle circuit C) which, if instantiated with any ℓ hash-functions $H_1, \ldots, H_\ell : \{0,1\}^* \rightarrow \{0,1\}^v$, is collision-resistant if at least k of the H_i's are. In order to show that a construction is a (k, ℓ)-robust combiner, one must provide an efficient procedure P which given two colliding inputs for the combiner, finds collisions for at least $\ell - k + 1$ of the underlying H_i's. In this paper we only consider black-box combiners as defined in [7], this means that C and P are only given oracle access to the H_i's.

The following definition of a (k, ℓ)-robust combiner is a generalization of the definition given in [2], where only the case $k = 1$ was considered.

Definition 1 *A combiner for ℓ collision-resistant hash-functions $\{0,1\}^* \rightarrow \{0,1\}^v$ is a pair (C, P) where C is an oracle circuit and P is an oracle probabilistic polynomial-time Turing machine (PPTM)*[8]

$$C : \{0,1\}^m \rightarrow \{0,1\}^n \qquad P : \{0,1\}^{2m} \rightarrow \{0,1\}^*.$$

There are ℓ types of oracle gates (tapes) in C (P). With $B^{H_1,\ldots,H_\ell}(X)$ (where B is C or P) we denote the output of B on input X when the ℓ types of oracle gates are instantiated with functions $H_1, \ldots, H_\ell : \{0,1\}^* \rightarrow \{0,1\}^v$ respectively.

*We say that P k-*succeeds* on $M, M' \in \{0,1\}^*$ and oracles $H_1, \ldots, H_\ell$ if its output contains collisions for all but at most $k - 1$ of the H_i's, i.e. for*

$$P^{H_1,\ldots,H_\ell}(M, M') \rightarrow (U_1, \ldots, U_\ell, U'_1, \ldots, U'_\ell)$$

we have

$$\exists J \subseteq \{1, \ldots, \ell\}, |J| \geq \ell - k + 1 : (U_i, U'_i) \textit{ is a collision for } H_i.$$

Let $Adv_P^k[(H_1, \ldots, H_\ell), (M, M')]$ denote the probability (over P's coin tosses) that $P^{H_1,\ldots,H_\ell}(M, M')$ k-succeeds. Then (C, P) is an ϵ**-secure** (k, ℓ)**-combiner***, if for all (compatible) $H_1, \ldots, H_\ell$ and all collisions (M, M') on $C^{H_1,\ldots,H_\ell}$ we have*

$$Adv_P^k[(H_1, \ldots, H_\ell), (M, M')] > 1 - \epsilon.$$

[8] The only reason P is defined as a Turing machine and not as a circuit is that we don't want to put an a priori bound on the output length of P.

We say that (C,P) is an (k,ℓ)-robust combiner if it is ϵ-secure for a small ϵ.[9]

For example consider the following (k,ℓ)-robust combiner (C,P)

$$C^{H_1,\ldots,H_\ell}(M) \to H_1(M)\|\ldots\|H_{\ell-k+1}(M)$$

$$P^{H_1,\ldots,H_\ell}(M,M') \to (M,\ldots,M),(M',\ldots,M')$$

As any collision M,M' for $C^{H_1,\ldots,H_\ell}$ is a collision for H_i for $i=1,\ldots,\ell-k+1$,

$$Adv_P^k[(H_1,\ldots,H_\ell),(M,M')] = 1.$$

So (C,P) can be considered a secure (k,ℓ)-robust combiner, as from any collision on $C^{H_1,\ldots,H_\ell}$ we get from P collisions for all but $k-1$ of the H_i's, thus if k of the H_i's are secure, also $C^{H_1,\ldots,H_\ell}$ must be secure. The output length of C is $n = v(\ell - t + 1)$, by the following theorem this cannot be significantly improved.

Theorem 1 *Let (C,P) be a (k,ℓ)-robust combiner, where $C : \{0,1\}^m \to \{0,1\}^n$ has q_C oracle gates and P makes at most q_P oracle calls. Suppose that*

$$n < (v - 2\log(2q_C))(\ell - k + 1) - \ell - 1 \quad \textit{and} \quad m > n. \tag{2}$$

Then there exist $M,M' \in \{0,1\}^m$ and functions $\hat{H}_i : \{0,1\}^* \to \{0,1\}^v$ for $i = 1,\ldots,\ell$ relative to which

$$Adv_P^k[(\hat{H}_1,\ldots,\hat{H}_\ell),(M,M')] \le \frac{(q_P+q_C)^2+k}{2^v}. \tag{3}$$

For the special case where $k=1$ and C queries each $\hat{H}_i$ exactly once (which are the constructions considered in [2]) the bound on n can be improved to

$$n < v\ell - 1 \quad \textit{and} \quad m > n$$

or

$$n < v\ell \quad \textit{and} \quad m - 1 > n.$$

The last statement slightly improves on the main result from [2] where a stronger $n < m - \log \ell$ bound was needed in order to get $n < v\ell$. As we can find a collision for any function with range $\{0,1\}^v$ in $2^{v/2}$ steps, in order to reason about CRHFs with range $\{0,1\}^v$ the value $2^{v/2}$ must be unfeasibly large. In particular for any reasonable combiner $q_P + q_C \ll 2^{v/2}$ and thus the advantage (3) will be very small.

Let us remark that in [2] the ranges of the H_i's were allowed to be different, for the sake of exposition we drop this generalization, but it is straight forward to adapt (the proof of) Theorem 1 to this more general case. Note that when the H_i's have different output lengths, say H_i has length v_i where $v_1 \le v_2 \le v_3 \le \ldots \le v_\ell$, then we can construct a (k,ℓ)-robust combiner by concatenating the outputs of $H_1,\ldots,H_{\ell-k+1}$ (i.e. the H_i's with the shortest outputs), which

[9] Here “small” usually means negligible in some security parameter.

will give a combiner with output length $\sum_{i=1}^{\ell-k+1} v_i$. Again, this is basically best possible, as for this setting Theorem 1 holds by generalizing equation (2) to

$$n < \sum_{i=1}^{\ell-k+1} (v_i - 2\log(2q_C)) - \ell - 1 \quad \text{and} \quad m > n.$$

and replacing v with v_1 in (3).

Following [2], to prove Theorem 1 it is sufficient to prove that hash-functions $H_1, \ldots, H_\ell$ and a collision M, M' exists where in the computation of $C^{H_1,\ldots,H_\ell}$ on inputs M and M' at least k of the H_i's are not queried on two distinct inputs X, X' where $H_i(X) = H_i(X')$. Note that this means that one does not trivially get a collision for those H_i's when learning M, M'. Let $J \subseteq \{1, \ldots, \ell\}, |J| = k$ be the indices of these k H_i's. We prove the existence of such H_i's and M, M' in Lemma 2 below. Then, from such $H_1, \ldots, H_\ell$ and M, M' we can get the $\hat{H}_1, \ldots, \hat{H}_\ell$ as required by Theorem 1, by setting $\hat{H}_i(X) = H_i(X)$ for all inputs X which appear as input to H_i in the computation of $C^{H_1,\ldots,H_\ell}(M)$ or $C^{H_1,\ldots,H_\ell}(M')$, and $\hat{H}_i(X)$ is assigned a random value otherwise. Clearly M, M' is also a collision for $C^{\hat{H}_1,\ldots,\hat{H}_\ell}$, moreover all $\hat{H}_i$ where $i \in J$ are "very" collision-resistant, as we just randomly defined their outputs, except on a subset of inputs which itself does not contain a collision, Lemma 1 below is a formal statement of this intuitive argument.

Proof (of Theorem 1). The theorem follows from Lemmata 1 and 2.

In the lemmata below[10] let

- $\mathbf{W}_i(X)$ be the set of oracle queries to H_i made while evaluating $C^{H_1 \ldots H_\ell}(X)$.
- $\mathbf{V}_i(X) = \{H_i(W) : W \in \mathbf{W}_i(X)\}$ be the set of corresponding outputs (taken without repetition).

Lemma 1 *Let (C, P) be a (k, ℓ)-robust combiner, where C has q_C oracle gates and P makes at most q_P oracle calls. Assume there exist oracles $H_i : \{0,1\}^* \to \{0,1\}^v, i = 1, \ldots, \ell$ and messages M, M' such that*

- *$M \neq M'$ and $C^{H_1,\ldots,H_\ell}(M) = C^{H_1,\ldots,H_\ell}(M')$.*
- *$|\mathbf{V}_j(M) \cup \mathbf{V}_j(M')| = |\mathbf{W}_j(M) \cup \mathbf{W}_j(M')|$ for at least k different $j \in \{1, \ldots, \ell\}$.*

Then there exist deterministic $\hat{H}_i : \{0,1\}^* \to \{0,1\}^v, i = 1, \ldots, \ell$ relative to which

$$Adv_P^k[(\hat{H}_1, \ldots, \hat{H}_\ell), (M, M')] \le \frac{(q_P + q_C)^2 + k}{2^v}.$$

Proof. Let $J \subseteq \{1, \ldots, \ell\}, |J| = k$ be the indices of the k hash-functions for which no collision occurs during the computation of $C^{H_1,\ldots,H_\ell}$ on input M and M', i.e.

$$\forall j \in J : |\mathbf{V}_j(M) \cup \mathbf{V}_j(M')| = |\mathbf{W}_j(M) \cup \mathbf{W}_j(M')|.$$

[10] Our Lemma 1 is basically Theorem 2 from [2], the only difference is that we consider (k, ℓ)-robust combiners whereas [2] were only interested in the case $k = 1$.

For $i \notin J$ we let $\hat{H}_i := H_i$, and for each $i \in J$ let $R_i : \{0,1\}^* \to \{0,1\}^v$ be uniformly random and

$$\hat{H}_i(W) := \begin{cases} H_i(W) \text{ if } W \in \mathbf{W}_i(M) \cup \mathbf{W}_i(M') \\ R_i(W) \text{ otherwise} \end{cases}$$

Note that $C^{\hat{H}_1,\ldots,\hat{H}_\ell}(M) = C^{\hat{H}_1,\ldots,\hat{H}_\ell}(M')$ as for each i, $H_i(W) = \hat{H}_i(W)$ for inputs $W \in \mathbf{W}_i(M) \cup \mathbf{W}_i(M')$ which come up on the computation of $C^{H_1,\ldots,H_\ell}$ on inputs M, M', let $\mathbf{Q}$ denote all those inputs together with the corresponding outputs.

$$\mathbf{Q} = \bigcup_{i=1}^{\ell} \{\mathbf{V}_i(M), \mathbf{W}_i(M), \mathbf{V}_i(M'), \mathbf{W}_i(M')\}$$

Let P' be the oracle PPTM which makes at most q_P oracle calls and maximizes the probability α defined below.

$$\alpha = \Pr_{P'^{\hat{H}_1,\ldots,\hat{H}_\ell}(\mathbf{Q}) \to \{U_1,\ldots,U_\ell,U'_1,\ldots,U'_\ell\}} [\exists i \in J : U_i \neq U'_i \wedge \hat{H}_i(U_i) = \hat{H}_i(U'_i)] \quad (4)$$

α is an upper bound on $Adv_P^k[(\hat{H}_1,\ldots,\hat{H}_\ell),(M,M')]$, as one possible strategy for P' is to first compute M, M', which given $\mathbf{Q}$ can be done without access to the $\hat{H}_i$ oracles, and then simulate $P^{\hat{H}_1,\ldots,\hat{H}_\ell}(M,M')$ and output the output of this simulation.[11] To save on notation let P^* denote $P'^{\hat{H}_1,\ldots,\hat{H}_\ell}(\mathbf{Q})$. We say that P^* found a collision if for some[12] $\hat{H}_i, i \in J$ it makes an oracle query $\hat{H}_i(X)$ where either for a previous query $X' \neq X$ to $\hat{H}_i$ we have $\hat{H}_i(X) = \hat{H}_i(X')$ or $\hat{H}_i(X) \in \mathbf{V}_i(M) \cup \mathbf{V}_i(M')$ and $X \notin \mathbf{W}_i(M) \cup \mathbf{W}_i(M')$. For $i = 1, \ldots, q_P$ let $\mathcal{C}_i$ denote the event that P^* found a collision after the i'th oracle query is made. If the i'th oracle query is to a $\hat{H}_j$ where $j \notin J$ or a query which has already been made we cannot get a collision, so

$$\Pr[\mathcal{C}_i | \neg\mathcal{C}_{i-1}] = 0.$$

So assume that the i'th oracle query is a new query X to a $\hat{H}_j$ where $j \in J$. Then $\hat{H}_i(X) = R_i(X)$ is uniformly random and independent of any previous outputs, thus the probability that it will collide with any of the $\leq i$ previous queries to $\hat{H}_i$ or with one the $\leq 2q_C$ values in $\mathbf{V}_i(M) \cup \mathbf{V}_i(M')$ is at most $(2q_C + i)/2^v$, we get

$$\Pr[\mathcal{C}_{q_P}] = \sum_{i=1}^{q_P} \Pr[\mathcal{C}_i | \mathcal{C}_{i-1}] \leq \sum_{i=1}^{q_P} \frac{2q_C + i}{2^v} \leq \frac{q_P(2q_C + q_P)}{2^v} \leq \frac{(q_P + q_C)^2}{2^v}.$$

[11] The reason we give away the full $\mathbf{Q}$ is that that M, M' will usually leak some information on $\mathbf{Q}$, and the simplest way to deal with this leakage is to simply assume that P' knows all those values.

[12] Note that we don't care about collision for $\hat{H}_i, i \notin J$ as $\mathbf{Q}$ contains collisions for those $\hat{H}_i$'s.

Even if $\neg\mathcal{C}_{q_P}$, i.e. P^* does not find a collision for some $\hat{H}_i, i \in J$, there still is a tiny chance that P^* guesses U_i, U_i' where $\hat{H}_i(U_i) = \hat{H}_i(U_i')$ for some of the $i \in J$. The probability of this is at most $|J|/2^v \leq k/2^v$. Taking everything together:

$$Adv_P^k[(\hat{H}_1, \ldots, \hat{H}_\ell), (M, M')] \leq \alpha \leq \Pr[\mathcal{C}_{q_P}] + k/2^v \leq \frac{(q_P + q_C)^2 + k}{2^v}. \qquad (5)$$

We're almost done, except that in the above inequality, the $\hat{H}_i$'s are not deterministic as required by the lemma, but randomized (as the R_i's were chosen at random). We can get fixed $\hat{H}_i$'s for which (5) holds by choosing the R_i's so they minimize the left hand side of (5). □

Lemma 2 *Let $C : \{0,1\}^m \to \{0,1\}^n$ be as in the previous lemma. Then whenever*

$$n < (v - 2\log(2q_C))(\ell - k + 1) - \ell - 1 \quad \textit{and} \quad m > n$$

there exist functions $H_1, \ldots, H_\ell$ and messages M, M' such that

- *$M \neq M'$ and $C^{H_1,\ldots,H_\ell}(M) = C^{H_1,\ldots,H_\ell}(M')$.*
- *$|\mathbf{V}_j(M) \cup \mathbf{V}_j(M')| = |\mathbf{W}_j(M) \cup \mathbf{W}_j(M')|$ for at least k different $j \in \{1, \ldots, \ell\}$.*

For the special case where $k = 1$ and C queries each H_i exactly once (which are the constructions considered in [2]) the bounds on n can be improved to

$$n < v\ell - 1 \quad \textit{and} \quad m > n$$

or

$$n < v\ell \quad \textit{and} \quad m - 1 > n.$$

Proof. Consider the following random experiment. First we sample ℓ functions $H_i : \{0,1\}^* \to \{0,1\}^v$ uniformly at random.[13] Then $M, M' \in \{0,1\}^m$ are sampled uniformly at random. We define the events $\mathcal{E}_1$ and $\mathcal{E}_2$ as

$$\begin{aligned} \mathcal{E}_1 &\iff M \neq M' \text{ and } C^{H_1,\ldots,H_\ell}(M) = C^{H_1,\ldots,H_\ell}(M') \\ \mathcal{E}_2 &\iff \exists J \subseteq \{1, \ldots, \ell\}, |J| > \ell - k \\ &\qquad \text{where } \forall j \in J : |\mathbf{V}_j(M) \cup \mathbf{V}_j(M')| \neq |\mathbf{W}_j(M) \cup \mathbf{W}_j(M')| \end{aligned}$$

We will show that $\Pr[\mathcal{E}_1] > \Pr[\mathcal{E}_2]$, which then implies $\Pr[\mathcal{E}_1 \wedge \neg\mathcal{E}_2] > 0$. This will prove the lemma as it shows that random $H_1, \ldots, H_\ell$ and M, M' have the property as claimed by the lemma with non-zero probability, and thus $H_1, \ldots, H_\ell$ and M, M' with this property exist.

As $\Pr[M = M'] = 2^{-m}$, $\Pr[C^{H_1,\ldots,H_\ell}(M) = C^{H_1,\ldots,H_\ell}(M')] \geq 2^{-n}$ and $m > n$ we get

$$\Pr[\mathcal{E}_1] \geq 2^{-n} - 2^{-m} \geq 2^{-n-1}. \qquad (6)$$

[13] One can't simply sample a H_i as this would need infinite randomness, but one can use lazy sampling here, this means that $H_i(X)$ is only assigned a (random) value when H_i is actually invoked on input X.

Let q_i denote the number of H_i oracle gates in C, note that $\sum_{i=1}^{\ell} q_i = q_C$. We can upper bound $\Pr[\mathcal{E}_2]$ by the probability that the best oracle algorithm $A^{H_1,\ldots,H_\ell}$ which can query the i'th oracle H_i at most $2q_i$ times finds a collision for at least $\ell - k + 1$ of the H_i's.[14] As the H_i's are all independent random functions, the best A can do is to query it i'th oracle on $2q_i$ distinct inputs (which ones is irrelevant), by the birthday bound[15] the probability of finding a collision for any H_i is at most $2q_i(2q_i - 1)/2^{v+1}$, now

$$\begin{aligned}
\Pr[\mathcal{E}_2] &\le \Pr[A^{H_1,\ldots,H_\ell} \text{ finds } \ell - k + 1 \text{ collisions }] \\
&\le \sum_{\substack{J \subseteq \{1,\ldots,\ell\} \\ |J| = \ell - k + 1}} \Pr[\forall i \in J : A^{H_1,\ldots,H_\ell} \text{ finds a collision for } H_i] \\
&\le \sum_{\substack{J \subseteq \{1,\ldots,\ell\} \\ |J| = \ell - k + 1}} \prod_{i \in J} \frac{2q_i(2q_i - 1)}{2^{v+1}} \\
&< \sum_{\substack{J \subseteq \{1,\ldots,\ell\} \\ |J| = \ell - k + 1}} \frac{(2q_C^2)^{\ell - k + 1}}{2^{v(\ell - k + 1)}} \le \binom{\ell - k + 1}{\ell} \frac{(2q_C^2)^{\ell - k + 1}}{2^{v(\ell - k + 1)}} < \frac{2^\ell (2q_C^2)^{\ell - k + 1}}{2^{v(\ell - k + 1)}}.
\end{aligned}$$

From the above equation, (6) and $n < (v - 2\log(2q_C))(\ell - k + 1) - \ell - 1$ we now get $\log(\Pr[\mathcal{E}_1]) > \log(\Pr[\mathcal{E}_2])$, and thus $\Pr[\mathcal{E}_1] > \Pr[\mathcal{E}_2]$, as

$$\log(\Pr[\mathcal{E}_1]) \ge \log(2^{-n-1}) = -n - 1 > -(v - 2\log(2q_C))(\ell - k + 1) + \ell$$

and

$$\log(\Pr[\mathcal{E}_2]) < \log\left(\frac{2^\ell (2q_C^2)^{\ell - k + 1}}{2^{v(\ell - k + 1)}}\right) = -(v - 2\log(2q_C))(\ell - k + 1) + \ell$$

Our estimate on $\Pr[\mathcal{E}_2]$ has some slack as to keep the expression simple. For the special case $k = 1$ and $q_i = 1, i = 1, \ldots, \ell$ which covers the constructions considered in [2] we get

$$\Pr[\mathcal{E}_2] \le \prod_{i \in \{1,\ldots,\ell\}} \frac{2q_i(2q_i - 1)}{2^{v+1}} = 2^{-v\ell}$$

which satisfies $\Pr[\mathcal{E}_1] > \Pr[\mathcal{E}_2]$ already for $n < v\ell - 1$. If we additionally assume that $n < m - 1$ (not just $n < m$) then we can strengthen (6) to $\Pr[\mathcal{E}_1] > 2^{-n-1}$ and $\Pr[\mathcal{E}_1] > \Pr[\mathcal{E}_2]$ holds for the optimal $n < v\ell$. □

[14] This is an upper bound as one possible strategy for $A^{H_1,\ldots,H_\ell}$ is to simply evaluate $C^{H_1,\ldots,H_\ell}$ on two random inputs M, M' to get success probability exactly $\Pr[\mathcal{E}_2]$.

[15] This bound states that when randomly throwing q balls into N buckets, some bucket will contain more than one element with probability at most $q(q-1)/2N$.

References

1. C. A. Asmuth and G. R. Blakley. An efficient algorithm for constructing a cryptosystem which is harder to break than two other cryptosystems. *Computers and Mathematics with Applications*, pages 447–450, 1981.
2. Dan Boneh and Xavier Boyen. On the impossibility of efficiently combining collision resistant hash functions. In *CRYPTO*, 2006.
3. Scott Contini, Arjen K. Lenstra, and Ron Steinfeld. Vsh, an efficient and provable collision-resistant hash function. In *EUROCRYPT*, pages 165–182, 2006.
4. Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-damgård revisited : How to construct a hash function. In *Advances in Cryptology — CRYPTO '05*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448, 2005.
5. Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In *TCC*, pages 188–209, 2005.
6. Shimon Even and Oded Goldreich. On the power of cascade ciphers. *ACM Trans. Comput. Syst.*, 3(2):108–116, 1985.
7. Danny Harnik, Joe Kilian, Moni Naor, Omer Reingold, and Alon Rosen. On robust combiners for oblivious transfer and other primitives. In *EUROCRYPT*, pages 96–113, 2005.
8. Amir Herzberg. On tolerant cryptographic constructions. In *CT-RSA*, pages 172–190, 2005.
9. Antoine Joux. Multicollisions in iterated hash functions. application to cascaded constructions. In *CRYPTO*, pages 306–316, 2004.
10. Jonathan Katz and Chiu-Yuen Koo. On constructing universal one-way hash functions from arbitrary one-way functions, 2005. Cryptology ePrint Archive: Report 2005/328.
11. Ueli M. Maurer and James L. Massey. Cascade ciphers: The importance of being first. *J. Cryptology*, 6(1):55–61, 1993.
12. Remo Meier and Bartosz Przydatek. On robust combiners for private information retrieval and other primitives. In Cynthia Dwork, editor, *CRYPTO '06*, volume 4117 of *Lecture Notes in Computer Science*, pages 555–569, 2006.
13. Remo Meier, Bartosz Przydatek, and Jürg Wullschleger. Robuster combiners for oblivious transfer. In *TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 404–418, 2007.
14. Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *STOC*, pages 33–43, 1989.
15. Phillip Rogaway. Formalizing human ignorance: Collision-resistant hashing without the keys, 2006. Cryptology ePrint Archive: Report 2006/281.
16. John Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.
17. Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *EUROCRYPT*, pages 334–345, 1998.
18. Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full sha-1. In *CRYPTO*, pages 17–36, 2005.
19. Xiaoyun Wang and Hongbo Yu. How to break md5 and other hash functions. In *EUROCRYPT*, pages 19–35, 2005.