Building a Collision-Resistant Compression Function from Non-Compressing Primitives

Thomas Shrimpton[1] and Martijn Stam[2]

[1] University of Lugano and Portland State University thomas.shrimpton@unisi.ch
[2] EPFL martijn.stam@epfl.ch

Abstract. We consider how to build an efficient compression function from a small number of random, non-compressing primitives. Our main goal is to achieve a level of collision resistance as close as possible to the optimal birthday bound. We present a $2n$-to-n bit compression function based on three independent n-to-n bit random functions, each called only once. We show that if the three random functions are treated as black boxes then finding collisions requires $\Theta(2^{n/2}/n^c)$ queries for $c \approx 1$. This result remains valid if two of the three random functions are replaced by a fixed-key ideal cipher in Davies-Meyer mode (i.e., $E_K(x) \oplus x$ for permutation E_K). We also give a heuristic, backed by experimental results, suggesting that the security loss is at most four bits for block sizes up to 256 bits. We believe this is the best result to date on the matter of building a collision-resistant compression function from non-compressing functions. It also relates to an open question from Black et al. (Eurocrypt'05), who showed that compression functions that invoke a single non-compressing random function cannot suffice.
We also explore the relationship of our problem with that of doubling the output of a hash function and we show how our compression function can be used to double the output length of ideal hashes.

Keywords. Hash Functions, Random Oracle Model, Compression Functions, Collision Resistance.

1 Introduction

Hash functions are a central cryptographic primitive, appearing in countless protocols and applications. Dedicated hash functions such as MD5 and SHA1 have dominated practice, as these are relatively fast and, until recently, they were believed to resist collision-finding attacks. But this belief has been shown to be unfounded: successful attacks have been published against most members of the MD/SHA-family. This has prompted a renewed interest in design methodologies for hash functions, with a particular emphasis on providing formal guarantees of collision resistance (along with other properties).

The design of hash functions usually proceeds in two stages. First one designs a compression function with fixed domain, typically bitstrings of some small length. One then applies a domain extension method, such as the Merkle-Damgård transform [11, 24], to the compression function in order to construct a hash function for messages of arbitrary length. The first part has our interest; in particular, the central problem considered by this paper is the following one:

> Given a (small) number of independent n-to-n bit random (one-way) functions, construct a $2n$-to-n bit compression function with provable collision resistance as close as possible to the optimal $2^{n/2}$ birthday bound.

This problem is related to one recently considered by Maurer and Tessaro [22], who consider the problem of constructing a function $C : \{0,1\}^{m(n)} \to \{0,1\}^{l(n)}$ given a n-to-n bit random one-way function f. Setting $m(n) = 2n$ and $l(n) = n$ gives rise to a $2n$-to-n bit compression function. Maurer and Tessaro cast security in the indifferentiability framework, in which case domain extension is actually far more challenging than

This is the full version of a paper appearing at ICALP'08 [37], an earlier version [36] was presented at the Ecrypt Workshop on Hash Functions (May 2007).

range extension. For all $\epsilon > 0$ they give a construction indifferentiable from a random function against adversaries making $\Theta(2^{n(1-\epsilon)})$ queries. Indifferentiability is a much stronger requirement than just collision resistance, indeed for $\epsilon \geq 1/2$ their functions have better proven security than ours. Unfortunately, the construction by Maurer and Tessaro is not very efficient, to get the required collision resistance for a $2n$-to-n bit compression function requires 99 calls to the underlying primitive f.

This raises the question whether more efficient constructions are possible, at least when one focuses primarily on collision resistance. Ideally we would like to make do with just one random function, and to invoke it once for each n-bit block of message digested; such a compression function would be called rate-1. Unfortunately, Black et al. [6] have given a negative result that all but rules this out. One can also show that rate-1/2 compression functions (with optimal collision resistance) are unlikely to exist. Thus the best one can hope for will be a rate-1/3 compression function.

We show that this hope *can* be realized. In particular, we present a compression function that calls random n-to-n bit functions f_1, f_2, f_3, and that has almost optimal collision resistance (here n is a parameter that can be chosen freely). When we consider the construction for increasing n, we show that any adversary making $\Theta(2^{n/2}/n^c)$ total oracle queries, for $c > 1$, has a vanishing probability of finding a collision in H. On the other hand, for $c < 1$ we provide compelling arguments that an adversary exists that will find a collision with high probability. Thus it is fair to say that finding collisions takes around $\Theta(2^{n/2}/n)$ queries. For concreteness' sake, we also develop a heuristic indicating that for n of cryptographic relevance (up to 256 bits), the loss is at most 4 bits of collision resistance. Of course, the standard caveats apply to these results when one instantiates the random functions in practice.

Central to our proof is the concept of the *yield* of a set of queries. This is the number of compression function evaluations an adversary can make given a certain number of queries to the underlying primitives. Somewhat surprisingly, we can show that for our compression function an adversary can do not much better than simply optimizing his yield and hoping for a collision via the birthday bound. The yield can also be used to get (crude) negative results, for instance for any rate-1/2 compression function there exists a (greedy) adversary with yield $2^{n/2}$ using only $2^{n/4}$ queries. Since we expect good hash functions to behave randomly, this indicates it is unlikely to find a rate-1/2 compression function with good collision resistance. This use of the yield to obtain impossibility results has recently been extensively generalized by Rogaway and Steinberger [34] and later Stam [38], who pointed out some problems with the above reasoning. In particular, he demonstrates a rate-1/2 compression function that is collision resistant up to almost $2^{n/3}$ queries (beating the intuitive $2^{n/4}$ bound).

Our rate-1/3 construction itself is as follows:

$$H^{f_1,f_2,f_3}(V, M) = f_3\left(f_1\left(M\right) \oplus f_2\left(V\right)\right) \oplus f_1\left(M\right) \ .$$

A picture of the compression function is given in Figure 1. The compression function can easily be transformed into a hash function with arbitrary domain while preserving the collision resistance (e.g., using the Merkle-Damgård transform). As a note of warning, we do not claim any "beyond-birthday" properties one might hope for from a hash function, such as resistance against multi-collisions and optimal preimage resistance. Indeed, preimages can typically be found in $O(2^{2n/3})$ queries, rather than the desired $\Omega(2^n)$. Rogaway and Steinberger [34] show that this reduced preimage-resistance is to a large extent inherent to rate-1/3 schemes and not an artefact of our particular scheme.

We also investigate what happens when one wants to instantiate f_1, f_2, and f_3 with a blockcipher with its key fixed (thus supplying the adversary with inverse oracles for each). When f_1 or f_2 are instantiated directly with a fixed-key ideal cipher (i.e., random two-way permutations instead of random functions), the construction breaks down badly. However, we show that by using a Davies-Meyer like construction in the

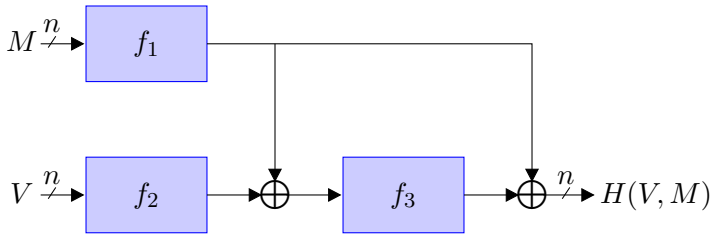

Fig. 1: The triple-function compression function. The functions f_1, f_2, f_3 are random n-to-n bit functions.

place of f_1 and f_2 we can also deal with a fixed-key ideal cipher in those places without loss of security. Using a fixed-key ideal cipher for f_3 does not seem to affect collision resistance, yet we do not have a full proof for this.

The emphasis on blockcipher based hashing can be understood both historically and practically. Blockciphers have long been the central primitive in symmetric key cryptography, and there exists some measure of confidence in blockcipher designs. From a practical perspective, one might like to reuse optimized code or hardware implementations of blockciphers that are already implemented for encryption and message authentication within certain applications. That said, there seems to be no intrinsic theoretical reason to restrict designs to using blockciphers, hence our focus on simple random one-way functions (cf. [5, 22]).

Interestingly, the problem of building a compression function using random functions (as opposed to a fixed-key ideal cipher) has an unexpected link with that of constructing *double-length* hash functions. Specifically:

> Given an ideal $2n$-to-n bit compression function create a $4n$-to-$2n$ bit compression function with collision resistance close to the optimal 2^n.

In Appendix A we show that range extenders could be used as an alternative means to turn a number of non-compressing random functions into a compressing one, and vice versa. Therefore one could consider *range extension* as the dual problem to domain extension. In particular, one can use our method to get a rate-1/3 $4n$-to-$2n$ bit double-length compression function with close to optimal collision resistance (in the output size) based on a set of $2n$-to-n bit random functions.

RELATED WORK. Bellare and Micciancio [2] introduce incremental hash functions, which in principle could be built upon a non-compressing primitive. Crucial differences with our work are that they build an entire hash function, not just a compression function, and that the collision resistance of their schemes is not based on query complexity (usually just n queries suffice for a collision), but on the presumed computational hardness of combining the query answers into an actual collision.

Bernstein [5] bases his Rumba20 compression function on these ideas. He xor's together the output of four pseudorandom generators, components of the Salsa20 streamcipher. By modelling the underlying primitives as (independent) random one-way functions, he shows an *upper* bound (and estimate) on the full complexity for collision finding of $O(2^{n/3})$, well below the birthday bound (and our *lower* bound). Note that the query complexity for finding collisions is $\Theta(2^{n/8})$ (cf. Wagner [40]). The expanding nature of Bernstein's primitives somewhat complicates theoretical efficiency comparisons, but the rate of his scheme is arguably 1/2, so more efficient than ours.

If one is willing to digest only *one bit* at a time (instead of n), then existing techniques already provide a way to construct an optimally collision-resistant hash function out of two random functions $f_0, f_1 \colon \{0,1\}^n \to \{0,1\}^n$. Define a compression function $g \colon \{0,1\}^{n+1} \to \{0,1\}^n$ by $g(b \,||\, V) = f_b(V)$ that takes a single bit b of message to select which function will process the chaining value V. One can show that g is a completely random function iff both f_0 and f_1 were, and by using g in the block-chaining hash function from Damgård [11] one obtains a hash function that digests one bit per iteration. As an aside, the very same hash function emerges by using Damgård's construction based on a pair of claw free permutations[10], but using the functions f_0 and f_1 instead.

We already mentioned the work of Maurer and Tessaro [22]. Prior to this work Aiello and Venkatesan [1] considered the secret key equivalent of doubling the length of a random function. That is, given a set of n-to-n bit functions f_i (for $i = 1, \ldots, 8$) construct a $2n$-to-$2n$ bit function F that is indistinguishable to a random $2n$-to-$2n$ bit function for an adversary making up to 2^n adaptive queries to F. The crucial difference with our work (and that of Maurer and Tessaro) is that they do not allow the adversary access to the inner random functions f_i. Consequently there is no a priori reason to expect that schemes proven secure in the secret model, even when beyond the birthday barrier, have good collision resistance in the public world.

Nonetheless, the scheme presented by Aiello and Venkatesan, dubbed Benes, bears some resemblance with ours that is worth pointing out. Given two n bit inputs M and V, first a butterfly construct is used to compute intermediate values $X = f_1(M) \oplus f_2(V)$ and $Y = f_3(M) \oplus f_4(V)$. This pattern is then repeated to get the final output $G = f_5(X) \oplus f_6(Y)$ and $H = f_7(X) \oplus f_8(Y)$. If one ignores the first output (G), rendering f_5 and f_6 superfluous and if one simplifies further by setting $f_2 = f_3 = 0$ (the constant zero functionality) and f_8 the identity function, one obtains our construction. As such, the proof we provide for the collision resistance of our scheme can likely be reused (and probably even strengthened a bit) to prove collision resistance of the general rate-1/6 Benes scheme with one output chopped.

There has also been extensive research into the construction of hash functions based on blockciphers. Davies-Meyer [23], Matyas-Meyer-Oseas [21], and Miyaguchi-Preneel [26, 32] are all well-known $2n$-to-n bit compression functions based on a single call to a blockcipher with n-bit key operating on n-bit blocks. These type of *rate-1* constructions were later systematically studied by Preneel et al. [32] and Black et al. [7], who identified twelve distinct constructions that provide optimal collision resistance when the blockcipher is modelled as an ideal cipher. Black et al. [7] showed that an additional eight constructions do not yield collision resistant compression functions, yet still lead to collision resistant hash functions when properly MD-iterated.

Most of the work related to blockcipher-based compression functions allows per-round rekeying [7, 16, 28, 32]. As such, the primitive already compresses in the sense that it takes more input (key and plaintext data) than that it provides output. This significantly eases design and proof. Unfortunately, rekeying has the drawback of entailing a significant computational cost: Gladman's implementation survey [15] shows that AES key scheduling would account for nearly 50% of the overall runtime. Thus a lower rate fixed-key solution might actually be more efficient. Fixing the key would make use of the blockcipher in a more natural way, namely setting up a key once and then processing relatively large amounts of data with it (the way that blockciphers are used for encryption). A fixed-key blockcipher would be modelled as a random two-way permutation.

Preneel et al. [31] propose a family of fixed-key constructions, but no formal security proof is given. The rate of their scheme is always strictly smaller then 1/2 and typically between 1/4 and 1/8. Knudsen [19] proposed a rate-1 fixed-key construction, however, it was quickly broken [29].

Black et al. [6] explicitly consider fixed-key blockciphers and show an impossiblity result for rate-1 schemes. They show that if one uses the MD-transform over a $2n$-to-n-bit compression function that makes

only one call to a random n-to-n-bit function, then there always exists an information-theoretic adversary that finds collisions efficiently (i.e., using just few calls to the non-compressing primitive). Although Black et al. [6] phrase their results in terms of two-way random permutations, their result holds for the random one-way functions we consider as well. However, they do not consider compression functions that call out to more than one primitive per message block (and thus have a rate less than one).

Recently (but subsequent to the initial presentation of our work [36]), Rogaway and Steinberger [34] have given impossibility results for schemes calling out more than once per message block, although under the assumption that the compression function is suitably uniform (a refinement of their work is given by Stam [38]). They have also constructed a rate-1/3 scheme based on a fixed-key ideal cipher that achieves security comparable to ours [33].

2 Preliminaries

GENERAL NOTATION. For a positive integer n, we write $\{0,1\}^n$ for the set of all bitstrings of length n. When X and Y are strings we write $X \,||\, Y$ to mean their concatenation and $X \oplus Y$ to mean their bitwise exclusive-or (xor). Unless specified otherwise, we will consider bitstrings as elements in the group $(\{0,1\}^n, \oplus)$.

For positive integers m and n, we let $\mathrm{Func}(m,n)$ denote the set of all functions mapping $\{0,1\}^m$ into $\{0,1\}^n$. We write $f \xleftarrow{\$} \mathrm{Func}(m,n)$ to denote random sampling from the set $\mathrm{Func}(m,n)$ and assignment to f. Unless otherwise specified, all finite sets are equipped with a uniform distribution.

DISTRIBUTIONS AND TENSORS. With $(\{0,1\}^n)^q$ we denote the set of q-element vectors, or *q-vectors*, in which each element is an n-bit string. When $\mathbf{a} \in (\{0,1\}^n)^q$ and $\mathbf{b} \in (\{0,1\}^n)^q$, we will write $\mathbf{a} = (a_1, \ldots, a_q)$ and $\mathbf{b} = (b_1, \ldots, b_q)$ when we wish to stress its components.

Fix a value q, and let $Q = q^2$. We define $\mathbf{a} \otimes \mathbf{b} \in (\{0,1\}^n)^Q$ as the tensor product under exclusive-or, where we identify $(\{0,1\}^n)^{q \times q}$ with $(\{0,1\}^n)^{q \cdot q} = (\{0,1\}^n)^Q$. More concretely $(\mathbf{a} \otimes \mathbf{b})_{i,j} = a_i \oplus b_j$ for i and j in $[1, \ldots, q]$. (Whenever possible we will try to use dummy i to refer to elements of $\mathbf{a}$ and dummy j to refer to those of $\mathbf{b}$.) If A and B are both distributions over $(\{0,1\}^n)^q$, this tensor operation induces a distribution over $(\{0,1\}^n)^Q$, which we will denote by the symbol $A \boxplus B$. Unless otherwise specified, we will assume throughout that A and B are two distributions induced by sampling from $(\{0,1\}^n)^q$ *without* replacement. We will use U to denote the uniform distribution over $(\{0,1\}^n)^Q$ (where n and Q will often follow from the context). Thus U corresponds to sampling Q strings from $\{0,1\}^n$ uniformly and independently *with* replacement.

If in a random sample some value appears *exactly* k times, we say there is a *k-way collision* in that sample. Let $M_U(k)$ be the random variable describing the number of k-way collisions when the samples are drawn according to the distribution U. Similarly, let $M_{A \boxplus B}(k)$ be the random variable describing the number of k-way collisions when the samples are drawn according to the distribution $A \boxplus B$.

COMPRESSION FUNCTION SECURITY. When algorithms are provided with oracles, we write them as superscripts. A *compression function* is a mapping from $\{0,1\}^n \times \{0,1\}^m$ to $\{0,1\}^n$ for some $m, n > 0$. For us, a compression function H must be given by a program that, given (V, M), computes $H^{\cdots}(V, M)$ via access to a finite number of specified oracles. A *collision-finding adversary* is an algorithm with access to one or more oracles, whose goal it is to find collisions in some specified compression function.

Definition 1 *Let $n, \ell > 0$ be integer parameters. Let $H \colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a compression function taking ℓ oracles. Let $\mathcal{A}$ be a collision-finding adversary for H that takes ℓ oracles. The collision-*

finding advantage of $\mathcal{A}$ is defined to be

$$\mathbf{Adv}^{\mathrm{coll}}_{H(n)}(\mathcal{A}) = \Pr\Big[f_1,\ldots,f_\ell \xleftarrow{\$} \mathrm{Func}(n,n), (V,M),(V',M') \leftarrow \mathcal{A}^{f_1(\cdot),\ldots,f_\ell(\cdot)} : (V,M) \neq (V',M') \text{ and } H^{f_1,\ldots,f_\ell}(V,M) = H^{f_1,\ldots,f_\ell}(V',M')\Big]$$

Furthermore, for $q > 0$, we define $\mathbf{Adv}^{\mathrm{coll}}_{H(n)}(q)$ as the maximum of $\mathbf{Adv}^{\mathrm{coll}}_{H(n)}(\mathcal{A})$ over all adversaries $\mathcal{A}$ making at most q queries.

When the compression function H is defined for arbitrary positive integers n, we will be able to make asymptotic statements about collision resistance.

We consider information-theoretic adversaries in order to make a strong statement about collision resistance. That is, our adversaries are computationally unbounded and their complexity is measured only by the number of queries made to the oracles for the non-compressing primitives. Of course, in practice, time complexity or even time-space complexity (the product of space and time) are arguably more relevant. But query complexity is the most conservative resource measure: it yields a lower bound on the other two measures (and hence the actual costs of mounting an attack). Moreover, finding *lower* bounds in the other two models, other than by query complexity, is notoriously hard. Without loss of generality, we assume that adversaries do not repeat queries to oracles and that they do not query an oracle outside of its specified domain.

3 Some Background on Collision Probabilities

COLLISIONS IN UNIFORM SAMPLES. The susceptibility of hash functions to collisions has been heavily studied. The generic case can be termed in the language of occupancy urn models, a well-known tool from discrete probability theory. (Johnson and Kotz [17] and Feller [12] are the standard references, Girault et al. [14] and Preneel [30, Appendix B] are cryptographically oriented.) We give a brief overview of the main results that are relevant to our work.

One central concern for collision-finding attacks is to determine how many k-way collisions are expected for some fixed k, typically $k = 2$. If the hash function is modelled as a random oracle with range $\{0,1\}^n$, and Q domain points are hashed, then the number of k-way collisions is $M_U(k)$ with expected value $\mathbb{E}\left[M_U(k)\right] = N\binom{Q}{k}\left(\frac{1}{N}\right)^k\left(1-\frac{1}{N}\right)^{Q-k}$, where $N = 2^n$. Thus, $\mathbb{E}[M_U(k)]$ follows a (scaled) binomial distribution with parameters $1/N$ and Q. If we let Q and N grow such that $Q/N < 1$, the binomial distribution can be approximated by a Poisson distribution with parameter $\lambda = \frac{Q}{N}$. This well known "urns and balls" result is captured in the following theorem (by Kolchin [**?**]), which also gives a more general result on the distribution of k-way collisions.

Theorem 2 *Let Q and N be positive integers, and let $\lambda = \frac{Q}{N}$. Then for $k \geq 2$ the random variable $M_U(k)$ follows asymptotically a Poisson distribution with parameter λ_k, where*

$$\Pr[M_U(k) = t] = e^{-\lambda_k}\frac{\lambda_k^t}{t!} \qquad \text{and} \qquad \lambda_k = Ne^{-\lambda}\frac{\lambda^k}{k!}$$

when Q, N tend to infinity such that $\lambda \to 0$.

The expected number of k-way collisions is therefore λ_k (since the parameter of a Poisson distribution equals its expected value). Furthermore, the λ_k themselves are also distributed according to a scaled Poisson

distribution with parameter λ, as expected. The probability of finding any sort of collision is approximately $\frac{Q^2}{2N}$.

Sometimes one is more interested in the expected waiting time before the first k-way collision occurs. If the probability for finding a collision after q queries is at most ϵ, then the expected waiting time is at least $(1-\epsilon)q$. Thus upper bounding collision probabilities simultaneously lower bounds the expected waiting time, which is why we restrict our attention to the former.

In bounding the probability of finding a collision in our compression function, we will also need the distribution of k-way collisions when the samples from $\{0,1\}^n$ are distributed according to $A \oplus B$. It seems that occupancy is very similar to that of a fully uniform distribution, and we conjecture that the occupancy distribution arising from $A \oplus B$ is asymptotically the same as the one from Theorem 2. We could find neither proof nor reference for this conjecture in full, but there is no need to. For the sequel, to bound the collision finding probability either upwards or downwards, the following lemma suffices (proven in Appendix B).

Lemma 3 *For some positive integers q, n, let $Q = q^2$ and $N = 2^n$, and let $\lambda = Q/N$. Let q-vectors $\mathbf{a}$ and $\mathbf{b}$ have elements drawn according to A and B, respectively (i.e., uniformly from $\{0,1\}^n$ without replacement). Then*

1. *Asymptotically $\mathbb{E}[M_{A\oplus B}(k)] = \lambda_k = Ne^{-\lambda}\frac{\lambda^k}{k!}$ when q, n tend to infinity such that $\lambda \to 0$.*
2. *For all $k > 0$ we have*

$$\sum_{\kappa \geq k} \Pr[M_{A\oplus B}(\kappa) > 0] \leq \frac{(q!)^2 2^n (2^n - k)!}{((q-k)!)^2 k! (2^n)!} \; .$$

For notational convenience, the righthand side of the inequality will be denoted by $P_{q,k,n}$.

4 The Rate-1/3 Compression Function

We want to show that H (see Figure 1) is collision-resistant, so that it can be iterated to create a collision-resistant hash function [3, 11, 13, 24]. In the sequel, we will model f_1, f_2, and f_3 as three independent, uniform elements of $\mathrm{Func}(n,n)$, and simply refer to our construction as H^{f_1,f_2,f_3} (or just H when it is not necessary to make the component functions explicit). We proceed to bound the probability that a computationally unbounded adversary can find a collision as a function of the number of its oracle queries. In particular, we will prove the following statement.

Theorem 4 *Fix $n > 0$ and let H^{f_1,f_2,f_3} be as previously defined. Then for all $k, q \geq 0$ we have $\mathbf{Adv}^{\mathrm{coll}}_{H(n)}(q) \leq \frac{q^2}{2^n} + \frac{(kq)^2}{2^n} + P_{q,k,n}$.*

Section 5 is dedicated to a proof of this theorem. What is more, the following lemma provides asymptotic upper and lower bounds on the number of queries required to find collisions. The first item is a corollary of Theorem 4 and we will give a short proof sketch. The second item is proven after the proof of Theorem 4. Lemma 5 can be loosely rephrased by saying that to find collisions with any constant (non-zero) probability, $\tilde{\Theta}(2^{n/2})$ queries are necessary and sufficient.

Lemma 5 *1. For any $c > 1$ and all adversaries making at most $O(2^{n/2}/n^c)$ queries it holds that*

$$\lim_{n\to\infty} \mathbf{Adv}^{\mathrm{coll}}_{H(n)}(\mathcal{A}) = 0 \; .$$

2. *For any $c < 1$ there exists an $\epsilon > 0$ and an adversary asking $\Theta(2^{n/2}/n^c)$ queries for which, under a uniformity assumption,*

$$\mathbf{Adv}^{\mathrm{coll}}_{H(n)}(\mathcal{A}) > \epsilon \qquad \textit{for all sufficiently large } n\ .$$

Proof: The first item is a corollary of Theorem 4. Let d be such that $c > d > 1$ and consider the upper bound from Theorem 4 with $k = n^d$ and $q = 2^{n/2}/n^c$. Then for $n \to \infty$ all three terms tend to zero.

The first term will equal $1/n^{2c}$, whereas substitution in the second term, gives $n^{2(d-c)}$ which indeed both tend to zero iff $c > d > 1$. For the third term, we need to upper bound $P_{q,k,n}$ (Lemma 3). Using Stirling's formula $k! \approx \sqrt{2\pi k}(\frac{k}{e})^k$, substituting $q = 2^{n/2}/n^c$ and $k = n^d$ and taking logarithms leads to:

$$\begin{aligned}
\ln p &\leq (2q+1)\ln q - 2q + n\ln 2 + (2^n - k + \frac{1}{2})\ln(2^n - k) - (2^n - k) \\
&\quad - (2(q-k)+1)\ln(q-k) + 2(q-k) - (2^n + \frac{1}{2})\ln 2^n + 2^n - k\ln k + k + o(1) \\
&\leq 2k\ln qk - k\ln k + n\ln 2 - kn\ln 2 + o(1) \\
&\leq 2k(\frac{n}{2}\ln 2 - c\ln n) - k\ln k + n\ln 2 - kn\ln 2 + o(1) \\
&\leq -(2c+d)n^d\ln n + n\ln 2 + o(1)\ .
\end{aligned}$$

Hence for $d \geq 1$ this logarithm tends to minus infinity, and thus the probability of a k-way collision occuring, and with it the advantage of the adversary in finding a collision in H, tends to zero.

Q.E.D.

In the remainder of this section we build some intuition about the compression function and the necessary requirements on f_1, f_2, f_3 when instantiated in practice. For a classical birthday attack, an attacker would need to evaluate the compression function H on roughly $2^{n/2}$ inputs to succeed. Clearly, one can obtain this many evaluations by querying each of the f_1, f_2, f_3 oracles on this many points. However, the structure of the compression function may make things easier for the adversary. In particular, asking q queries to each of the oracles can provide more than q evaluations of H, due to internal xor-collisions at the input to f_3. In the next section we will introduce the *yield* of a query set, and use it to measure the number of H evaluations an adversary can make given q queries to each of the oracles.

PRACTICAL CONSIDERATIONS. Firstly, we note that in the construction of H any bijection can be applied to the inputs and the output without affecting the collision resistance as bounded by Theorem 4. This freedom might yield a possible avenue to strengthen the hash function when iterating the compression function, without aversely affecting the security of the compression function itself.

Secondly, a collision in either f_1 or f_2 easily leads to a collision on the full compression function. In fact, it is even worse, since a single colliding pair M, M' for f_1 can be used for any chaining value V. That is, if $f_1(M) = f_1(M')$, then for all V it holds that $H(V, M) = H(V, M')$. The precise ramifications of such an attack are unclear, although it for instance allows finding k-way collisions in a standard (strengthened) MD-iterate in query complexity $\Theta(2^{n/2})$, which is an improvement over Joux' [18] $\Theta(2^{n/2}\log k)$. We will not delve into this in great detail; actual attacks based on this property will also crucially rely on the iteration method used (and if multi-collisions are an issue one should not rely on the standard MD-transform). We do note that for instance by changing f_2's input to $V \oplus M$, a collision in f_1 still leads to many collisions in H, but in contrast with the previous version they will be of the form $H(V, M) = H(V', M')$ with $V \neq V'$. This might hinder chaining the collisions when iterated.

Thirdly, from the proof it will be clear that our construction is equally secure if the random functions are replaced by random one-way permutations. However, if either f_1 or f_2 are invertible, an adversary can find collisions in H by making $O(2^{n/4})$ oracle queries. Say that f_2 is invertible. Then the adversary makes $2^{n/4}$ queries to both f_1 and f_3. With reasonable probability this will result in an internal xor-collision $f_1(M) \oplus f_3(Z) = f_1(M') \oplus f_3(Z')$. Inverting f_2 on $Z \oplus f_1(M)$, resp. $Z' \oplus f_1(M')$ will give a collision for H. Similarly if f_1 is invertible, call f_2 and f_3 each $2^{n/4}$ times to find an internal xor-collision $f_2(V) \oplus f_3(Z) \oplus Z = f_2(V') \oplus f_3(Z') \oplus Z'$. Now inverting f_1 on $Z \oplus f_2(V)$ and $Z' \oplus f_2(V')$ will complete the collision. If both f_1 and f_2 are invertible, only two calls to f_3 are needed to find a collision. Thus we will need (at least) for f_1 and f_2 to be collision-resistant and one-way. In particular, this rules out the straightforward blockcipher implementation $f_i(M) = E_{K_i}(M)$ for fixed (distinct) keys K_i, $i \in \{1, 2\}$, as this violates the one-way requirement. Nonetheless, one could instantiate the functions f_1, f_2 with a simple blockcipher-based function. In Appendix C we show that, for example, instantiation as $f_i(X) = E_{K_i}(X) \oplus X$, ($i \in \{1, 2\}$, where K_1 and K_2 are fixed and publicly known keys) gives a collision-resistant compression function in a combined model using a random oracle for f_3 and an ideal cipher for E.

Instantiating f_3 with $f_3(X) = E_{K_3}(X) \oplus X$ is pointless: it essentially results in the original rate-1/3 scheme with the inputs swapped and f_3 replaced with E_{K_3}. Luckily, neither invertibility of f_3 nor collisions in f_3 appear to be useful for finding collisions in H. It would be interesting to see whether our construction can be proven secure (with similar collision resistance), in the ideal cipher model for f_3. Note that invertibility of f_3 is useful for finding preimages, allowing a meet-in-the-middle attack using only $\Theta(2^{n/2})$ (as shown later).

Again, we stress that there is no need to restrict oneself to blockcipher-based implementations of functions f_1, f_2, f_3. Faster alternatives might be available by using for instance streamcipher-based components (cf. [5]).

A potential speedup could be obtained in practice when two of the three functions are identical, in particular if $f_1 = f_2$. We have reasons to believe that this only results in a marginally decreased security (the proof carries through, apart from the bounding of a k-way collision).

5 Proof of Theorem 4 and Lemma 5

In this section we will prove Theorem 4 and the second item of Lemma 5. Following the proof, we will also give some intuition why one expects that $\tilde{\Theta}(2^{2n/3})$ queries are necessary and sufficient to find preimages. This provides a fairly complete asymptotic characterization of the newly proposed construction.

SETTING UP THE PROOF. We will distinguish between three ways for an adversary to find a collision in H. It can try to find a collision in f_1 or f_2, since either would lead to a collision in H, as already shown above. Failing that, it can try to find a collision in the final output. This leads to the following upper bound

$$\begin{aligned}\mathbf{Adv}^{\mathrm{coll}}_{H(n)}(\mathcal{A}) \leq\ & \Pr[\mathcal{A} \text{ finds collision in } f_1] + \Pr[\mathcal{A} \text{ finds collision in } f_2] \\ & + \Pr[\mathcal{A} \text{ finds collision in } H | \text{no collisions in } f_1 \text{ or } f_2]\end{aligned}$$

and the corresponding lower bound

$$\max \begin{pmatrix} \Pr[\mathcal{A} \text{ finds collision in } f_1], \\ \Pr[\mathcal{A} \text{ finds collision in } f_2], \\ \Pr[\mathcal{A} \text{ finds collision in } H | \text{ no collisions in } f_1 \text{ or } f_2] \end{pmatrix} \leq \mathbf{Adv}^{\mathrm{coll}}_{H(n)}(\mathcal{A}) \ .$$

The probabilities of finding a collision in f_1 or f_2 are ordinary collision-finding problems and hence well understood. For $q \leq \sqrt{N}$ these probabilities roughly sum up to (and are upper bounded by) $\frac{q^2}{N}$, the first term in the upper bound of Theorem 4. Needless to say, if f_1 and f_2 are (random) permutations, no collisions exist and both probabilities are always zero. In any case, we can concentrate on the probability of $\mathcal{A}$ finding a collision in H when f_1 and f_2 are collision free. Henceforth we will therefore model f_1 and f_2 as random (one-way) permutations, but f_3 still as a random (one-way) function.

We also make the following standard assumptions, all without loss of generality. Firstly, we assume that the adversary makes exactly q queries to each of the three oracles, f_1, f_2, f_3: for any adversary that makes q_i queries to f_i there is an adversary that makes $q = \max(q_1, q_2, q_3)$ queries to *each* of the oracles with identical success probability. Secondly, we will assume that adversaries actually compute $H^{f_1,f_2,f_3}(V, M)$ and $H^{f_1,f_2,f_3}(V', M')$ before outputting their candidate collisions. In particular, this means that all necessary queries to f_1, f_2 and f_3 are made before halting.

REMOVING THE ADVERSARY. Normally we would imagine that the adversary makes queries to all three oracles in some adaptive, probabilistic manner. But here we cannot only argue away the adversary's adaptivity, but we can remove the adversary altogether. Recall that f_1, f_2, f_3 are independent oracles, and let us focus for a moment on the adversary's queries to f_1 and f_2.

Knowing that $\mathcal{A}$ makes q queries to each, we can imagine preparing the answers in advance. That is, before the adversary starts querying the oracles, we make two lists, each of q random elements, and when the adversary makes a query to one of the two oracles f_1 or f_2, we supply it with the next element of the respective list. Because the inputs to f_1 and f_2 are not used elsewhere in the compression function, the actual correspondence between query and response is irrelevant. Consequently, we might as well have provided the two lists to the adversary *before* any queries to f_1 or f_2, and even at the very beginning of the collision-finding game, in advance of any f_3 queries, given f_3's independence of f_1 and f_2.

DEFINING THE YIELD. A central quantity in bounding an adversary's succes is what we call the *yield*. Formally, given a vector $\mathbf{c} = (c_1, \ldots, c_Q) \in (\{0,1\}^n)^Q$, define

$$\text{yield}(\mathbf{c}) = \max_{\substack{G \subseteq \{0,1\}^n \\ |G|=q}} \sum_{g \in G} \sum_{i=1}^{Q} [c_i = g]$$

where $[\text{true}] = 1$ and $[\text{false}] = 0$. Thus the yield counts the total number of occurences of the q most frequent elements in a vector. We also define the yield over the tensor of two q-vectors. Given vectors $\mathbf{a} = (a_1, \ldots, a_q)$ and $\mathbf{b} = (b_1, \ldots, b_q)$ in $(\{0,1\}^n)^q$, we will define the yield of tensor $\mathbf{a} \otimes \mathbf{b}$ to be

$$\text{yield}(\mathbf{a} \otimes \mathbf{b}) = \max_{\substack{G \subseteq \{0,1\}^n \\ |G|=q}} \sum_{g \in G} \sum_{i=1}^{q} \sum_{j=1}^{q} [a_i \oplus b_j = g] \; .$$

Let us give some intuition for this latter definition, in particular. Recall that we will give the response lists of f_1 and f_2, call these $\mathbf{a} = (a_1, \ldots, a_q)$ and $\mathbf{b} = (b_1, \ldots, b_q)$ (resp.), to the adversary prior to its making any f_3 queries. These q queries to f_3 can be made according to any strategy. One such strategy, already mentioned in the previous section, is to query the f_3 oracle on those values for which it knows the greatest total number of xor-preimages, thereby maximizing the number of inputs for which it can evaluate the compression function. It is precisely this number that the yield of $\mathbf{a} \otimes \mathbf{b}$ represents: the number of compression function outputs that the adversary can evaluate by asking q queries to f_3.

CONNECTING THE PIECES. We still need to show how the yield relates to the collision-finding probability of the adversary. Let d_r, for $r = 1, \ldots, q$, denote the number of pairs (i, j) such that $a_i \oplus b_j$ equals the input of the r-th query to f_3. Suppose that after $r - 1$ queries to f_3, the adversary still has not found a collision. Then it will be able to output preimages for $\sum_{s=1}^{r-1} d_s$ hash values (or, equivalently, it will be able to output the hash value for that many preimages). With its rth call to f_3 it will be able to evaluate d_r new hash values, and the probability that one collides with one of the older values is therefore upper bounded by $d_r \sum_{s=1}^{r-1} d_s/2^n$. The upper bound is not always tight. Consequently, picking the elements corresponding to the maximal yield is sometimes not the optimal strategy for finding a collision; picking elements that are slightly less common might actually increase the chances of finding a collision, nonetheless the same upper bound will apply.

Summing over all queries to f_3 leads us to the following upper bound

$$\Pr[\mathcal{A} \text{ finds collision in } H | \text{ no collisions in } f_1 \text{ or } f_2] \leq \sum_{r=1}^{q} \sum_{s=1}^{r-1} d_r d_s / 2^n \ .$$

What can we say about this value? Firstly, the possible values of d_r are determined by $\mathbf{a}$ and $\mathbf{b}$ and the maximum $\sum_{s=1}^{q} d_s = \mathrm{yield}(\mathbf{a} \otimes \mathbf{b})$. Suppose we allow the adversary to partition $\mathrm{yield}(\mathbf{a} \otimes \mathbf{b})$ arbitrarily in q (real) parts d_r. The optimal way, in the sense of maximizing the sum above, is then to choose $d_r = \mathrm{yield}(\mathbf{a} \otimes \mathbf{b})/q$ for all $r = 1, \ldots, q$ (optimality of this choice can be shown by induction). In that case we have

$$\begin{aligned} \Pr[\mathcal{A} \text{ finds collision in } H | \text{ no collisions in } f_1 \text{ or } f_2] &\leq \sum_{r=1}^{q} \sum_{s=1}^{r-1} d_r d_s / 2^n \\ &\leq \sum_{r=1}^{q} \sum_{s=1}^{r-1} (\mathrm{yield}(\mathbf{a} \otimes \mathbf{b})/q)^2 / 2^n \\ &\leq \mathrm{yield}(\mathbf{a} \otimes \mathbf{b})^2 / 2^{n+1} \ . \end{aligned}$$

Moreover, if we would assume that the compression function outcomes for an adversary optimizing its yield are uniformly distributed, the probability that a collision occurs will satisfy the birthday bound, that is $\mathbf{Adv}^{\mathrm{coll}}_{H(n)}(\mathcal{A}) \approx 0.63 \cdot \mathrm{yield}(\mathbf{a} \otimes \mathbf{b})^2 / 2^{n+1}$ giving us nearly matching upper and lower bounds. Our task then is to put bounds on the expected value of $\mathrm{yield}(\mathbf{a} \otimes \mathbf{b})$, or rather its square, where the elements in $\mathbf{a}$ and $\mathbf{b}$ are chosen independently, uniformly at random from $\{0, 1\}^n$ (without replacement).

To upper bound the yield we recall that it is the sum of the frequencies of the q most frequent elements in $\mathbf{a} \otimes \mathbf{b}$. As such, the trivial upper bound on the yield is the cardinality of $\mathbf{a} \otimes \mathbf{b}$, that is $Q = q^2$. Moreover, if all collisions in $\mathbf{a} \otimes \mathbf{b}$ are less than k-way, then the yield is (strictly) smaller than kq. We can combine the two bounds as well. Let p be an upper bound on the probability that at least one collision that is at least k-way occurs in $\mathbf{a} \otimes \mathbf{b}$. Then conditioning on this event and employing the above observations yields that

$$\Pr[\mathcal{A} \text{ finds collision in } H | \text{ no collisions in } f_1 \text{ or } f_2] \leq (kq)^2 / 2^n + p \ .$$

By Lemma 3 we can use $P_{q,k,n}$ as our upper bound p. This concludes the proof of the upper bound of an adversary's advantage.

To lower bound the adversary's advantage, we need to lower bound the expected yield. We will only do this asymptotically, proving the second item of Lemma 5. Given $c < 1$, let d be such that $c < d < 1$ and

consider the expected number of k-way collisions for $k = n^d$. By Lemma 3, this number is given by λ_k. We will show that λ_k/q tends to infinity for increasing n. Now, since

$$\lambda_k/q = \frac{2^{n/2}(\frac{1}{n^{2c}})^{n^d} n^c}{(n^d)!}$$

we can take logarithms, use Stirling's formula to approximate the factorial, and ignore smaller order terms, thus arriving at

$$\ln(\lambda_k/q) \approx \frac{\ln 2}{2} n - (2c + d) n^d \ln n \; .$$

Since $d < 1$ the term $\frac{\ln 2}{2} n$ will eventually dominate, showing that the expected number of k-way collisions asymptotically exceeds q. Consequently the yield is at least $kq = 2^{n/2} n^{d-c}$ and the probability of finding a collision tends to one.

A NOTE ON PREIMAGE RESISTANCE. Although our goal is to demonstrate a construction that yields a compression function with good collision resistance, other useful properties should also be mentioned. Ideally, finding a preimage takes expected time 2^n for an n-bit primitive. To get an idea of the preimage resistance of the current proposal, we can look at the value of q for which the yield is around 2^n. If $q > 2^{n/2}$, a lower bound (and reasonable estimate) for the yield is $q^3/2^n$. Since $q^3/2^n \approx 2^n$ for $q \approx 2^{\frac{2}{3}n}$ it follows that our construction is not as preimage resistant as one might wish for. However, Rogaway and Steinberger [34] recently showed that this reduced preimage resistance is all but inevitable. Indeed, for any rate-1/3 scheme there exists an adversary whose yield is at least 2^n after $2^{2n/3}$ queries, which will likely lead to a preimage.

When f_3 is a random two-way permutation instead of a random one-way function, finding preimages becomes easier due to a meet-in-the-middle attack pointed out to us by Antoine Joux. Given a target h, the adversary queries f_1 on $q = 2^{n/2}$ arbitrary values leading to $f_1(V_1), \ldots, f_1(V_q)$ and subsequently queries f_3^{-3} on the values $h \oplus f_1(V_i)$ for $i = 1, \ldots, q$. After querying f_2 on q arbitrary values (leading to $f_2(M_1), \ldots, f_2(M_q)$) a preimage is obtained if $f_2(M_i) = f_3^{-1}(h \oplus f_1(V_j))$ which will occur with probability $q^2/2^n \approx 1$. One could defeat the meet-in-the-middle attack for instance by feed-forwarding M as well, so $H'(M, V) = M \oplus H(M, V)$. Unfortunately this tweak also destroys our proof of collision resistance, even though we expect H' to be as collision resistant as H. In a way, we need to compromise on some security properties in order for our proof of collision-resistance to go through (this is not unlike certain public key schemes where efficiency is sacrificed for security proofs to go through).

POISSON HEURISTIC TO APPROXIMATE THE YIELD. In Appendix D we develop an alternative characterization of yield$(\mathbf{a} \otimes \mathbf{b})$ and recast the problem of finding the expected value $\text{yield}_n^{A \oplus B}(q)$ into that of determining a certain property of the tail of a Poisson distribution. Experimental results give concrete estimates of the collision resistance of our proposal in practice, and it turns out that for n up to 256, the loss in collision resistance is at most four bits.

6 Conclusion

In this paper we have proposed a rate-1/3 $2n$-to-n bit compression function based on three random n-to-n bit functions. If the three underlying functions are modelled as random oracles, finding collisions requires roughly $2^{n/2}/n$ queries. Preimage resistance is loosely estimated to be around $2^{2n/3}$. Since the attacks based on optimizing the yield are inherently time and space consuming, it is unclear whether in practice algorithms

can be found with a time complexity matching these query complexities (meaning our scheme will be harder to break).

ACKNOWLEDGEMENTS. We would like to thank Antoine Joux, Thomas Ristenpart and David Wagner for clarifying discussions. We also thank the anonymous reviewers for their feedback. Much of this paper was developed while the first author was visiting LACAL at EPFL, and he thanks them for their hospitality. Also, he was supported in part by NSF grant CNS-0627752.

References

1. W. Aiello and R. Venkatesan. Foiling birthday attacks in length-doubling transformations - Benes: A non-reversible alternative to Feistel. In *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *LNCS*, pages 307–320. Springer-Verlag, 1996.
2. M. Bellare and D. Micciancio. A new paradigm for collision-free hashing: incrementality at reduced cost. In *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *LNCS*, pages 163–192. Springer-Verlag, 1997.
3. M. Bellare and T. Ristenpart. Multi-property-preserving hash domain extension and the EMD transform. In *Advances in Cryptology – ASIACRYPT'06*, volume 4284 of *LNCS*, pages 299–314. Springer-Verlag, 2006.
4. M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Advances in Cryptology – EUROCRYPT'06*, volume 4004 of *LNCS*, pages 409–426. Springer-Verlag, 2006.
5. D. Bernstein. The Rumba20 compression function. http://cr.yp.to/rumba20.html, 2007.
6. J. Black, M. Cochran, and T. Shrimpton. On the impossibility of highly efficient blockcipher-based hash functions. In *Advances in Cryptology – EUROCRYPT '05*, volume 3494 of *LNCS*, pages 526–541. Springer-Verlag, 2005.
7. 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 *LNCS*. Springer-Verlag, 2002.
8. D. Chang, S. Lee, M. Nandi, and M. Yung. Indifferentiable security analysis of popular hash functions with prefix-free padding. In *Advances in Cryptology – ASIACRYPT'06*, volume 4284 of *LNCS*, pages 283–289. Springer-Verlag, 2006.
9. J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Dåmgard revisited: How to construct a hash function. In *Advances in Cryptology – CRYPTO '05*, volume 3621 of *LNCS*, pages 430–448. Springer-Verlag, 2005.
10. I. Damgård. Collision free hash functions and public key signature schemes. In D. Chaum and W. L. Price, editors, *Advances in Cryptology – EUROCRYPT '87*, volume 304 of *LNCS*, pages 203–216. Springer-Verlag, 1988.
11. I. Damgård. A design principle for hash functions. In G. Brassard, editor, *Advances in Cryptology – CRYPTO '89*, volume 435 of *LNCS*. Springer-Verlag, 1990.
12. W. Feller. *An Introduction to Probability Theory and its Applications*, volume 1. John Wiley and Sons, Inc., 1968.
13. P. Gauravaram, W. Millan, E. Dawson, and K. Viswanathan. Constructing secure hash functions by enhancing Merkle-Damgård construction. In *Information Security and Privacy*, volume 4058 of *LNCS*, pages 407–420. Springer-Verlag, 2006.
14. M. Girault, R. Cohen, and M. Campana. A generalized birthday attack. In C. G. Guenther, editor, *Advances in Cryptology – EUROCRYPT '88*, volume 330 of *LNCS*, pages 129–156. Springer-Verlag, 1988.
15. B. Gladman. Implementation experience with AES candidate algorithms. In *Second AES Conference*, 1999.
16. S. Hirose. Provably secure double-block-length hash functions in a black-box model. In *Information Security and Cryptology – ICISC '04*, volume 3506 of *LNCS*, pages 330–342. Springer-Verlag, 2005.
17. N. L. Johnson and S. Kotz. *Urn Models and Their Applications*. John Wiley and Sons, Inc., 1977.
18. A. Joux. Multicollisions in iterated hash functions. Application to cascaded constructions. In M. K. Franklin, editor, *Advances in Cryptology – CRYPTO '04*, volume 3621 of *LNCS*, pages 306–316. Springer-Verlag, 2004.
19. L. Knudsen. SMASH: A cryptographic hash function. In *Fast Software Encryption – FSE'05*, volume 3557 of *LNCS*, pages 228–242. Springer-Verlag, 2005.
20. L. Knudsen and F. Muller. Some attacks against a double length hash proposal. In *Advances in Cryptology – ASIACRYPT'06*, volume 4284 of *LNCS*, pages 462–473. Springer-Verlag, 2006.
21. S. Matyas, C. Meyer, and J. Oseas. Generating strong one-way functions with cryptographic algorithms. *IBM Technical Disclosure Bulletin*, 27(10a):5658–5659, 1985.
22. U. Maurer and S. Tessaro. Domain extension of public random functions: Beyond the birthday barrier. In *Advances in Cryptology – CRYPTO '07*, volume 4622 of *LNCS*, pages 187–204. Springer-Verlag, 2007.
23. A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
24. R. Merkle. One way hash functions and DES. In G. Brassard, editor, *Advances in Cryptology – CRYPTO '89*, volume 435 of *LNCS*, pages 428–466. Springer-Verlag, 1990.

25. I. Mironov and A. Narayanan. Domain extension for random oracles: Beyond the birthday-paradox bound. ECRYPT Hash Workshop 2007, May 24–25, Barcelona, 2007.
26. S. Miyaguchi, M. Iwata, and K. Ohta. New 128-bit hash function. In *Proceedings 4th International Joint Workshop on Computer Communications*, pages 279–288, 1989.
27. M. Nandi, W. Lee, K. Sakurai, and S. Lee. Security analysis of a 2/3-rate double length compression function in black-box model. In *Fast Software Encryption – FSE'05*, volume 3557 of *LNCS*, pages 243–254. Springer-Verlag, 2005.
28. T. Peyrin, H. Gilbert, F. Muller, and M. Robshaw. Combining compression functions and block cipher-based hash functions. In *Advances in Cryptology – ASIACRYPT'06*, volume 4284 of *LNCS*, pages 315–331. Springer-Verlag, 2006.
29. N. Pramstaller, C. Rechberger, and V. Rijmen. Breaking a new hash function design strategy called SMASH. In *Selected Areas in Cryptography – SAC'06*, volume 3897 of *LNCS*, pages 233–244. Springer-Verlag, 2006.
30. B. Preneel. *Analysis and design of cryptographic hash functions*. PhD thesis, Katholieke Universiteit Leuven, 1993.
31. B. Preneel, R. Govaerts, and J. Vandewalle. On the power of memory in the design of collision resistant hash functions. In *Advances in Cryptology – Auscrypt '92*, volume 718 of *LNCS*, pages 105–121. Springer-Verlag, 1992.
32. B. Preneel, R. Govaerts, and J. Vandewalle. Hash functions based on block ciphers: A synthetic approach. In *Advances in Cryptology – CRYPTO '93*, volume 773 of *LNCS*, pages 368–378. Springer-Verlag, 1994.
33. P. Rogaway and J. Steinberger. Constructing cryptographic hash functions from fixed-key blockciphers. In *Advances in Cryptology – CRYPTO '08*, volume 5157 of *LNCS*, pages 433–450. Springer-Verlag, 2008.
34. P. Rogaway and J. Steinberger. Security/efficiency tradeoffs for permutation-based hashing. In *Advances in Cryptology – EUROCRYPT '08*, volume 4965 of *LNCS*, pages 220–236. Springer-Verlag, 2008.
35. Y. Seurin and T. Peyrin. Security analysis of constructions combining FIL random oracles. In *Fast Software Encryption (FSE'07)*, volume 4593 of *LNCS*, pages 119–136. Springer-Verlag, 2007.
36. T. Shrimpton and M. Stam. Efficient collision-resistant hashing from fixed-length random oracles. ECRYPT Hash Workshop 2007, May 24–25, Barcelona, 2007.
37. T. Shrimpton and M. Stam. Building a collision-resistant compression function from non-compressing primitives. In *ICALP 2008, Part II*, volume 5126, pages 643–654. Springer-Verlag, 2008.
38. M. Stam. Beyond uniformity: Better security/efficiency tradeoffs for compression functions. In *Advances in Cryptology – CRYPTO '08*, volume 5157 of *LNCS*, pages 397–412. Springer-Verlag, 2008.
39. J. Steinberger. The collision intractability of MDC-2 in the ideal-cipher model. In *Advances in Cryptology – EUROCRYPT'07*, volume 4515 of *LNCS*, pages 34–51. Springer-Verlag, 2007.
40. D. Wagner. A generalized birthday problem. In M. Yung, editor, *Advances in Cryptology – CRYPTO '02*, volume 2442 of *LNCS*, pages 288–303. Springer-Verlag, 2002.

A The Dual Relationship with Double-Length Constructions (or Range Extenders)

In the main body we have considered the problem of extending the domain of a cryptographic primitive. One could consider *range extension* as the dual problem to domain extension. A relevant example instance of this problem is:

> Given a $2n$-to-n bit compression function create a $4n$-to-$2n$ bit compression function with collision resistance close to the optimal 2^n.

Perhaps a more common approach is to take as the starting primitive a blockcipher operating on n-bit blocks and having keysize (some multiple of) n; see, for example, Peyrin et al. [28]. (This has the practical advantage that one can use blockciphers of 128-bit blocks, which would not be advisable if a square root attack running in time 2^{64} were possible.)

When hash functions are expected to behave as random oracles, Maurer and Tessaro [22] make the point that domain extension is much harder than range extension. There is an easy counting argument underlying this claim (extending the domain increases the number of possible functions far more than a comparable extension of the range). However, in the context of collision-resistant hashing some other issues come to the fore. The difficulty in range extension is that the new construction should satisfy considerably stronger collision-resistance bound than the underlying primitive. Said another way, why bother with range extension if the longer hash value does not provide comensurately better security against generic (read birthday)

attacks? As a result, satisfactory bounds on the collision resistance of double-length hash functions have been elusive. For example, Steinberger [39] showed that the collision-resistance of the standardized MDC-2 construction is at least $\Omega(2^{3n/5})$ in the ideal cipher model, when iterated; still considerably shy of the desired $\Theta(2^n)$. Mironov and Narayanan [25] recently claimed that if the MMO constructions in MDC-2 are replaced with random $2n$-to-n bit compression functions, the double length construction does have collision resistance $\Theta(2^n)$.

Somewhat paradoxically, range extenders can also be used to solve the problem of building a compressing function out of non-compressing primitives, and vice versa. As such, they can truly be regarded as dual problems. The resulting constructions are not necessarily very elegant, however, as our two transformations below will show. If applied to our constuction (see Figure 2) one needs six calls to process two blocks of n-bits (of message) simultaneously, making it rate-1/3. It inherits the collision resistance from Theorem 4, so one provably needs almost $2^n/2n$ queries to find collisions. This compares favourably with Nandi et al.'s construction of rate 1/3 and collision-resistance at most $\Theta(2^{2n/3})$ [20, 27] or Peyrin et al.'s construction of rate 1/5 and collision-resistance $\Theta(2^{2n/3})$ [28, 35].

Let $H^{\cdots} : \{0,1\}^{2n} \to \{0,1\}^n$ be some construction to build an optimally collision resistant compression function out of non-compressing primitives $f_1, \ldots, f_k : \{0,1\}^n \to \{0,1\}^n$, for arbitrary n. For example, our triple function construction is such an H for $k = 3$. Now, let $g_1, \ldots, g_{2k} : \{0,1\}^{2m} \to \{0,1\}^m$ be some given (single length) compression functions; we desire to use these $2k$ functions to build a double length compressing function $G^{\cdots} : \{0,1\}^{4m} \to \{0,1\}^{2m}$. We do so as follows. Set $n = 2m$ and define $f_i = g_i || g_{2k+1-i}$ for $i \in \{1, \ldots, k\}$. If we model the g_i as $2m$-to-m bit, independent random oracles then the f_i are $2m$-to-$2m$ bit (i.e., non-compressing) random oracles. Now we apply H to these f_i to obtain a $4m$-to-$2m$ bit compression function, ultimately based on the g_i, whose collision resistance is 2^m, as desired.

Conversely (and for even k and n), let $G^{\cdots} : \{0,1\}^{4m} \to \{0,1\}^{2m}$ be an optimally collision resistant double length combiner for primitives $g_1, \ldots, g_k : \{0,1\}^{2m} \to \{0,1\}^m$. Let $f_1, \ldots, f_{k/2} : \{0,1\}^n \to \{0,1\}^n$ be given. Again, let $n = 2m$ and define the g_i such that $f_i = g_i || g_{k+1-i}$ for $i \in \{1, \ldots, k/2\}$. Run the range extender G with components g_i, yielding a hash function $G : \{0,1\}^{2n} \to \{0,1\}^n$ with collision resistance $2^{n/2}$. (Note that in this construction one should expect to throw half of f's output away all the time.)

B Proof of Lemma 3

Lemma 6 *For some positive integers q, n, let $Q = q^2$ and $N = 2^n$, and let $\lambda = Q/N$. Let q-vectors* $\mathbf{a}$ *and* $\mathbf{b}$ *have elements drawn according to A and B, respectively (i.e., uniformly from $\{0,1\}^n$ without replacement). Then*

1. *Asymptotically $\mathbb{E}[M_{A \oplus B}(k)] = \lambda_k = Ne^{-\lambda}\frac{\lambda^k}{k!}$ when q, n tend to infinity such that $\lambda \to 0$.*
2. *for all $k > 0$ we have*

$$\Pr[M_{A \oplus B}(k) > 0] \leq \frac{(q!)^2 2^n (2^n - k)!}{((q-k)!)^2 k! (2^n)!} .$$

Proof: Define $F_{x,k}$ as the binary random variable taking on 1 iff the value $x \in \{0,1\}^n$ occurs exactly k times (when sampling from $A \oplus B$). This variable is related to the variable G_x denoting how often x occurs, indeed $\mathbb{E}(F_{x,k}) = \Pr(G_x = k)$. Let us try to determine $\Pr(G_x = k)$. This probability is the same for all x, so we can concentrate on $\Pr(G_0 = k)$. Since we also have that $M_{A \oplus B}(k) = \sum_x F_{x,k}$ and linearity of

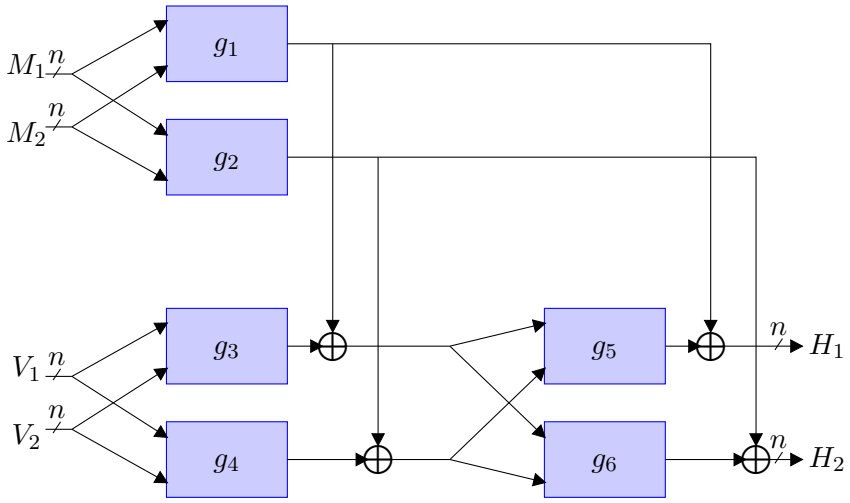

Fig. 2: The rate-1/3 double-length hash function. The functions $g_1, g_2, g_3, g_4, g_5, g_6$ are random $2n$-to-n bit functions.

expectancy (even over dependent variables), we get

$$\mathbb{E}(M_{A\oplus B}(k)) = \sum_x \mathbb{E}(F_{x,k}) = N \Pr[G_0 = k] \; .$$

A collision occurs iff $a_i \oplus b_j = 0^n$, where a_i is an element in the vector $\mathbf{a}$ and b_j in the vector $\mathbf{b}$; equivalently, $a_i = b_j$. This problem has been studied, for example in the context of meet-in-the-middle attacks [14], and it turns out that the probability of a k-way collision follows a hypergeometric distribution, thus $\Pr(G_0 = k) = \binom{q}{k}\binom{N-q}{q-k}/\binom{N}{q}$. Asymptotically this means that $\mathbb{E}(M_{A\oplus B}(k))$ behaves as a scaled Poisson distribution with parameter Q/N, as claimed.

For the second part, we need to upper bound p, the probability of a k-way collision in $\mathbf{a} \otimes \mathbf{b}$ when $\mathbf{a}$ and $\mathbf{b}$ are drawn independent of each other and both uniformly at random from $\{0,1\}^n$ without replacement. Now suppose that we have a k-way collision, then we can look at the positions in the matrix $\mathbf{a} \otimes \mathbf{b}$ that contribute to this collision. Suppose both (i,j) and (i',j') are involved, so $a_i \oplus b_j = a_{i'} \oplus b_{j'}$. We claim that if $i = i'$ (or $j = j'$), then also $(i,j) = (i',j')$. Clearly, if $i = i'$ then also $a_i = a_{i'}$ and, because both pair of indices xor to the same value, $b_j = b_{j'}$. But $\mathbf{b}$ does not contain any collisions, thus $j = j'$.

This allows us to bound the probability on a k-way collision. Firstly, we need to choose the k indices (i,j) that lead to the collision. For the indices i and j there are $\binom{q}{k}$ possibilities each, moreover given k indices i and k indices j, there are $k!$ ways of hooking them up. Given the locations of the collision, we have 2^n different values the collision can take. Then the values of a_i are still unrestricted, but the values of b_j are now determined. Without restrictions, there would have been $N!/(N-q)!$ ways to choose $\mathbf{b}$, but this now reduces to $(N-k)!/(N-q)!$. This leads us to:

$$p \leq \binom{q}{k}^2 k! 2^n \frac{(2^n-k)!}{(2^n)!} = \frac{(q!)^2 2^n (2^n-k)!}{((q-k)!)^2 k! (2^n)!} \; .$$

Q.E.D.

C Instantiating f_1 and f_2 in the Ideal Cipher Model

In this appendix we will consider the Davies-Meyer like instantiation $f_1(X) = E_{K_1}(X) \oplus X$ and $f_2(X) = E_{K_2}(X) \oplus X$ (with $K_1 \neq K_2$). These are optimally inversion and collision-resistant in the ideal cipher model [7]. They are not, however, indifferentiable from a random oracle, even when the blockcipher E is modelled as an ideal cipher [9] (nor are the other 19 provably collision-resistant constructions [8]). As a result, it is not immediate that our bounds on the collision resistance of the compression function will still hold. We will show that if E is modelled as an ideal cipher (and f_3 as a random oracle as before) the proof goes through with only some minor modifications to show that the oracle acces to both E_{K_1} and its inverse will not help the adversary to get a list of outputs of f_1 that significantly deviates from the list used in the original proof (where f_1 was modelled by a random oracle). A central tool for this argument it the Permutation-Pseudo-Permutation (PPP) Switching Lemma below.

A pseudopermutation consists of a pair of oracles E and D with joint state, where D can be thought of as the inverse of E. Essentially, E and D implement *dependent* random functions. Specifically, there is a single, finite set S that serves as both domain and range of both E and D. The joint state consists of a list of pairs $(x, y) \in S \times S$ such that $y = E(x)$ and $x = D(y)$. Initially the list is empty. When E is queried on a point x that does not appear in the list (rather there is no y for which (x, y) is on the list), it picks a random value y in S, adds (x, y) to the list and returns y. If E is queried on a point x and there exists a y such that (x, y) is in the list, this y is returned (this is done uniformly if several y satisfy this criterion). Similary, if D is queried on y, it returns a random element from S unless there exists a pair (x, y) in the list; in this case, D returns x (uniformly if necessary).

Considered separately, both D and E would (perfectly) implement a random function. The joint state ensures that D and E are consistent with respect to being each others inverse, however it is not sufficient to ensure both end up being permutations. The PPP Switching Lemma states that as long as no collision is found (that is $x \neq x'$ such that $E(x) = E(x')$ or $y \neq y'$ such that $D(y) = D(y')$) a random pseudopermutation is indistinguishable from a true random permutation and its inverse. Moreover, the probability of finding a collisions as expressed in the number of queries made adheres to the birthday paradox.

Lemma 7 (PPP Switching Lemma) *Let S be a finite set of cardinality N. Let π be a random permutation on S, and let π^{-1} be its inverse. Let the pair E, D be a random pseudopermutation on S. Let $\mathcal{A}$ be an adversary with an interface for two oracles making at most q queries in total. Moreover, assume that $\mathcal{A}$ is query-respecting. Then the PPP-distinguishing advantage of $\mathcal{A}$ is upper bounded as follows:*

$$|\Pr[\mathcal{A}^{\pi,\pi^{-1}} = 1] - \Pr[\mathcal{A}^{E,D} = 1]| \leq \frac{q^2}{2N} .$$

Proof: We refer to Figure 3. Let $\mathcal{A}^{G0}$ denote the experiment of running adversary $\mathcal{A}$ with oracles described by game $G0$, and similarly let $\mathcal{A}^{G1}$ denote the experiment of running $\mathcal{A}$ with oracles described by game $G1$. We claim that $\Pr[\mathcal{A}^{\pi,\pi^{-1}} = 1] = \Pr[\mathcal{A}^{G0} = 1]$ and $\Pr[\mathcal{A}^{E,D} = 1] = \Pr[\mathcal{A}^{G1} = 1]$.

For the latter, the proper simulation of E and D is clear by inspection. Note that since lines 16 and 25 do not exist in game $G1$, if $\mathcal{O}_1$ is asked x and the test on line 10 is false, then a random value y is returned and (x, y) is added to the list. On the other hand, if the test is true, lines 11-12 ensure that $\mathcal{O}_1$ returns a uniform y from satisfying pairs (x, y) in the list. Likewise, if $\mathcal{O}_2$ is asked y and the test on line 20 is false, then a random value x is returned and (x, y) is added to the list. If the test is true, lines 21-22 ensure that a uniform x from the satisfying pairs (x, y) is returned.

Arguing that $\Pr[\mathcal{A}^{\pi,\pi^{-1}} = 1] = \Pr[\mathcal{A}^{G0} = 1]$ is less obvious. In particular, a permutation π should never have a set Y on line 11 (or X on line 21) with more than one element. We must argue that this is the

case. If $|Y| > 1$, then there are some two pairs $(x_0, y), (x_0, y')$ in L. Notice these cannot appear in the list as a result of two $\mathcal{O}_1$ queries: asking x_0 the first time will add (x_0, y) to the list (at line 17), and asking x_0 a second time will return y without adding any pairs to the list; that is, asking x_0 twice to $\mathcal{O}_1$ will not result in pairs $(x_0, y), (x_0, y')$ being added to L. Similarly, these cannot appear in the list as a result of two $\mathcal{O}_2$ queries. So it must be that (x_0, y) and (x_0, y') are added to the list as a result of a query to $\mathcal{O}_1$ and a query to $\mathcal{O}_2$.

- Say that x_0 is asked to $\mathcal{O}_1$ and that this causes a new pair (x_0, y) to be added to the list. A subsequent query of y' ($\neq y$) to $\mathcal{O}_2$ will never add (x_0, y') to the list: if line 23 selects x_0 , then line 24 will catch this and resample from points not yet assigned as x values.
- Say that y is is asked to $\mathcal{O}_2$ and that this causes a new pair (x_0, y) to be added to the list. A subsequent query of x_0 to $\mathcal{O}_1$ will simply return y, and not add a new pair to the list.

So, in game G0, the set Y will never contain more than 1 element. A similar argument shows that the set X will never contain more than 1 element. Moreover, the (random) resampling on lines 16 and 25 ensures that both oracles observe permutivity, and so $\mathcal{O}_1$ and $\mathcal{O}_2$ simulate a random permutation and its inverse in game G0. Thus,

$$\left|\Pr[\mathcal{A}^{\pi,\pi^{-1}} = 1] - \Pr[\mathcal{A}^{E,D} = 1]\right| = \left|\Pr[\mathcal{A}^{G0} = 1] - \Pr[\mathcal{A}^{G1} = 1]\right|.$$

Notice now that games G0 and G1 are identical-until-BAD, as defined in [4], and so we can state that

$$\left|\Pr[\mathcal{A}^{G0} = 1] - \Pr[\mathcal{A}^{G1} = 1]\right| \leq \Pr[\mathcal{A}^{G1} \text{sets BAD to } \mathsf{true}]$$

It remains to bound the probability that BAD is set in game G1. But this is almost immediate, since each oracle query adds at most one new pair (x, y) to the list. Thus the probability that BAD is set on the i-th oracle query is at most $(i-1)/N$, and a union bound gives us that $\Pr[\mathcal{A}^{G1} \text{ sets BAD to } \mathsf{true}] \leq q^2/2N$. *Q.E.D.*

procedure $\mathcal{O}_1(x)$:		**procedure** $\mathcal{O}_2(y)$:	G0 G1
10	**if** $\exists y$ such that $(x, y) \in L$ **then**	20	**if** $\exists x$ such that $(x, y) \in L$ **then**
11	Let $Y = \{y \mid (x, y) \in L\}$	21	Let $X = \{x \mid (x, y) \in L\}$
12	**return** $y \xleftarrow{\$} Y$	22	**return** $x \xleftarrow{\$} X$
13	$y \xleftarrow{\$} S$	23	$x \xleftarrow{\$} S$
14	**if** $\exists x'$ such that $(x', y) \in L$ **then**	24	**if** $\exists y'$ such that $(x, y') \in L$ **then**
15	$\text{BAD} \leftarrow \mathsf{true}$	25	$\text{BAD} \leftarrow \mathsf{true}$
[16]	$y \xleftarrow{\$} \overline{\text{Range}}(L)$	[25]	$x \xleftarrow{\$} \overline{\text{Domain}}(L)$
17	$L \leftarrow L \cup \{(x, y)\}$	26	$L \leftarrow L \cup \{(x, y)\}$
18	**return** y	27	**return** x

Fig. 3: Games utilized in proof of Lemma 7. The list L is initially empty. Game G0 includes the instructions with boxed-in line numbers, while game G1 does not. The set $\overline{\text{Range}}(L)$ is the set of all y not yet assigned to some pair (x, y) in the list L. Similarly, the set $\overline{\text{Domain}}(L)$ is the set of all x not yet assigned to some pair (x, y) in the list.

We are now ready to show that replacing the random one-way functions f_1 and f_2 with respective random two-way permuations π_1 and π_2 (with respective inverses π_1^{-1} and π_2^{-1}) in Davies-Meyer mode (that is $f_1(M) = \pi_1(M) \oplus M$ and $f_2(V) = \pi_2(V) \oplus V$) hardly affects our proof of security and that, as before, we can replace the queries to π_1, π_2 and their inverses by two random lists $\mathbf{a}$ and $\mathbf{b}$. For this we replace the pseudopermutation D, E of the previous Lemma by a slightly different construction D', E'. Instead of generating the output of D' and E' on-the-fly for fresh queries, we now commit to a list $\mathbf{a}$ of elements in $S = \{0,1\}^n$ in advance and on the i'th fresh query (be it y_i to D' or x_i to E') we return the queried value plus the i'th element in the list (so $D'(y_i) = y_i \oplus a_i$, resp. $E'(x_i) = x_i \oplus a_i$). It is not hard to see that D', E' has the exact same output distribution as D, E (provided the list $\mathbf{a}$ is long enough), so an adversary cannot distinguish between the two. Furthermore, the effect of using D', E' in the place of π, π^{-1} in the construction $f(x) = \pi(x) \oplus x$ is that, regardless of which queries the adversary makes, we get $f(x_i) = a_i$.

Another (less exciting) consequence of the PPP Switching Lemma is that finding collisions in the rate-1/2 compression function $f_1(M) \oplus f_2(V)$ requires $\Theta(2^{n/4})$ queries in the ideal cipher model when $f_1(M) = E_{K_1}(M) \oplus M$ and $f_2(V) = E_{K_2}(V) \oplus V$ (with $K_1 \neq K_2$).

D Poisson Heuristic to Approximate the Yield

In this appendix, we will derive an alternative characterization of $\mathrm{yield}(\mathbf{a} \otimes \mathbf{b})$. We then combine it with the conjectured distribution of k-way collisions in $A \oplus B$ in order to recast the problem of finding the expected value $\mathrm{yield}_n^{A \oplus B}(q)$ into that of determining a certain property of the tail of a Poisson distribution. The latter problem can be tackled much more easily numerically for larger values of n. Indeed, we provide experimental results to both validate our reformulation, as well as determining concrete estimates of the collision resistance of our proposal in practice. It turns out that for n up to 256, the loss in collision resistance is at most four bits.

THEORETICAL BACKGROUND. Let $\mathbf{c} \in (\{0,1\}^n)^Q$ be given (we can ignore for the moment from which distribution $\mathbf{c}$ has arisen). Recall that $\mathrm{yield}(\mathbf{c})$ is the sum of the frequencies of the most frequent elements in $\mathbf{c}$. Thus, to determine $\mathrm{yield}(\mathbf{c})$, it suffices to know how many k-way collisions there are, for all k. Recalling that $M_{\mathbf{c}}(k)$ denotes the number of k-way collisions in $\mathbf{c}$, then we have:

$$\mathrm{yield}(\mathbf{c}) = \max_{\substack{\mathbf{w} \in [0,\ldots,Q]^{Q+1} \\ w_k \leq M_{\mathbf{c}}(k), \sum_{k=0}^{Q} w_k = q}} \sum_{k=0}^{Q} k w_k \ .$$

To approximate $\mathrm{yield}_n^{A \oplus B}(q)$, we can look what happens if we look at $\mathrm{yield}(\mathbf{c})$ for an (imaginary) sample $\mathbf{c}$ whose number of k-way collisions exactly equals the expectation of $M_{A \oplus B}(k)$ (for all k). That such an ideal $\mathbf{c}$ might not exist, since the expected value of $M_{A \oplus B}(k)$ could be fractional, is irrelevant. We refer Lemma 3 for the expectation of $M_{A \oplus B}(k)$. (If $\mathbf{c}$ were drawn uniformly at random, rather than from $A \oplus B$, we could use Theorem 2, leading to the same approximation.) Let $\mathrm{yield}_n^P(q)$ be the yield based on the Poisson heuristic corresponding to a sample size of $Q = q^2$ elements of $\{0,1\}^n$. Then we pose the following approximation, where $\lambda = Q/N$:

$$\mathrm{yield}_n^{A \oplus B}(q) \approx \mathrm{yield}_n^P(q) = \max_{\substack{\mathbf{w} \in [0,\ldots,Q]^{Q+1} \\ w_k \leq \frac{N \lambda^k}{k! e^\lambda}, \sum_{k=0}^{Q} w_k = q}} \sum_{k=0}^{Q} k w_k \ .$$

Note that we make two types or error in this approximation. The distribution of $\mathbb{E}(M_{A\oplus B}(k))$ is not exactly a scaled Poisson distribution; moreover, the maximum of an expected vector is not the same as the expected maximum of a vector. It is however a lower bound.

EXPERIMENTAL JUSTIFICATION We provide experimental results to support our claim on the behaviour of expected number of k-way collisions for the distribution $A\oplus B$. It also suggests that $\mathrm{yield}_n^P(q)$ is a very good estimator for the actual $\mathrm{yield}_n^{A\oplus B}(q)$.

In order to get accurate estimates, we performed the experiment of picking from A and B a large number of times per pair (n, q). (Anywhere from 10^3 to 10^6 runs per pair.) Given that each run takes time 2^n with similar space requirements, we only have the data for n up to 22.

In Table 1 the data is given for query complexity $q = 2^{n/2}$. Since $N = 2^n$ and $Q = q^2$ this means that $\lambda = 1$. The first column gives the values of n, from which the other parameters (q, Q, and N) can be deduced. We then give three columns with the average yield (or rather the logarithms thereof). The first is the result of experiments based on the distribution $A\oplus B$, the second of experiments based on a uniform distribution U and the final is the Poisson estimation. The remaining five columns give the average numbers $M_{A\oplus B}(k)$, normalized by N. These are the numbers we conjecture are distributed according to a Poisson distribution with parameter Q/N. So, in the final row we give the Poisson distribution with $\lambda = 1$ for reference.

Table 2 gives the analogous results, but for $q = 2^{n/2-1}$, corresponding to $\lambda = \frac{1}{4}$.

ESTIMATED COLLISION-RESISTANCE. We are now ready to estimate the actual level of collision-resistance our construction offers. For each value of n, we have determined the smallest q for which the resulting average yield exceeds $2^{n/2}$. (Note that this is not exactly equivalent to having probability half of finding collisions, but we are confident it gives a faithful indication.)

In Table 3a we have tabulated $\log_2 q$ for small values of n. Included are three versions. First we give the value of q that, when simulating the experiment of picking A and B and computing $\mathrm{yield}_{A\oplus B}$, gives an average yield exceeding $2^{n/2}$. Second up is the corresponding result for the experiment of picking U directly uniformly at random. Finally we also give the value of $\log_2 q$ that follows from the Poisson estimation. For the latter we also provide the value of c such that $q = 2^{n/2}/n^{-c}$, that is $c = \log_n \frac{2^{n/2}}{q}$.

In Table 3b we have only given the values corresponding to the Poisson estimation, for n of cryptographic relevance. Note the very small loss of actual security. Even for a 512-bit primitive the loss is less than five bits. The table also shows the very slow increase in c.

As a consequence of $\log_2 q$ being fairly close to $\frac{n}{2}$, one might want to take the probability of finding a collision in either f_1 or f_2 into account as well. Assuming f_1 and f_2 are random functions, the probability of finding a collision in either is about 2^{-6} for $n = 160$, decreasing even further to 2^{-7} for $n = 256$, where q is chosen according to Table 3. (Again, if f_1 and f_2 are random permutations, this issue is moot.)

n	$\log_2$ yield			Average nr. of k-way collisions / N				
	$A \oplus B$	U	Poisson	$M_{A\oplus B}(1)$	$M_{A\oplus B}(2)$	$M_{A\oplus B}(3)$	$M_{A\oplus B}(4)$	$M_{A\oplus B}(5)$
4	3.21	3.20	3.15	0.366	0.206	0.054	0.012	0.0016
6	4.47	4.49	4.45	0.367	0.189	0.060	0.014	0.0027
8	5.74	5.75	5.74	0.368	0.185	0.061	0.015	0.0030
10	6.90	6.90	6.90	0.368	0.184	0.061	0.015	0.0030
12	8.09	8.10	8.10	0.368	0.184	0.061	0.015	0.0031
14	9.19	9.19	9.19	0.368	0.184	0.061	0.015	0.0031
16	10.35096	10.35301	10.35384	0.3678	0.1839	0.0613	0.0153	0.00307
18	11.42019	11.42016	11.42020	0.3679	0.1839	0.0613	0.0153	0.00307
20	12.51232	12.51221	12.51233	0.3679	0.1839	0.0613	0.0153	0.00306
22	13.63089	13.63084	13.63085	0.3679	0.1839	0.0613	0.0153	0.00307
Poisson, $\lambda = 1$				0.3679	0.1839	0.0613	0.0153	0.00307

Table 1: Comparison of experimental yield and its Poisson estimate for $q = 2^{\frac{n}{2}}$ samples.

n	$\log_2$ yield			Average nr. of k-way collisions / N				
	$A \oplus B$	U	Poisson	$M_{A\oplus B}(1)$	$M_{A\oplus B}(2)$	$M_{A\oplus B}(3)$	$M_{A\oplus B}(4)$	$M_{A\oplus B}(5)$
6	2.52	2.53	2.54	0.787	0.0968	0.00543	0.000636	0.0000313
8	3.85	3.90	3.93	0.781	0.0975	0.00736	0.000517	0.0000302
10	5.10	5.10	5.10	0.780	0.0972	0.00791	0.000501	0.0000283
12	6.20	6.20	6.20	0.779	0.0974	0.00805	0.000505	0.0000261
14	7.37	7.37	7.37	0.779	0.0974	0.00810	0.000506	0.0000259
16	8.613	8.617	8.619	0.779	0.0974	0.00810	0.000507	0.0000254
18	9.653	9.652	9.652	0.779	0.0973	0.00811	0.000507	0.0000254
Poisson, $\lambda = \frac{1}{4}$				0.779	0.0974	0.00811	0.000507	0.0000254

Table 2: Comparison of experimental yield and its Poisson estimate for $q = 2^{\frac{n}{2}-1}$ samples.

n	q $(\log_2 q)$			c
	$A \oplus B$	U	Poisson	
8	9 (3.17)	12 (3.58)	9 (3.16993)	0.28
10	16 (4.00)	17 (4.09)	16 (4.00000)	0.30
12	30 (4.91)	31 (4.95)	30 (4.90689)	0.30
14	56 (5.81)	60 (5.91)	56 (5.80735)	0.31
16	105 (6.71)	114 (6.83)	105 (6.71425)	0.32
18	195 (7.61)	210 (7.71)	195 (7.60733)	0.33
20	360 (8.49)	363 (8.50)	360 (8.49185)	0.35
22	676 (9.40)	676 (9.40)	676 (9.40088)	0.36
24	1338 (10.39)	1343 (10.39)	1338 (10.3859)	0.35

(a) Small n

n	$\log_2 q$	c
32	14.2148	0.36
64	29.6518	0.39
96	45.2324	0.42
128	60.9975	0.43
160	76.8034	0.44
192	92.5954	0.45
224	108.415	0.46
256	124.3	0.46
384	187.907	0.48
512	251.601	0.49

(b) Large n

Table 3: The relative bit-security provided against collision resistance based on n-bit primitives.