Collision Attacks against the Knudsen-Preneel Compression Functions[*]

Onur Özen[†] and Martijn Stam

LACAL, EPFL
Station 14, CH-1015 Lausanne, Switzerland
{onur.ozen, martijn.stam}@epfl.ch

Abstract. Knudsen and Preneel (Asiacrypt'96 and Crypto'97) introduced a hash function design in which a linear error-correcting code is used to build a wide-pipe compression function from underlying blockciphers operating in Davies-Meyer mode. Their main design goal was to deliver compression functions with collision resistance up to, and even beyond, the block size of the underlying blockciphers. In this paper, we present new collision-finding attacks against these compression functions using the ideas of an unpublished work of Watanabe and the preimage attack of Özen, Shrimpton, and Stam (FSE'10). In brief, our best attack has a time complexity strictly smaller than the block-size for all but two of the parameter sets. Consequently, the time complexity lower bound proven by Knudsen and Preneel is incorrect and the compression functions do not achieve the security level they were designed for.

Keywords: Collision attack, coding theory, compression function.

1 Introduction

Hash functions are currently at the centre of the cryptographic community's attention. While most of this attention is geared directly towards the SHA-3 competition (by analysing its remaining candidates), other, arguably more fundamental questions regarding hash function design should not be forgotten. After all, the study of the underlying principles of hash function design are potentially beneficial for the SHA-3 decision process.

The two most revered principles in hash function design are (i) the Merkle-Damgård iterative construction, or more generally the principle of designing a secure *compression* function and (ii) the Davies-Meyer construction, or more generally the principle of using a *blockcipher* as underlying primitive. Indeed, the currently standardized hash functions from the SHA family follow this approach (as did their predecessor MD5) as well as several of the SHA-3 candidates.

[*] This work has been supported in part by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II.

[†] Supported by a grant of the Swiss National Science Foundation, 200021-122162.

It was already recognized early on that the output sizes of traditional blockciphers are insufficient to yield a secure compression function [16]. This still holds true today: for all the (optimally secure) PGV blockcipher-based compression functions [10,1,12] based on an n-bit blockcipher, the (time) complexity of collision- and preimage-finding attacks is at most $2^{n/2}$, resp. 2^n; when $n = 128$ (e.g. AES) the resulting bounds are mostly unacceptable for current practice (in particular for collision resistance).

To achieve acceptable security (based on small block sizes) it is necessary to output a multiple of the block-length. In the 1990s many constructions were proposed for this goal, mostly outputting $2n$ bits with the explicit collision resistance target of 2^n (see [3,9] for an overview). The standard goal for these constructions has been optimal collision-resistance: a target output size is fixed and the compression function should be collision resistant up to the birthday bound for that digest size. In three papers [4,5,6], Knudsen and Preneel adopted a different approach, namely to fix a particular security target and let the output size (and relatedly the number of blockcipher calls) vary as needed in order to guarantee a particular security target *without* imposing optimal security.

Specifically, given r independent ideal compression functions $f_1, \ldots, f_r$, each mapping cn bits to n bits, they create a new 'bigger' compression function outputting rn bits. Following principles (i) and (ii) already mentioned, they then propose to instantiate the underlying ideal compression functions with a blockcipher run in Davies-Meyer mode and to iterate the compression function to obtain a full blockcipher-based hash function. However, they derive their security from the compression function, so that is where we will focus our attention.

The $f_1, \ldots, f_r$ are run in parallel where each of their inputs is some linear combination of the blocks of message and chaining variable that are to be processed; the rn-bit output of their construction is the concatenation of the outputs of these parallel calls. The elegance of the KP construction is in *how* the inputs to $f_1, \ldots, f_r$ are computed. They use the generator matrix of an $[r, k, d]$ error-correcting code over $\mathbb{F}_{2^c}$ to determine how the ck input blocks of the 'big' compression function are xor'ed together to form the inputs to the underlying r functions. (In a generalization they consider the f_i as mapping from bcn' to bn' bits instead and use a code over $\mathbb{F}_{2^{bc}}$.)

The (deliberate) effect of this design is that when two inputs to the 'big' compression function differ, the corresponding inputs for the underlying functions will differ for at least d functions. In particular, when using a systematic generator, a change in the systematic part of the input results in at least $d-1$ so-called *active* functions in the non-systematic part. Intuitively this means that one has to find a preimage, resp. a collision for the $d-1$ active functions in parallel. Based on this observation, Knudsen and Preneel claim that (under an assumption) any collision attack needs time at least $2^{(d-1)n/2}$ (and as many f_i evaluations) and they conjecture that a preimage attack will require time at least $2^{(d-1)n}$. Additionally, they give preimage and collision attacks (sometimes matching their lower bounds).

Table 1. A summary of collision attacks on the Knudsen-Preneel compression functions, with constant and polynomial factors (in n) ignored. Non-MDS parameters are in italic, for $e \in \{2, 4\}$ the underlying primitive is $f_i : \{0,1\}^{2n} \to \{0,1\}^n$, and for $e = 3$ it is $f_i : \{0,1\}^{3n} \to \{0,1\}^n$.

Code	Algorithm 4, [8] Complexity		Algorithm 2 Complexity	ÖSS Complexity		Knudsen-Preneel	
	Query	Time	Time/Query	Query	Time	Lower Bound	Attack Time
$[r,k,d]_{2^e}$	$2^{kn/(3k-r)}$	Thm. 4, [8]	$2^{dn/(d+1)}$	$2^{rn/(2k)}$	[8]	$2^{(d-1)n/2}$	[6]
$[5,3,3]_4$	$2^{3n/4}$	$2^{3n/4}$	$2^{3n/4}$	$2^{5n/6}$	$2^{4n/3}$	2^n	$2^{4n/3}$
$[\mathit{8},\mathit{5},\mathit{3}]_{\mathit{4}}$	$2^{5n/7}$	$2^{5n/7}$	$2^{3n/4}$	$2^{4n/5}$	$2^{7n/5}$	2^n	$2^{3n/2}$
$[\mathit{12},\mathit{9},\mathit{3}]_{\mathit{4}}$	$2^{3n/5}$	$2^{3n/5}$	$2^{3n/4}$	$2^{2n/3}$	$2^{4n/3}$	2^n	$2^{3n/2}$
$[\mathit{9},\mathit{5},\mathit{4}]_{\mathit{4}}$	$2^{5n/6}$	$2^{5n/6}$	$2^{4n/5}$	$2^{9n/10}$	$2^{11n/5}$	$2^{3n/2}$	2^{2n}
$[\mathit{16},\mathit{12},\mathit{4}]_{\mathit{4}}$	$2^{3n/5}$	$2^{4n/5}$	$2^{4n/5}$	$2^{2n/3}$	$2^{7n/3}$	$2^{3n/2}$	2^{2n}
$[6,4,3]_{16}$	$2^{2n/3}$	$2^{2n/3}$	$2^{3n/4}$	$2^{3n/4}$	$2^{3n/2}$	2^n	$2^{5n/4}$
$[8,6,3]_{16}$	$2^{3n/5}$	$2^{3n/5}$	$2^{3n/4}$	$2^{2n/3}$	$2^{4n/3}$	2^n	$2^{7n/6}$
$[12,10,3]_{16}$	$2^{5n/9}$	$2^{5n/9}$	$2^{3n/4}$	$2^{3n/5}$	$2^{6n/5}$	2^n	$2^{11n/10}$
$[9,6,4]_{16}$	$2^{2n/3}$	$2^{2n/3}$	$2^{4n/5}$	$2^{3n/4}$	2^{2n}	$2^{3n/2}$	$2^{3n/2}$
$[16,13,4]_{16}$	$2^{13n/23}$	$2^{20n/23}$	$2^{4n/5}$	$2^{8n/13}$	2^{2n}	$2^{3n/2}$	$2^{3n/2}$
$[4,2,3]_8$	2^n	2^n	$\times$	2^n	2^{2n}	2^n	$2^{3n/2}$
$[6,4,3]_8$	$2^{2n/3}$	$2^{2n/3}$	$2^{3n/4}$	$2^{3n/4}$	$2^{3n/2}$	2^n	$2^{5n/4}$
$[9,7,3]_8$	$2^{7n/12}$	$2^{7n/12}$	$2^{3n/4}$	$2^{9n/14}$	$2^{8n/7}$	2^n	$2^{8n/7}$
$[5,2,4]_8$	$\times$	$\times$	$\times$	$2^{5n/4}$	2^{3n}	$2^{3n/2}$	$2^{7n/4}$
$[7,4,4]_8$	$2^{4n/5}$	$2^{4n/5}$	$2^{4n/5}$	$2^{7n/8}$	$2^{9n/4}$	$2^{3n/2}$	$2^{3n/2}$
$[10,7,4]_8$	$2^{7n/11}$	$2^{9n/11}$	$2^{4n/5}$	$2^{5n/7}$	2^{2n}	$2^{3n/2}$	$2^{3n/2}$

Watanabe [14] already pointed out a collision attack beating the one given by Knudsen and Preneel for many of the parameter sets. In particular, his differential attack works whenever $r < 2k$ and has a query and time complexity of essentially $k2^n$. Thus he demonstrated that the *proven* collision resistance *lower* bound given by Knudsen and Preneel is incorrect whenever $r < 2k$ and $d > 3$. For a code with minimum distance $d = 3$ he matches the Knudsen-Preneel 2^n collision-resistance lower bound, but does not violate it; the two codes proposed by Knudsen and Preneel with $r \geq 2k$ (namely $[4,2,3]_8$ and $[5,2,4]_8$) seem beyond reproach. Yet this was the first indication that something is amiss with the claim by Knudsen and Preneel. A second indication arrived at FSE'10, when Özen, Shrimpton, and Stam [7] demonstrated a remarkably efficient preimage attack that, for 9 out of 16 cases, runs in time $2^{rn/k}$ which was shown optimal. Moreover, using a yield-based argument, they showed that an information-theoretic adversary in principle should be able to find collisions after only $2^{rn/(2k)}$ queries.

Our contribution. In this paper we deal what we believe will be the final blow against the Knudsen-Preneel compression functions. Our contribution is fourfold, with a summary provided in Table 1. For completeness, we have also investigated the time complexity that one would obtain by straightforward adaptation of the ideas and query complexities of ÖSS; we refer to the full version of this paper [8] for the details.

The *mise en place* in Section 4 provides a detailed mathematical characterization of the Knudsen-Preneel compression function's preprocessing. As a first simple result, this allows us in Section 5.1 to revise the attack of Watanabe in a way that slightly reduces the time requirements, yet significantly increases the number of collisions it can produce. More precisely, after an initial effort of $d2^n$, we can generate (up to) $2^{(k-d)n}$ collisions in constant time (for $k > d$).

In our revised version of Watanabe's attack, we fix a pure tensor to create a differential. By adaptively looking for an arbitrary tensor and using the same type of queries as Özen, Shrimpton, and Stam we arrive in Section 5.2 at a new, symbiotic collision-finding attack with time complexity $d2^{dn/(d+1)}$. The attack works whenever $d \leq k$ (as in Watanabe's case). Even more amazing is that if the inequality is strict, that is if $d < k$, the adversary can create further collisions (like our revised attack) in *constant* time (up to $2^{(k-d)n}$ collisions).

Thirdly, in Section 6.1 we introduce a parametrized information-theoretic collision attack. It turns out that the new symbiotic attack and the old ÖSS information-theoretic collision attack are both on opposite ends of the spectrum of this parametrized attack, yet optimality is typically achieved somewhere in the middle—with $\mathsf{KP}([4,2,3]_8)$ and $\mathsf{KP}([5,2,4]_8)$ again as exceptions—yielding query complexity $2^{kn/(3k-r)}$.

Our final contribution is a reduced-time variant of our optimized attack above. For this we use the same ideas as ÖSS, but with a crucial twist: where they used the dual code to look for preimages efficiently, we will use the dual *shortened* code to search for collisions efficiently. As a result, for 12 out of 16 suggested parameters we can mount a collision attack whose time complexity matches its query complexity (ignoring constants and logarithmic factors). Even better, only for $\mathsf{KP}([5,2,4]_8)$ we are unable to beat the time-complexity of any prior attack we are aware of, for the rest we set new records.

2 Preliminaries

With some minor modifications, we will adhere to the notation also used by Özen, Shrimpton, and Stam.

Linear error correcting codes. An $[r,k,d]_{2^e}$ linear error correcting code $\mathcal{C}$ is the set of elements (codewords) in a k-dimensional subspace of $\mathbb{F}_{2^e}^r$ (for $r \geq k$), where the minimum distance d is defined as the minimum Hamming weight (taken over all nonzero codewords in $\mathcal{C}$). The dual code $[r, r-k, d^\perp]_{2^e}$ is the set of all elements in the $r-k$-dimensional subspace orthogonal to $\mathcal{C}$ (with respect to the usual inner product), and its minimum distance is denoted $d^\perp$. The Singleton bound puts a limit on the minimum distance: $d \leq r-k+1$. Codes matching the Singleton bound are called maximum distance separable (MDS). An important property of an MDS code is that its duals is MDS as well, so $d^\perp = k+1$.

An $[r,k,d]_{2^e}$ code $\mathcal{C}$ can be generated by a matrix $G \in \mathbb{F}_{2^e}^{k\times r}$, meaning that $\mathcal{C} = \{x \cdot G | x \in \mathbb{F}_{2^e}^{k}\}$ (using row vectors throughout). A generator matrix G is called systematic iff it has the form $G = [I_k|P]$ for $P \in \mathbb{F}_{2^e}^{k\times(r-k)}$ and I_k the identity matrix in $\mathbb{F}_{2^e}^{k\times k}$. Furthermore, G is the generator matrix of an MDS code iff any k columns are linearly independent. For an index set $\mathcal{I} \subseteq \{1,\ldots,r\}$ we define $G_{\mathcal{I}} \in \mathbb{F}_{2^e}^{k\times|\mathcal{I}|}$ as the restriction of G to those columns indexed by $\mathcal{I}$. For a code and any index set $\mathcal{I} \subseteq \{1,\ldots,r\}$, we want to define $\tilde{\mathcal{I}} \subset \{1,\ldots,r\}$ such that $G_{\tilde{\mathcal{I}}}$ is invertible (thus in particular $|\tilde{\mathcal{I}}| = k$) and $\tilde{\mathcal{I}} \subseteq \mathcal{I}$ or $\mathcal{I} \subseteq \tilde{\mathcal{I}}$. For MDS codes, the existence of such an $\tilde{\mathcal{I}}$ can be shown easily (and we can impose uniqueness e.g. by virtue of an ordering). For non-MDS codes there exist some $\mathcal{I}$ for which such an $\tilde{\mathcal{I}}$ does not exist (for example the $\mathcal{I} \subset \{1,\ldots,r\}$ for which $|\mathcal{I}| = k$ but $G_{\mathcal{I}}$ is not invertible), however for any target cardinality it is possible to find an $\mathcal{I}$ (of that cardinality) that does have an $\tilde{\mathcal{I}}$ (e.g. by first going through the systematic columns); we call such an $\mathcal{I}$ *admissible*.

A given $[r,k,d]_{2^e}$ code $\mathcal{C}$ can be *shortened* to obtain a new, derived code $\mathcal{C}'$. Let $i \in \{1,\ldots,r\}$, then consider the set of all codewords in $\mathcal{C}$ that are 0 on position i. The new code $\mathcal{C}'$ consists of these codewords with position i dropped, however we sometimes 'quasi-shorten' and keep the superfluous zeroes present (we always keep the original indexing). It is easy to see that $\mathcal{C}'$ is an $[r-1,k-1,d]_{2^e}$ code unless all codewords in $\mathcal{C}$ had a 0 on position i or $k=1$ (in the latter case the shortening might result in the trivial one-codeword code $\{0^{r-1}\}$). The shortening of an MDS code is an MDS code itself. By repeated application one can shorten by any index set $\mathcal{I}_0 \subset \{1,\ldots,r\}$ for which $\theta = |\mathcal{I}_0| < k$ to obtain a derived $[r-\theta, k-\theta, d]$ MDS code $\mathcal{C}'$. If G is systematic and $\mathcal{I}_0 = \{1,\ldots,\theta\}$ we can generate the shortened code by dropping the first θ rows of $G_{\mathcal{I}}$, where $\mathcal{I} = \{1,\ldots,r\}\backslash\mathcal{I}_0 = \{\theta+1,\ldots,r\}$. (For the four non-MDS codes used by Knudsen and Preneel we will perform a separate analysis on repeated shortening.)

Blockwise-linear compression functions. A *compression function* is a mapping $H : \{0,1\}^{tn} \to \{0,1\}^{sn}$ for some blocksize $n > 0$ and integer parameters $t > s > 0$. For positive integers c and n, we let $\mathrm{Func}(cn,n)$ denote the set of all functions mapping $\{0,1\}^{cn}$ into $\{0,1\}^{n}$. A compression function is *Public Random Function (PuRF)-based* if its mapping is computed by a program with oracle access to a finite number of specified oracles $f_1,\ldots,f_r$, where $f_1,\ldots,f_r \xleftarrow{\$} \mathrm{Func}(cn,n)$. When a PuRF-based compression function operates on input W, we write $H^{f_1,\ldots,f_r}(W)$ for the resulting value. Of primary interest for us will be *single-layer* PuRF-based compression functions without feedforward. These call all oracles in parallel and compute the output based only on the results of these calls; in particular, input to the compression function is not considered.

Most PuRF-based (and blockcipher-based) compression functions are of a special type. Instead of arbitrary pre- and postprocessing, one finds only functions that are blockwise linear. The Knudsen-Preneel construction is also blockwise linear, so let us recall from [7] what is a *blockwise-linear scheme.*

Definition 1 (Blockwise-linear scheme). *Let r, c, b, t, s be positive integers and let matrices $\mathsf{C}^{\text{PRE}} \in \mathbb{F}_2^{rcb\times tb}$, $\mathsf{C}^{\text{POST}} \in \mathbb{F}_2^{sb\times rb}$ be given. We define $H = \mathsf{BL}^b(\mathsf{C}^{\text{PRE}}, \mathsf{C}^{\text{POST}})$ to be a family of single-layer PuRF-based compression functions $H_n : \{0,1\}^{tn} \to \{0,1\}^{sn}$, for all positive integers n with $b|n$. Specifically, let $n'b = n$, and $f_1, \ldots, f_r \in \text{Func}(cn, n)$. Then on input $W \in \{0,1\}^{tn}$ (interpreted as column vector), ${H_n}^{f_1 \cdots f_r}(W)$ computes the digest $Z \in \{0,1\}^{sn}$ as follows:*

1. *Compute $X \leftarrow (\mathsf{C}^{\text{PRE}} \otimes I_{n'}) \cdot W$;*
2. *Parse $X = (x_i)_{i=1\ldots r}$ and for $i = 1 \ldots r$ compute $y_i = f_i(x_i)$;*
3. *Parse $(y_i)_{i=1\ldots r} = Y$ and output $Z = (\mathsf{C}^{\text{POST}} \otimes I_{n'}) \cdot Y$.*

where $\otimes$ denotes the Kronecker product and $I_{n'}$ the identity matrix in $\mathbb{F}_2^{n'\times n'}$.

In the definition above we silently identified $\{0,1\}^n$ with the vector space $\mathbb{F}_2^n$. The map corresponding to $(\mathsf{C}^{\text{PRE}} \otimes I_{n'})$ will occasionally be denoted C^{PRE} and its image $\Im(C^{\text{PRE}}) \subseteq \{0,1\}^{rcn}$. It will be convenient for us to write the codomain of C^{PRE} as a direct sum, so we identify $\{0,1\}^{rcn}$ with $\bigoplus_{i=1}^{r} V_i$ where $V_i = \mathbb{F}_2^{cn}$ for $i= 1, \ldots, r$. If $x_1 \in V_1$ and $x_2 \in V_2$, then consequently $x_1 + x_2$ will be in $V_1 \oplus V_2$. (This extends naturally to $L_1 + L_2$ when $L_1 \subset V_1, L_2 \subset V_2$.)

Knudsen-Preneel compression functions. Knudsen and Preneel [4,5] introduced a family of hash functions employing error correcting codes. (We use the journal version [6] as our frame of reference). Although their work was ostensibly targeted at blockcipher-based designs, the main technical thread of their work develops a transform that extends the range of an 'ideal' compression function (blockcipher-based, or not) in a manner that delivers some target level of security. As is nowadays typical, we understand an ideal compression function to be a PuRF. In fact, the KP transform is a special instance of a blockwise-linear scheme (Definition 1), in which the inputs to the PuRFs are determined by a linear code over a binary field with extension degree $e > 1$, i.e. $\mathbb{F}_{2^e}$, and with C^{POST} being the identity matrix over $\mathbb{F}_2^{rb\times rb}$ (corresponding to concatenating the PuRF outputs). The extension field itself is represented as a subring of the matrix ring (of dimension equalling the extension degree) over the base field. We formalize this by an injective ring homomorphism $\varphi : \mathbb{F}_{2^e} \to \mathbb{F}_2^{e\times e}$ and let $\bar{\varphi} : \mathbb{F}_{2^e}^{r\times k} \to \mathbb{F}_2^{re\times ke}$ be the component-wise application of φ and subsequent identification of $(\mathbb{F}_2^{e\times e})^{r\times k}$ with $\mathbb{F}_2^{re\times ke}$ (we will use $\bar{\varphi}$ for matrices over $\mathbb{F}_{2^e}$ of arbitrary dimensions). For completeness, there is also a group homomorphism $\psi : \mathbb{F}_{2^e} \to \mathbb{F}_2^e$ such that for all $g, h \in \mathbb{F}_{2^e}$ it holds that $\psi(gh) = \varphi(g) \cdot \psi(h)$.

Definition 2 (Knudsen-Preneel transform). *Let $[r, k, d]$ be a linear code over $\mathbb{F}_{2^e}$ with generator matrix $G \in \mathbb{F}_{2^e}^{k\times r}$. Let $\varphi : \mathbb{F}_{2^e} \to \mathbb{F}_2^{e\times e}$ be an injective ring homomorphism and let b be a positive divisor of e such that $ek > rb$. Then the Knudsen-Preneel compression function $H = \mathsf{KP}^b([r, k, d]_{2^e})$ equals $H = \mathsf{BL}^b(\mathsf{C}^{\text{PRE}}, \mathsf{C}^{\text{POST}})$ with $\mathsf{C}^{\text{PRE}} = \bar{\varphi}(G^T)$ and $\mathsf{C}^{\text{POST}} = I_{rb}$.*

If $H = \mathsf{KP}^b([r, k, d]_{2^e})$, then $H_n : \{0,1\}^{kcn} \to \{0,1\}^{rn}$ with $c = e/b$ is defined for all n for which b divides n. Moreover, H_n is based on r PuRFs in $\text{Func}(cn, n)$.

For use of H in an iterated hash function, note that per invocation (of H) one can compress $(ck - r)$ message blocks (hence the requirement $ek > rb$ ensures actually compression is taking place), and the rate of the compression function is $ck/r - 1$. We will concentrate on the case $(b, e) \in \{(1, 2), (2, 4), (1, 3)\}$ and then in particular on the 16 parameter sets given by Knudsen and Preneel.[1] Since b is uniquely determined given e (and c), we will often omit it.

Security notions. A *collision-finding adversary* is an algorithm whose goal is to find two distinct inputs W, W' that hash to the same value, so $H(W) = H(W')$. We will consider adversaries in two scenarios: the information-theoretic one and a more realistic concrete setting. For information-theoretic adversaries the only resource of interest is the number of queries made to their oracles. Otherwise, these adversaries are considered (computationally) unbounded. In the concrete setting, on the other hand, we are interested in the actual runtime of the algorithm and, to a lesser extent, its memory consumption (and code-size).

3 Prior Art on the Knudsen-Preneel Hash Functions

Knudsen and Preneel's security claims. Knudsen and Preneel concentrate on the collision resistance of their compression function in the complexity theoretic model. Under a fairly generous (but plausible) assumption, they essentially show that if $H = \mathsf{KP}^b([r, k, d]_{2^e})$, then finding collisions in H_n takes time at least $2^{(d-1)n/2}$. For preimage resistance Knudsen and Preneel do not give a corresponding theorem and assumption, yet they do *conjecture* it to be essentially the square of the collision resistance.

Knudsen and Preneel also present two attacks, one for finding preimages [6, Proposition 3] and one for finding collisions [6, Proposition 4] (see results in Table 1). Both attacks revolve around finding multi-preimages for the systematic part of the construction in sufficient numbers to make it likely that completion to the non-systematic part will yield a full preimage respectively a full collision.

Watanabe's collision-finding attack. Knudsen and Preneel left a considerable gap between the actual complexity of attacks and their lower bounds in the case of collision resistance. Watanabe [14] has pointed out a collision attack that runs in time $k2^n$ (and as many PuRF evaluations). Thus, for many of the parameter sets, it beats the one given by Knudsen and Preneel. More interestingly, his attack serves as proof that the *lower* bound given by Knudsen and Preneel is *incorrect* for a large class of parameters: whenever $r < 2k$ and $d > 3$, which involves 6 out of 16 parameter sets. (See also Table 1.)

Assume that the code's generator matrix is systematic, that is $G = (I_k|P)$ with $P \in \mathbb{F}_{2^e}^{k\times(r-k)}$. Then the goal is to generate, for each $i \in \{1, \ldots, k\}$, a colliding pair of inputs $x_i \neq x'_i$ (and $f_i(x_i) = f_i(x'_i)$) in such a way that their completion to full 'codewords' satisfies $x_i = x'_i$ for $i \in \{k + 1, \ldots, r\}$. This is done by ensuring that $x_i \oplus x'_i = \Delta_i$ where $\Delta = \sum_{i=1}^{k} \Delta_i \in \mathbb{F}_2^{ken'} \backslash \{0\}$ is in the

[1] We note that our analysis is also valid for $c = 5$ (mimicking the MD4/5 situation).

kernel of $\bar{\varphi}(P^T) \otimes I_{n'}$ (since $r - k < k$ the kernel is guaranteed to contain a nontrivial element). Mutual independence of the inputs to the PuRFs corresponding to the code's systematic part allow the initial collision searches to be mounted independently. Unfortunately, since the collisions need to be rather special (due to fixed Δ_i's), the birthday paradox does not apply and a collision search costs about 2^n queries and time (per PuRF). On the plus side, the attack is trivially memoryless and parallellizable.

Özen-Shrimpton-Stam preimage-finding attack. An extensive security analysis for the preimage resistance of KP-constructions, falsifying the designers' conjectured lower bound, has been provided by Özen, Shrimpton, and Stam [7]. Additionally, they also provided a related collision-finding attack with a surprisingly low query complexity: $2^{rn/(2k)}$ (but no analysis of its time complexity).

At the core of the Özen-Shrimpton-Stam attacks is the simple observation that $(0^a \,||\, x_1) \oplus (0^a \,||\, x_2)$ yields a string of the form $(0^a \,||\, X)$. More generally, any linear combination of strings with the same pattern of fixed zero bits will yield a string with the same form. By restricting PuRF queries to strings with the same (blockwise) pattern one can optimize the *yield* of these queries (i.e. the maximum number of KP compression function evaluations an adversary can compute for a given number of queries). Matching the yield with the size of the codomain (resp. its square root) gives rise to an information-theoric preimage (resp. collision) attack.

A second observation is that, in the case of a preimage attack, the dual code can be used to find the preimage far more efficiently than a naive method. Direct application of this method however is disappointing (see Table 1). The resulting time complexities are typically much higher than the corresponding query complexities and the attack is seldom competitive with that of Knudsen and Preneel, let alone with that of Watanabe.

4 Decoding the Knudsen-Preneel Preprocessing

An important property that is exploited by both Watanabe and ÖSS is linearity of C^{PRE}. Indeed, the image $\Im(C^{\text{PRE}})$ itself can be regarded as an ekn'-dimensional subspace of $\mathbb{F}_2^{ern'}$, or equivalently as an $[ern', ekn', d']_2$ code $\mathcal{C}^{\otimes}$ (where the minimum distance d' is largely irrelevant; it satisfies $d \leq d' \leq de$). This has the consequence that if $X = C^{\text{PRE}}(W)$ and $X' = C^{\text{PRE}}(W')$ collide, it is guaranteed that $\Delta = X \oplus X' \in \Im(C^{\text{PRE}})$, i.e. the difference Δ itself is a (nonzero) codeword in $\mathcal{C}^{\otimes}$. Below we will give a more detailed mathematical characterization of $\Im(C^{\text{PRE}})$, with a special eye towards the improved collision-finding algorithms we will give later on. Most of the results below are mathematically rather straightforward (and the proofs are left to the full version); the machinery is mainly needed to ensure that, when using canonical bases for the various vector spaces, everything lines up correctly and consistently with the actual Knudsen-Preneel compression function.

Recall that we are given an injective ring homomorphism $\varphi : \mathbb{F}_{2^e} \to \mathbb{F}_2^{e \times e}$ and a group isomorphism $\psi : \mathbb{F}_{2^e} \to \mathbb{F}_2^e$ that satisfy $\varphi(g)\psi(h) = \psi(gh)$ for all

$g, h \in \mathbb{F}_{2^e}$. Let $[r, k, d]_{2^e}$ be a linear code with generator matrix $G \in \mathbb{F}_{2^e}^{k\times r}$, let b be a positive divisor of e such that $ek > rb$ and finally let $n = bn'$ be a multiple of b. Then the input processing $C^{\text{PRE}} : \{0,1\}^{ekn'} \to \{0,1\}^{ern'}$ of the Knudsen-Preneel compression function is defined by $C^{\text{PRE}}(W) = (\bar{\varphi}(G^T) \otimes I_{n'}) \cdot W$ (and note that $ern' = rcn$).

Characterization of $\Im(C^{\text{PRE}})$ as a sum. We have already written the codomain of C^{PRE} as a direct sum of *PuRF inputs* by identifying $\{0,1\}^{ren'}$ with $\bigoplus_{i=1}^{r} V_i$ where $V_i = \mathbb{F}_2^{en'}$ for $i= 1, \dots, r$. Here we will use a second interpretation that emphasizes the *code*. We will consider $\bigoplus_{j=1}^{n'} U_j$ where $U_j = \mathbb{F}_{2^e}^r$ for $j= 1, \dots, n'$. Since $\mathbb{F}_{2^e}^r$, and by extension $\bigoplus_{j=1}^{n'} U_j$, is a vector space over $\mathbb{F}_{2^e}$, whereas $\{0,1\}^{ern'}$ is a stand-in for the vector space $\mathbb{F}_2^{ern'}$ over $\mathbb{F}_2$, we cannot find a vector space isomorphism (as for the earlier direct sum). Nonetheless we can find a suitable group isomorphism from $\bigoplus_{j=1}^{n'} U_j$ to $\{0,1\}^{ern'}$.

To define the group isomorphism we exploit that, luckily, the underlying $\mathbb{F}_{2^e}$ arithmetic is essentially preserved by $C^{\text{PRE}} : \{0,1\}^{ekn'} \to \{0,1\}^{ern'}$, even though the '$\otimes I_{n'}$' in $C^{\text{PRE}}(W) = (\bar{\varphi}(G^T) \otimes I_{n'}) \cdot W$ garbles things up. To formalize this, let $\rho : \mathbb{F}_{2^e}^{n'} \to \mathbb{F}_2^{en'}$ be the group isomorphism such that $\rho(g\delta) = (\varphi(g) \otimes I_{n'}) \cdot \rho(\delta)$ for all $\delta \in \mathbb{F}_{2^e}^{n'}$ and $g \in \mathbb{F}_{2^e}$.

As usual, we will extend ρ to e.g. r-tuples of elements in $\mathbb{F}_{2^e}^{n'}$ (and hence to vectors in $\mathbb{F}_{2^e}^{n'r}$) by component-wise application, i.e. $\bar{\rho} : \mathbb{F}_{2^e}^{n'r} \to \mathbb{F}_2^{en'r}$. This suffices for a group isomorphism from $\bigoplus_{j=1}^{n'} U_j$ to $\{0,1\}^{ern'}$ as well.

Lemma 1. *Let $\mathcal{I}_0 \subset \{1, \dots, r\}$, let $\mathcal{C}'$ be the (quasi) shortening of $\mathcal{C}$ on $\mathcal{I}_0$ and let $\mathcal{C}'_j = \mathcal{C}' \subseteq U_j$ for $j= 1, \dots, n'$. Then $X = \sum_{i=1}^{r} x_i \in \Im(C^{\text{PRE}})$ with $x_i = 0$ for all $i \in \mathcal{I}_0$ iff $\exists\ !\ \forall_{j=1,\dots,n'}\ g_j = \sum_{i=1}^{r} g_{ji} \in \mathcal{C}'_j$ such that $x_i = \rho(\sum_{j=1}^{n'} g_{ji})$.*

The following proposition develops the key idea on how to *recognize* that a given $X \in \mathbb{F}_2^{ern'}$ is an element of $\Im(C^{\text{PRE}})$. This result is exploited in [7] to efficiently find preimages for Knudsen-Preneel compression functions.

Proposition 1. *Let $H = \mathsf{KP}^b([r, k, d]_{2^e})$, $M \in \mathbb{F}_2^{e\times re/b}$ and a nonzero $X \in \mathbb{F}_2^{ern'}$ be given. Suppose that $M = \bar{\varphi}(h^T)$ for some $h \in \mathcal{C}^\perp$, then $X \in \Im(C^{\text{PRE}})$ iff for all positive integers n' it holds that $(M \otimes I_{n'}) \cdot X = 0$.*

Since $\mathbb{F}_{2^e}^{n'r}$ is isomorphic (as vector space over $\mathbb{F}_{2^e}$) to the tensor product $\mathbb{F}_{2^e}^r \otimes \mathbb{F}_{2^e}^{n'}$ this leads in a natural way to a function from $\mathbb{F}_{2^e}^r \times \mathbb{F}_{2^e}^{n'}$ to $\{0,1\}^{ren'}$ by considering pure tensors $g \otimes \delta$ with $g \in \mathbb{F}_{2^e}^r$ and $\delta \in \mathbb{F}_{2^e}^{n'}$. Note that we do not discriminate between different representatives, that is for nonzero $\beta \in \mathbb{F}_{2^e}$ we have that $g \otimes \delta = (\beta g) \otimes (\beta^{-1}\delta)$.

Lemma 2. *If $g \in \mathbb{F}_{2^e}^r$ and $\delta \in \mathbb{F}_{2^e}^{n'}$ then $\bar{\rho}(g \otimes \delta) \in \Im(C^{\text{PRE}})$ iff $g \in \mathcal{C}$ or $\delta = 0$.*

The following lemma states that invertibility of $G_{\tilde{\mathcal{I}}}$ suffices to invert C^{PRE}.

```
Algorithm 1 (Revised Watanabe Collision Attack).

Input: $H = \mathsf{KP}^b([r,k,d]_{2^e})$ satisfying $d \le k$, a nonzero $g \in \mathcal{C} \subseteq \mathbb{F}_{2^e}^r$ with
    $|\chi(g)| \le k$, a block size $n = bn'$, and an arbitrary nonzero $\delta \in \mathbb{F}_{2^e}^{n'}$.
Output: A colliding pair $(W, W') \in \left(\{0,1\}^{ekn'}\right)^2$ such that $H_n(W) =$
    $H_n(W')$, $W \ne W'$ and $C^{\text{PRE}}(W) \oplus C^{\text{PRE}}(W') = \bar{\rho}(g \otimes \delta)$.

1. INITIALIZATION. Compute $\Delta \leftarrow \bar{\rho}(g \otimes \delta)$, set $\mathcal{I} \leftarrow \chi(g)$ and determine
   $\tilde{\mathcal{I}} \supseteq \mathcal{I}$ for which $G_{\tilde{\mathcal{I}}}$ is invertible.
2. QUERY PHASE. For $i \in \mathcal{I}$ do
   a. Generate a random $x_i \xleftarrow{\$} V_i (= \mathbb{F}_2^{en'})$ and set $x'_i \leftarrow x_i \oplus \Delta_i$;
   b. Query $y_i \leftarrow f_i(x_i)$ and $y'_i \leftarrow f_i(x'_i)$;
   c. If $y_i = y'_i$ then keep $(x_i, x'_i)$ and proceed to next $i$, else return to a.
3. DEGREES OF FREEDOM. For $i \in \tilde{\mathcal{I}} \backslash \mathcal{I}$ pick $x_i \xleftarrow{\$} V_i$ and set $x'_i \leftarrow x_i$.
4. FINALIZATION. Output $(W, W')$ where

   $W \leftarrow (\bar{\varphi}(G_{\tilde{\mathcal{I}}}^{-T}) \otimes I_{n'}) \cdot (\sum_{i \in \tilde{\mathcal{I}}} x_i)$   and   $W' \leftarrow (\bar{\varphi}(G_{\tilde{\mathcal{I}}}^{-T}) \otimes I_{n'}) \cdot (\sum_{i \in \tilde{\mathcal{I}}} x'_i)$ .
```

Lemma 3. *Let G be a generator matrix for an $[r,k,d]_{2^e}$ code. Let $\tilde{\mathcal{I}} \subset \{1,\ldots,r\}$ be such that $G_{\tilde{\mathcal{I}}}$ is invertible, with transposed inverse $G_{\tilde{\mathcal{I}}}^{-T}$. Let n' be an integer and, for $i = 1,\ldots,r$, let $V_i = \mathbb{F}_2^{en'}$ be a direct-sum-decomposition of $\mathbb{F}_2^{ern'}$ as before. If given $x_i \in V_i$ for $i \in \tilde{\mathcal{I}}$, or equivalently $\tilde{X} = \sum_{i \in \tilde{\mathcal{I}}} x_i$, then*

$$W = (\bar{\varphi}(G_{\tilde{\mathcal{I}}}^{-T}) \otimes I_{n'}) \cdot \tilde{X}$$

is the unique element for which $X' = C^{\text{PRE}}(W)$ satisfies $x'_i = x_i$ for $i \in \tilde{\mathcal{I}}$.

5 A New Symbiotic Collision-Finding Attack

5.1 Revising Watanabe's Attack

Watanabe's attack has complexity $k2^n$, requires $k > r - k$ and essentially finds a single collision. Below we give a revised and improved version of his algorithm. It only has complexity $d2^n$, requires $k \ge d$ and it potentially results in many, many collisions. More precisely, if $k > d$ then after the initial effort (of $d2^n$) we can find a new collision in *constant* time, for up to a whopping $2^{(k-d)n}$ collisions.

In his note, Watanabe describes his attack as a differential attack. Where originally Δ was computed as some non-trivial kernel element, we will compute it based on a codeword $g \in \mathcal{C}$ of sufficiently low weight and an arbitrary (nonzero) 'block multiplier' δ. In particular, we will set $\Delta = \bar{\rho}(g \otimes \delta)$. By using a minimal weight codeword the attack performs best.

For the revised attack to work, we need one further ingredient. Watanabe assumes a systematic code and exploits that, when $k < r - k$, there exists a nonzero codeword $g \in \mathcal{C}$ for which $\chi(g) \subseteq \{1, \ldots, k\}$. This allows easy completion of a partial collision to a full collision. Our revised version allows an arbitrary (nonzero) codeword g of weight at most k (existence of which requires $d \leq k$). Thus $\chi(g)$ might no longer map to the systematic part of the code. Luckily, Lemma 3 provides completion to a full collision, provided $\mathcal{I} = \chi(g)$ is *admissible*. For MDS codes all codewords are admissible; for the four non-MDS codes proposed by Knudsen and Preneel it can be verified that the minimum distance codewords are admissible.

Theorem 1 (Revised Watanabe attack). *Let $H = \mathsf{KP}^b([r,k,d]_{2^e})$ be given with $d \leq k$. Consider H_n (with $b|n$). Then Algorithm 1 using a minimum-weight codeword g (and an arbitrary nonzero δ) finds collisions for H_n in expected time $d2^n$ (using as many PuRF evaluations).*

5.2 A New Symbiotic Attack

Our revised version of Watanabe's attack clearly shows that an attacker potentially has a lot of freedom. Below we transform some of this freedom into a faster attack. More to the point, as in the revised Watanabe attack we still look for a collision with differential $\Delta = \bar{\rho}(g \otimes \delta)$ and fix the codeword $g \in \mathcal{C}$, but we do *not* fix the multiplier δ up front. Instead we determine it based on the outcomes of the queries we make. To increase our success probability, we restrict to the same kind of queries as Özen, Shrimpton, and Stam did.

Theorem 2 (Symbiotic attack). *Let $H = \mathsf{KP}^b([r,k,d]_{2^e})$ be given with $k \geq d$. Consider H_n (with $b|n$). Then Algorithm 2 finds collisions for H_n in $d2^{dn/(d+1)}$ time (using as many PuRF evaluations) and memory (expressed in n-bit blocks).*

Proof (Sketch). We will leave showing the correctness of Algorithm 2 to the full version of this paper [8] and only prove here that a collision is expected and that the query and time complexities are as claimed.

Since $|\mathcal{X}| = 2^{\alpha n}$ by construction, the attack has the stated query complexity (per PuRF) for $\alpha = d/(d+1)$ since all queries are made during the Query Phase. Using a naive approach, Local Collision Detection step can be performed in roughly $2^{dn/(d+1)}$ comparisons resulting in partial collision lists of expected cardinality $|L_i| \approx 2^{(2\alpha-1)n}$ for $i \in \mathcal{I}$.

For Global Collision Detection, we just enumerate one partial collision list and check for membership against the others. Assuming constant time memory access, the time complexity of this step is at most $(d-1)\max_{i \in \mathcal{I}} |L_i|$. Since $\alpha < 1$ it follows that $2\alpha - 1 < \alpha$ making the Query Phase dominant with its time complexity of $2^{\alpha n}$.

Since we have d active PuRFs in total, the probability of finding a common element among d such lists is then $(\prod_i |L_i|)/|\mathcal{X}|^{d-1}$, or $2^{((2\alpha-1)d-\alpha(d-1))n}$. To ensure an expected number of collisions of one, we need the second exponent to be at least zero, and indeed, solving for zero gives the desired $\alpha = d/(d+1)$. $\square$

Algorithm 2 (New Symbiotic Collision Attack).

```
Input: $H = \mathsf{KP}^b([r,k,d]_{2^e})$ satisfying $d \le k$, a $g \in \mathcal{C} \subseteq \mathbb{F}_{2^e}^r$ with $|\chi(g)| = d$,
    and a block size $n = bn'$.
Output: A colliding pair $(W, W') \in \left(\{0,1\}^{ekn'}\right)^2$ such that $H_n(W) =$
    $H_n(W')$, $W \neq W'$ and $C^{\text{PRE}}(W) \oplus C^{\text{PRE}}(W') = \bar{\rho}(g \otimes \delta)$ for some nonzero
    $\delta \in \mathbb{F}_{2^e}^{n'}$ to be determined.

1. INITIALIZATION. Set $\alpha = d/(d+1), \mathcal{I} = \chi(g)$ and determine $\tilde{\mathcal{I}}$. Let $g =$
   $(g_1, \dots, g_r)$ with $g_i \in \mathbb{F}_{2^e}$ for $i = 1, \dots, r$.
2. QUERY PHASE. Define
       $\mathcal{X} = (\{0\}^{\frac{n}{b} - \frac{\alpha n}{e}} \times \{0,1\}^{\frac{\alpha n}{e}})^e$
   and, for $i \in \mathcal{I}$ let $Q_i = \mathcal{X} \subset V_i$. Query $f_i \; \forall \; x_i \in Q_i$ and store the results.
3. LOCAL COLLISION DETECTION. For $i \in \mathcal{I}$ create a list $L_i$ of all tuples
   $(g_i^{-1} \cdot \rho^{-1}(x_i \oplus x_i'), x_i, x_i')$ satisfying $x_i, x_i' \in Q_i, x_i \neq x_i'$ and $f_i(x_i) = f_i(x_i')$.
4. GLOBAL COLLISION DETECTION. Find a set of $|\chi(g)|$ tuples in the respec-
   tive $L_i$ that all share the same first element. That is, for some $\delta \in \mathbb{F}_{2^e}^{n'}$
   and $(x_i, x_i')_{i \in \mathcal{I}}$ it holds for all $i \in \mathcal{I}$ that $(\delta, x_i, x_i') \in L_i$.
5. DEGREES OF FREEDOM. For $i \in \tilde{\mathcal{I}} \backslash \mathcal{I}$ pick $x_i \xleftarrow{\$} V_i$ and set $x_i' \leftarrow x_i$.
6. FINALIZATION. Output $(W, W')$ where
       $W \leftarrow (\bar{\varphi}(G_{\tilde{\mathcal{I}}}^{-T}) \otimes I_{n'}) \cdot (\sum_{i \in \tilde{\mathcal{I}}} x_i)$   and   $W' \leftarrow (\bar{\varphi}(G_{\tilde{\mathcal{I}}}^{-T}) \otimes I_{n'}) \cdot (\sum_{i \in \tilde{\mathcal{I}}} x_i')$ .
```

6 A Parametrized Collision-Finding Attack

6.1 Optimizing the Query Complexity

The symbiotic attack and the information-theoretic attack by Özen, Shrimpton, and Stam have completely different query complexities and which one is the best seems very parameter dependent. However, it turns out that both attacks are the extreme cases of a more general parametrized attack, as given by Algorithm 3.

Theorem 3. *Let $H = \mathsf{KP}^b([r,k,d]_{2^e})$ be given. Consider H_n (with $b|n$). Then collisions for H_n can be found with Alg. 3 using $2^{\alpha n}$ queries (per PuRF) where*

$$\alpha = \begin{cases} (r-\theta)/(2k-\theta) & \text{for } 0 \le \theta \le \min(r-d, r-k)\ ; \\ (r-\theta)/(r+k-2\theta) & \text{for } r-k \le \theta \le r-d\ . \end{cases}$$

Proof. That the attack has the stated query complexity follows readily from the usual observation that $|\mathcal{X}| = 2^{\alpha n}$ combined with the computation of α exactly matching the theorem statement. What remains to show is that collisions are indeed output and expected with good probability.

For correctness, let (W, W') be output by the algorithm and consider $X = C^{\text{PRE}}(W)$ and $X' = C^{\text{PRE}}(W')$. First, notice that Lemma 3 implies that projecting

Algorithm 3 (Parameterized Collision Attack).

Input: $H = \mathsf{KP}^b([r,k,d]_{2^e})$, an index set $\mathcal{I}_0 \subset \{1,\dots,r\}$ with $\theta = |\mathcal{I}_0|$ and $0 \le \theta \le r-d$, and a block size $n = bn'$.

Output: A colliding pair $(W, W') \in \left(\{0,1\}^{ekn'}\right)^2$ such that $H_n(W) = H_n(W'), W \neq W'$, and if $X = C^{\text{PRE}}(W)$ and $X' = C^{\text{PRE}}(W')$ then for all $i \in \mathcal{I}_0$ it holds that $x_i = x'_i$.

```
1. INITIALIZATION. Set $\mathcal{I} \leftarrow \{1,\dots,r\}\backslash\mathcal{I}_0$, determine $\tilde{\mathcal{I}}$, and set
   $$\alpha \leftarrow \begin{cases} (r-\theta)/(2k-\theta) & \text{for } 0 \le \theta \le \min(r-k, r-d)\ ; \\ (r-\theta)/(r+k-2\theta) & \text{for } r-k \le \theta \le r-d\ . \end{cases}$$
2. QUERY PHASE. As in Algorithm 2.
3. LOCAL COLLISION DETECTION. For $i \in \mathcal{I}$ create a list $L_i$ of all tuples
   $(x_i \oplus x'_i, x_i, x'_i)$ satisfying $x_i, x'_i \in Q_i, x_i \neq x'_i$ and $f_i(x_i) = f_i(x'_i)$.
4. MERGE PHASE. Create $\tilde{L}_{\mathcal{I}} = \left\{\sum_{i\in\mathcal{I}} (\mathit{\Delta}_i, x_i, x'_i) \,|\, (\mathit{\Delta}_i, x_i, x'_i) \in L_i\right\}$.
5. COLLISION PRUNING. Create $L_{\mathcal{I}}$ consisting precisely of those elements of
   $\tilde{L}_{\mathcal{I}}$ whose first vector (when mapped to the full space) is in $\Im(C^{\text{PRE}})$;
   $$L_{\mathcal{I}} = \left\{(\tilde{\mathit{\Delta}}, \tilde{X}, \tilde{X}') | (\tilde{\mathit{\Delta}}, \tilde{X}, \tilde{X}') \in \tilde{L}_{\mathcal{I}} \wedge \tilde{\mathit{\Delta}} + 0 \in \Im(C^{\text{PRE}})\right\}\ .$$
6. FILTERING. If $\tilde{\mathcal{I}} \subset \mathcal{I}$ then only select $(\tilde{\mathit{\Delta}}, \tilde{X}, \tilde{X}') \in L_{\mathcal{I}}$ for which $\tilde{X}$ is
   in the projection of $\Im(C^{\text{PRE}})$ onto $\bigoplus_{i\in\mathcal{I}} V_i$. Create $L_{\tilde{\mathcal{I}}}$ by projecting the
   selected elements in $L_{\mathcal{I}}$ to the subspace $\bigoplus_{i\in\tilde{\mathcal{I}}} V_i$.
7. DEGREES OF FREEDOM. If $\mathcal{I} \subset \tilde{\mathcal{I}}$, then for $i \in \tilde{\mathcal{I}}\backslash\mathcal{I}$ pick $x_i \xleftarrow{\$} V_i$ and set
   $x'_i \leftarrow x_i$. Create $L_{\tilde{\mathcal{I}}}$ by adding $\sum_{i\in\tilde{\mathcal{I}}\cap\mathcal{I}_0} (0, x_i, x'_i)$ to all elements in $L_{\mathcal{I}}$.
8. SKIP. If $\tilde{\mathcal{I}} = \mathcal{I}$ set $L_{\tilde{\mathcal{I}}} \leftarrow L_{\mathcal{I}}$.
9. FINALIZATION. For some $(\tilde{\mathit{\Delta}}, \tilde{X}, \tilde{X}') \in L_{\tilde{\mathcal{I}}}$ output $(W, W')$ where
   $$W \leftarrow (\bar{\varphi}(G_{\tilde{\mathcal{I}}}^{-T}) \otimes I_{n'}) \cdot \tilde{X} \qquad \text{and} \qquad W' \leftarrow (\bar{\varphi}(G_{\tilde{\mathcal{I}}}^{-T}) \otimes I_{n'}) \cdot \tilde{X}'$$
```

$(X \oplus X', X, X')$ onto $\bigoplus_{i\in\tilde{\mathcal{I}}} V_i$ is in $L_{\tilde{\mathcal{I}}}$. Now, either of the steps Degrees of Freedom, Filtering or Skip ensures that $(\tilde{\mathit{\Delta}}, \tilde{X}, \tilde{X}') \in L_{\mathcal{I}}$. Finally, since $L_{\mathcal{I}} \subseteq \tilde{L}_{\mathcal{I}}$ it follows that $(x_i, x'_i) \in L_i$ for $i \in \mathcal{I}$ and hence by construction (Local Collision Detection) we have $f_i(x_i) = f_i(x'_i)$ for those i.

Moreover Collision Pruning guarantees that $\tilde{\mathit{\Delta}} + 0 \in \Im(C^{\text{PRE}})$ and Degrees of Freedom ensures that the projections of $\tilde{\mathit{\Delta}} + 0$ and $X \oplus X'$ onto $\bigoplus_{i\in\tilde{\mathcal{I}}} V_i$ are equal. Hence, $x_i = x'_i$ for all $i \in \mathcal{I}_0$.

Let us move on to the number of expected collisions output. Since $|\mathcal{X}| = 2^{\alpha n}$, the expected number of local collisions found per active PuRF for $i \in \mathcal{I}$ is $|L_i| \approx |\mathcal{X}|^2/2^n = 2^{(2\alpha-1)n}$. Using that $|\mathcal{I}| = r - \theta$ we arrive at a total number of potential collisions of $|\tilde{L}_{\mathcal{I}}| \approx 2^{(2\alpha-1)(r-\theta)n}$. For a true collision to occur, we need to find a tuple $(x_i, x'_i)_{i\in\tilde{\mathcal{I}}}$ such that both $\sum_{i\in\tilde{\mathcal{I}}} x_i$ and $\sum_{i\in\tilde{\mathcal{I}}} x'_i$ can be completed to codewords subject to the constraint that $x_i = x'_i$ for $i \in \mathcal{I}_0$.

If the eventual collision consists of (X, X'), then $\varDelta = X \oplus X'$ is a codeword as well and the above implies that $\varDelta_i = 0$ for $i \in \mathcal{I}_0$. Hence, Lemma 1 applies and $\tilde{\varDelta} = \sum_{i \in \mathcal{I}} \varDelta_i$ is somehow 'spanned' by the shortened code. The restriction $\theta \leq r - d$ ensures nontriviality of the shortened code (shortening any further and the shortened code would consist of the zero codeword only resulting in $W = W'$, so no collision). In case of MDS codes, the shortened code has parameters $[r - \theta, k - \theta, d']_{2^e}$, in particular it has dimension $k - \theta$. (For non-MDS codes it is possible that a higher dimension is achieved.)

As a result, a fraction $2^{(k-r)\alpha n}$ of the differentials will be satisfactory, leading to an expected number of $|L_{\mathcal{I}}| \approx 2^{((2\alpha-1)(r-\theta)-\alpha(r-k))n}$. If $\mathcal{I} \subseteq \tilde{\mathcal{I}}$ or equivalently $r - \theta \leq k$ we are done whenever $|L_{\mathcal{I}}| \geq 1$. Since $r - \theta \leq k$ can be rewritten to $\theta \geq r - k$ we are in the second case, with $\alpha = (r - \theta)/(r + k - 2\theta)$. Writing $F = (\lg |L_{\mathcal{I}}|)/n$ and substitution lead to $F \approx (2\alpha - 1)(r - \theta) - \alpha(r - k) = \alpha(2r - 2\theta - r + k) - (r - \theta) = 0$ or $|L_{\mathcal{I}}| \approx 1$ as desired.

If on the other hand $\tilde{\mathcal{I}} \subset \mathcal{I}$, further filtering is needed. In particular, given a potential 'half' of a collision X we need to check if it can correspond to a codeword. Since $\tilde{\mathcal{I}} \subset \mathcal{I}$, we can uniquely complete X to a codeword given k of its elements (all within $\mathcal{I}$). The remaining $|\mathcal{I}| - k$ coordinates need to be in sync. Per remaining element, this occurs with probability $2^{-\alpha n}$, leading to $|\tilde{L}_{\tilde{\mathcal{I}}}| \approx |L_{\mathcal{I}}| \cdot 2^{-\alpha n(r-\theta-k)}$. Now we are in the first case since $0 \leq \theta \leq r - k$. Writing $F = (\lg \tilde{L}_{\tilde{\mathcal{I}}})/n$, we obtain $F \approx ((2\alpha-1)(r-\theta)-\alpha(r-k))-\alpha(r-\theta-k) = \alpha(2k - \theta) - (r - \theta)$. Since we aim for $F = 0$, $\alpha = (r - \theta)/(2k - \theta)$ as desired. □

Corollary 1. *Assuming $d \leq k$, substitution of $\theta = r - k$ in Theorem 3 gives $\alpha = k/(3k - r)$. This is optimal (for Algorithm 3) whenever $r \leq 2k$.*

Proof. That the substition does what it says can be readily verified, so we restrict ourselves to prove the optimality here. Let $f_1(\theta) = (r - \theta)/(2k - \theta)$ and $f_2(\theta) = (r - \theta)/(r + k - 2\theta)$ be two real valued functions defined over closed intervals $0 \leq \theta \leq r - k$ and $r - k \leq \theta \leq r - d$ respectively. Note that both $f_1(\theta)$ and $f_2(\theta)$ are continuous in their respective domains (since their respective poles fall outside the domains). So both $f_1(\theta)$ and $f_2(\theta)$ attain their maximum and minimum in the closed intervals $[0, r - k]$ and $[r - k, r - d]$ respectively. Since $f_1'(\theta) = (r - 2k)/(2k - \theta)^2 \leq 0$ (for $r \leq 2k$) and $f_2'(\theta) = (r - k)/(r + k - 2\theta)^2 \geq 0$ we can conclude that $f_1(\theta)$ is decreasing and $f_2(\theta)$ is increasing. Therefore, they both attain their minimum at their shared boundary $\theta = r - k$. □

Remark 1. The only two parameter sets proposed by Knudsen and Preneel not satisfying the conditions of the corollary above are $[4, 2, 3]_8$ and $[5, 2, 4]_8$. In both cases $d > k$ and only $f_1(\theta)$ is applicable. For $[5, 2, 4]_8$ one can check that $2k < r$ and $f_1'(\theta) \geq 0$. Hence, the minimum α is attained at $\theta = 0$. For $[4, 2, 3]_8$ it holds that $2k = r$, so that $f_1(\theta)$ is in fact a constant function and both $\theta = 0$ and $\theta = 1$ lead to the same α.

Remark 2. Substitution of $\theta = 0$ in Theorem 3 gives $\alpha = r/(2k)$ and the resulting query complexity coincides with that reported by Özen, Shrimpton, and Stam. On the other extreme, substitution of $\theta = r - d$ gives $\alpha = d/(2d - r + k)$

(assuming $d \leq k$). For MDS codes this simplifies to $\alpha = d/(d+1)$, this time duly coinciding with our symbiotic attack. For non-MDS codes there seems to be a slight mismatch. The reason is that if a non-MDS code is maximally shortened (by $\theta = r - d$), the shortened code has dimension 1, whereas in the derivation of Theorem 3 we pessimistically assumed $k - \theta = 0$ (at least for the KP non-MDS codes that satisfy $r - d = k$). Correcting for this looseness would result in a match with the symbiotic attack.

6.2 Generic Collision Attack against MDS Constructions

If we want to run Algorithm 3 (with fixed $\theta = r-k$ and $\alpha = k/(3k-r)$ as obtained in Corollary 1) we ideally want a time complexity almost coinciding with the targeted query complexity. For $\theta = r - k$ it holds that $\mathcal{I} = \tilde{\mathcal{I}}$, obviating the need for the steps Filtering and Degrees of Freedom. We have already seen that Local Collision Detection costs at most a small logarithmic factor, which leaves only the Merge Phase and Collision Pruning to worry about. Together, these two steps are designed to produce $L_{\mathcal{I}}$. A naive approach would enumerate all elements in the much larger $\tilde{L}_{\mathcal{I}}$, which is wasteful. Our task is therefore, given the lists of partial collisions L_i for $i \in \mathcal{I}$, to create $L_{\mathcal{I}}$ more efficiently.

In the sequel, we will follow in the footsteps of Özen, Shrimpton, and Stam who used the dual code in a similar problem related to their preimage-finding attack. An important innovation for the collision-finding attack stems from the realization that Δ can be regarded as belonging to the (quasi) shortened code. This allows the use of the dual of the shortened code to speed up the search. As the minimum distance of the dual code is an important parameter in determining the overall time-complexity and shortening a code reduces the minimum distance of its dual accordingly, we make a significant efficiency gain this way.

Road map. We present our collision attack against Knudsen-Prennel compression functions whose C^{PRE} is based on MDS codes in Alg. 4, whereas its analysis is given in Thm. 4. We leave the generalization of our attack to (KP-suggested) non-MDS parameters together with the proof of Thm. 4 to the full version of this work where we also investigate a more space efficient version of Alg. 4.

Reducing the Time Complexity. Since $\mathcal{I} = \tilde{\mathcal{I}}$ and $\theta = r - k$, we know from Algorithm 3 that it is enough to find a nonzero $\Delta \in \Im(C^{\text{PRE}})$ of the form $\Delta = \Delta' + 0$ for $\Delta' = \sum_{i \in \tilde{\mathcal{I}}} \Delta_i$ to complete the collision. Now notice that Δ' is lying in a smaller space $\Im({C'}^{\text{PRE}})$ identified by $\mathcal{C}'$ that is the $[r - \theta, k - \theta, d']$ shortened code obtained from $\mathcal{C}$ (by dropping the zeroes of the codewords from all the positions appearing in $\mathcal{I}_0$). This observation allows us to guarantee that $\Delta \in \Im(C^{\text{PRE}})$ once we determine that a candidate Δ' is in $\Im({C'}^{\text{PRE}})$. Hence, it is enough for our purposes to limit ourselves to $\Im({C'}^{\text{PRE}})$ rather than looking for membership in the larger space $\Im(C^{\text{PRE}})$.

To this end, we first identify an index set $\mathcal{I}_{h'} \subseteq \{1, \dots, r\}$ (the role of h' will be explained momentarily) defining a subspace $\bigoplus_{i \in \mathcal{I}_{h'}} V_i$ for which $\Im({C'}^{\text{PRE}})$

Algorithm 4 (Collision Attack against MDS-based schemes).

Input: $H = \mathsf{KP}^b([r,k,d]_{2^e})$, an index set $\mathcal{I}_0 \subset \{1,\ldots,r\}$ with $\theta = |\mathcal{I}_0| = r-k$ and a block size $n = bn'$.

Output: A colliding pair $(W, W') \in \left(\{0,1\}^{ekn'}\right)^2$ such that $H_n(W) = H_n(W'), W \neq W'$, and if $X = C^{\text{PRE}}(W)$ and $X' = C^{\text{PRE}}(W')$ then for all $i \in \mathcal{I}_0$ it holds that $x_i = x'_i$.

1. Initialization. Set $\mathcal{I} \leftarrow \{1,\ldots,r\}\backslash\mathcal{I}_0$ (with $|\mathcal{I}| = k$), $\mathcal{I} = \tilde{\mathcal{I}}$, and set $\alpha \leftarrow k/(3k-r)$. Obtain $\mathcal{C}'$ consisting of codewords $g' \in \mathcal{C}'$ that are constructed from $g \in \mathcal{C}$ by dropping zeroes of g from all the positions appearing in $\mathcal{I}_0$.
2. Query Phase. As in Algorithm 3.
3. Local Collision Detection. As in Algorithm 3.
4. Merge Phase. Find a nonzero codeword $h' \in \mathcal{C}'^{\perp}$ of minimum Hamming weight $d'^{\perp} = 2k - r + 1$. Let $h' = h'_0 + h'_1$ with $\chi(h'_0) \cap \chi(h'_1) = \emptyset$ and of Hamming weights $\lceil d'^{\perp}/2 \rceil$ and $\lfloor d'^{\perp}/2 \rfloor$ respectively. Create for $j = 0, 1$,
$$\tilde{L}_{h'_j} = \left\{ \left(\Delta'_{h'_j}, X_j, X'_j, (\bar{\varphi}(h'_j) \otimes I_{n'}) \cdot (\Delta'_{h'_j} + 0)\right) \mid (\Delta'_{h'_j}, X_j, X'_j) \in \sum_{i \in \chi(h'_j)} L_i \right\}$$
both sorted on their fourth component.
5. Join Phase. Create $L_{h'}$ consisting exactly of those elements $\Delta'_{h'_0} + \Delta'_{h'_1}$ for which $(\Delta'_{h'_0}, X_0, X'_0, Y_0) \in \tilde{L}_{h'_0}, (\Delta'_{h'_1}, X_1, X'_1, Y_1) \in \tilde{L}_{h'_1}$ and $Y_0 = Y_1$.
6. Collision Pruning. For all $(\Delta'_{h'}, X, X') \in L_{h'}$ create the unique Δ' corresponding to it and check whether it results in $\Delta_i \in L_i$ for all $i \in \mathcal{I}(= \tilde{\mathcal{I}})$. If so, keep $\Delta' = \sum_{i \in \tilde{\mathcal{I}}} \Delta_i$ in $L_{\mathcal{I}}$. Formally
$$L_{\mathcal{I}} = \left\{ (\Delta', \tilde{X}, \tilde{X}') = (\Delta'_{h'}, X, X') \in L_{h'} + \sum_{i \in \tilde{\mathcal{I}} \setminus \chi(h')} L_i | \Delta' \in \Im(C'^{\text{PRE}}) \right\} .$$
7. Skip. & 8. Finalization. As in Algorithm 3.

when restricted to this subspace, is not surjective. As a consequence, we will be able to prune significantly the total collection of candidate Δ's keeping only those that are possibly in $\Im(C'^{\text{PRE}})$ (restricted to $\bigoplus_{i \in \mathcal{I}_{h'}} V_i$). In the sequel, we will show how to *efficiently* find an index set $\mathcal{I}_{h'}$, and how to *efficiently* prune.

An important parameter determining the runtime of our collision attack is $d'^{\perp}$, the minimum distance of the dual shortened code. Let χ be the function that maps $h' \in \mathbb{F}_{2^e}^{r-\theta}$ to the set of indices of non-zero entries in h'. Thus, $\chi(h') \subseteq \{1,\ldots,r\}$ and $|\chi(h')|$ equals the Hamming weight of the codeword h'.

An easy adaptation of Proposition 1 shows that if we are given a codeword $h' \in \mathcal{C}'^{\perp}$ and an element $\Delta' \in \mathbb{F}_2^{(r-\theta)en'}$, then Δ' can only be in $\Im(C'^{\text{PRE}})$ if $(\bar{\varphi}(h'^T) \otimes I_{n'}) \cdot \Delta' = 0$, where the only parts of Δ' relevant for this check are those lining up with the nonzero entries of h'. Indeed, an element $\Delta'_{h'} \in \sum_{i\in\chi(h')} L_i$ can be completed to an element in the range of derived mapping C'^{PRE} iff $(\bar{\varphi}(h'^T) \otimes I_{n'}) \cdot (\Delta'_{h'} + 0) = 0$. Efficient creation of

$$L_{h'} = \left\{ (\Delta'_{h'}, X, X') \in \sum_{i\in\chi(h')} L_i \mid (\bar{\varphi}(h'^T) \otimes I_{n'}) \cdot (\Delta'_{h'} + 0) = 0 \right\}$$

can be done adapting standard techniques [2,13,11] by splitting the codeword in two and looking for all collisions in respective entries. That is, assume that $h' = h'_0 + h'_1$ with $\chi(h'_0) \cap \chi(h'_1) = \emptyset$, and define, for $j = 0, 1$

$$\tilde{L}_{h'_j} = \left\{ \left(\Delta'_{h'_j}, X_j, X'_j, (\bar{\varphi}(h'^T_j) \otimes I_{n'}) \cdot (\Delta'_{h'_j} + 0)\right) \mid (\Delta'_{h'_j}, X_j, X'_j) \in \sum_{i\in\chi(h'_j)} L_i \right\}$$

Then $L_{h'}$ consists of those elements $\Delta'_{h'_0} + \Delta'_{h'_1}$ for which $(\Delta'_{h'_0}, X_0, X'_0, Y_0) \in \tilde{L}_{h'_0}$, $(\Delta'_{h'_1}, X_1, X'_1, Y_1) \in \tilde{L}_{h'_1}$ and $Y_0 = Y_1$.

By sorting the two $\tilde{L}$'s the time complexity of creating $L_{h'}$ is then roughly the maximum cardinality of the two sets $\tilde{L}_{h'_0}$ and $\tilde{L}_{h'_1}$. Hence, the main trick to reduce the time complexity is to minimize the Hamming weights of h'_0 and h'_1, which is done by picking a codeword $h' \in \mathcal{C}'^{\perp}$ of minimum distance $d'^{\perp}$ and splitting it (almost) evenly. As a result, for the partial collision lists of (almost) same cardinality S, $L_{h'}$ can be constructed in $S^{\lceil d'^{\perp}/2 \rceil}$ time using $S^{\lfloor d'^{\perp}/2 \rfloor}$ memory (ignoring inconsequential factors). We summarize our analysis in Thm. 4.

Theorem 4. *Let $H = \mathsf{KP}^b([r,k,d]_{2^e})$ be given and $\mathcal{C}'$ be a shortened $[r-\theta, k-\theta, d]_{2^e}$ code derived from $\mathcal{C}$ for $\theta = r-k$. Let $d'^{\perp}$ be the minimum distance of the dual code of $\mathcal{C}'$. Suppose $\mathcal{C}$ is MDS (so is $\mathcal{C}'$ with $d'^{\perp} = 2k-r+1$) and consider the collision attack described in Alg. 4 run against H_n using $q = 2^{\alpha n}$ queries for $\alpha = k/(3k-r)$. Then the expected number of collision outputs is equal to one and the expectations for the internal list sizes are (for $i \in \mathcal{I}$):*

$$|L_i| = 2^{(2\alpha-1)n} \ , \ |L_{h'}| = 2^{((2\alpha-1)d'^{\perp}-\alpha)n} \ ,$$
$$|\tilde{L}_{h'_0}| = 2^{(2\alpha-1)\lceil \frac{d'^{\perp}}{2} \rceil n} \ , \ |\tilde{L}_{h'_1}| = 2^{(2\alpha-1)\lfloor \frac{d'^{\perp}}{2} \rfloor n}$$

The average case time complexity of the algorithm is $\max\left(q, |\tilde{L}_{h'_0}|, |L_{h'}|\right)$ *with a memory requirement of* $\max\left(q, |\tilde{L}_{h'_1}|\right)$ *(expressed in cn-bit blocks).*

7 Conclusion

In this paper we provide an extensive security analysis of the Knudsen-Preneel compression functions by focusing on their collision resistance. We present three improved collision attacks namely the revised Watanabe, symbiotic collision and parametrized collision attacks. Our new attacks work with the least number of queries reported so far. Moreover, except for only one out of 16 suggested parameters, these attacks beat the time-complexity of any prior attack we are aware of.

Acknowledgments. We thank Joachim Rosenthal and Thomas Shrimpton for their useful comments and suggestions.

References

1. Black, J., Rogaway, P., Shrimpton, T.: Black-box analysis of the block-cipher-based hash-function constructions from PGV. In: Yung [15], pp. 320–335
2. Chose, P., Joux, A., Mitton, M.: Fast correlation attacks: An algorithmic point of view. In: Knudsen, L. (ed.) Advances in Cryptography—Eurocrypt'02. LNCS, vol. 2332, pp. 209–221. Springer, Heidelberg (2002)
3. Knudsen, L., Muller, F.: Some attacks against a double length hash proposal. In: Roy, B.K. (ed.) Advances in Cryptography—Asiacrypt'05. LNCS, vol. 3788, pp. 462–473. Springer, Heidelberg (2005)
4. Knudsen, L.R., Preneel, B.: Hash functions based on block ciphers and quaternary codes. In: Kim, K., Matsumoto, T. (eds.) Advances in Cryptography—Asiacrypt'96. LNCS, vol. 1163, pp. 77–90. Springer, Heidelberg (1996)
5. Knudsen, L.R., Preneel, B.: Fast and secure hashing based on codes. In: Burt Kaliski, J., Burton, S. (eds.) Advances in Cryptography—Crypto'97. LNCS, vol. 1294, pp. 485–498. Springer, Heidelberg (1997)
6. Knudsen, L.R., Preneel, B.: Construction of secure and fast hash functions using nonbinary error-correcting codes. IEEE Transactions on Information Theory 48(9), 2524–2539 (2002)
7. Özen, O., Shrimpton, T., Stam, M.: Attacking the Knudsen-Preneel compression functions. In: Hong, S., Iwata, T. (eds.) FSE'10. LNCS, vol. 6147, pp. 94–115. Springer, Heidelberg (2010)
8. Özen, O., Stam, M.: Collision attacks against Knudsen-Preneel compression functions, full version of this paper, Manuscript (2010)
9. Özen, O., Stam, M.: Another glance at double-length hashing. In: Parker, M.G. (ed.) CCC'09. LNCS, vol. 5921, pp. 176–201. Springer, Heidelberg (2009)
10. Preneel, B., Govaerts, R., Vandewalle, J.: Hash functions based on block ciphers: A synthetic approach. In: Stinson, D. (ed.) Advances in Cryptography—Crypto'93. LNCS, vol. 773, pp. 368–378. Springer, Heidelberg (1993)
11. Schroeppel, R., Shamir, A.: A $T = O(2^{n/2})$, $S = O(2^{n/4})$ algorithm for certain NP-complete problems. SIAM Journal on Computing 10, 456–464 (1981)
12. Stam, M.: Blockcipher-based hashing revisited. In: Dunkelman, O. (ed.) FSE'09. LNCS, vol. 5665, pp. 67–83. Springer, Heidelberg (2009)
13. Wagner, D.: A generalized birthday problem. In: Yung [15], pp. 288–303
14. Watanabe, D.: A note on the security proof of Knudsen-Preneel construction of a hash function (2006), unpublished manuscript, Available at `http://csrc.nist.gov/groups/ST/hash/documents/WATANABE_kp_attack.pdf`
15. Yung, M. (ed.): Advances in Cryptography—Crypto'02, LNCS, vol. 2442. Springer, Heidelberg (2002)
16. Yuval, G.: How to swindle Rabin. Cryptologia 3, 187–189 (1979)