Blockcipher-Based Hashing Revisited

Martijn Stam*

LACAL, EPFL, Switzerland
martijn.stam@epfl.ch

Abstract. We revisit the rate-1 blockcipher based hash functions as first studied by Preneel, Govaerts and Vandewalle (Crypto'93) and later extensively analysed by Black, Rogaway and Shrimpton (Crypto'02). We analyse a further generalization where any pre- and postprocessing is considered. This leads to a clearer understanding of the current classification of rate-1 blockcipher based schemes as introduced by Preneel et al. and refined by Black et al. In addition, we also gain insight in chopped, overloaded and supercharged compression functions. In the latter category we propose two compression functions based on a single call to a blockcipher whose collision resistance exceeds the birthday bound on the cipher's blocklength.

1 Introduction

One of the oldest ideas to create a hash function is to base it on a blockcipher (e.g.,[11, 12, 14, 15]). Preneel et al. [15] studied the general construction $H(M, V) = E(K, X) \oplus U$ where $K, X, U \in \{0, M, V, M \oplus V\}$ (or affine offsets thereof). They concluded that of the $4^3 = 64$ possibilities all but 12 allow collision attacks on the compression function with a complexity beating the birthday bound of $2^{n/2}$. Later Black et al. [5] showed that in the ideal cipher model these 12 compression functions are indeed collision resistant up to the birthday bound, an additional 8 constructions were shown secure when properly iterated. Duo and Li [7] later gave an alternative proof resulting in improved bounds. Unfortunately neither of these articles provides a deeper understanding of what makes these 12 respectively 8 schemes special to make them secure as compression function respectively as iterated hash function: what do they have in common that sets them apart from the other 44 schemes?

We isolate the properties that make Duo and Li's proof go through in the ideal cipher model for the collision resistance of rate-1 blockcipher based compression functions and their iterated hash functions. This sheds new light on what it is that provides the provable security for these schemes; indeed the classification by Black et al. can be *derived* from it. Central to our result is a more general type of compression function, consisting of the following three simple steps (see Figure 1):

* This work was partially funded by the European Commission through the ICT programme under Contract ICT-2007-216646 ECRYPT II.

1. Prepare key and plaintext: $(K, X) \leftarrow C^{\text{PRE}}(M, V)$;
2. Make the call: $Y \leftarrow E(K, X)$;
3. Output the digest: $W \leftarrow C^{\text{POST}}(M, V, Y)$.

Here E is a blockcipher (where key size $k = |K|$ and blocksize $n = |X| = |Y|$ may differ) and C^{PRE} and C^{POST} can be arbitrary functions given their respective domain and codomain. To avoid complications we will initially assume that input and output sizes of the compression function match those of the blockcipher, that is $m := |M| = k$ and $s := |V| = |W| = n$.

Similar to prior art we consider two types of schemes. Type-I schemes give rise to collision resistant compression functions whereas Type-II schemes give rise to compression functions that will turn into collision resistant hash functions when (Merkle-Damgård) iterated. Each type is defined by a set of three conditions on C^{PRE} and C^{POST}. Both types share the first two conditions and only differ in the third. The first condition is bijectivity of C^{PRE}, ensuring that each query to E (or its inverse) can only be used to evaluate the compression function for a single input. The second condition is that for all M, V the postprocessing $C^{\text{POST}}(M, V, \cdot)$ is bijective. This causes optimal transfer of unpredictabilibity of encryption answers to the output W. For Type-I schemes, the third condition is similar in nature to the second, making sure that the unpredictability of decryption answers carries over to the digest W as well. Formally, for all K, Y the modified postprocessing $C^{\text{POST}}(C^{-\text{PRE}}(K, \cdot), Y)$ should be bijective. For Type-II schemes, the third condition captures that for each decryption answer the corresponding input chaining variable V is highly unpredictable. Formally, for all K, the function $C^{-\text{PRE}}(K, \cdot)$ restricted to its second output V is bijective.

We provide a proof in the ideal cipher model that the probability of finding a collision in the compression function (for Type-I) respectively in the iterated hash function (for Type-II) is upper bounded by $\frac{1}{2}q(q-1)/(2^n - q)$, where q is the number of queries allowed to the adversary and n is the block size. For Type-I schemes (everywhere) preimage resistance is upper bounded by $q/(2^n - q)$. We also investigate the ramifications of our general classification for the classical PGV schemes. We conclude that the Type-I schemes are exactly those 12 identified before by Preneel et al. and later Black et al. Our Type-II schemes include the 8 schemes identified as Type-II by Black et al., plus an additional 8 schemes that were already known to be Type-I.

The benefits of our generalized framework become even clearer when analysing three more complex scenarios, when the restrictions on the parameters n, k, s, and m are being relaxed. Here we achieve the following results:

Chopped Compression Functions. This corresponds to having an output size s of the compression function smaller than the blocksize n of the underlying blockcipher. A possible example is chopped Davies-Meyer; we show that, as one might expect, it is optimally collision resistant and preimage resistant. Note that chopping the output after each encryption frees up $n - s$ bits extra for message bits if we want to maintain $n + k = s + m$. In particular one can achieve compression even for fixed-key ($k = 0$) blockciphers.

Overloaded Compression Functions. Here one tries to cram the compression function by having more input to the compression function than the blockcipher can handle, i.e., $s + m > n + k$. Examples are the sponge construction [3, 4] or the (related) compression function of Cubehash [2]. Our bound on collision resistance of the compression function is worse than if we would chop the chaining variable (to make space for the message), which is partially due to an overly loose bound.

Supercharged Compression Functions. The exact opposite of the previous two cases, since here one attempts to boost collision resistance beyond the birthday bound on the blocksize by setting $s > n$. We present a general framework for the collision resistance of single call compression functions in the ideal cipher model. In particular, we give a variant of Stam's construction [18], collision resistant in the ideal cipher model (against adaptive adversaries). We also give a rate-1/2 compression function with collision resistance up to $2^{3n/4}$ queries based on a blockcipher with $k = n$ bit keys.

2 Background

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).

For positive integers k and n, we let $\mathrm{Block}(k, n)$ denote the set of all blockciphers with k-bit key and operating on n-bit blocks. Given that $E(K, \cdot)$ is a permutation for all $K \in \{0,1\}^k$, we write $D(K, \cdot)$ for its inverse.

Unless otherwise specified, all finite sets are equipped with a uniform distribution for random sampling. We use the convention to write oracles that are provided to an algorithm as superscripts.

2.1 Compression Functions and Hash Functions

A *compression function* is a mapping H from $\{0,1\}^m \times \{0,1\}^s$ to $\{0,1\}^s$ for some $m, s > 0$. A *blockcipher-based* compression function is a mapping $H : \{0,1\}^m \times \{0,1\}^s \to \{0,1\}^s$ given by a program that, given (M, V), computes $H^E(M, V)$ via access to an oracle $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ modeling an (ideal) blockcipher with k-bit key and operating on n-bit blocks. A *single-call* blockcipher-based compression function calls its encryption oracle only once. Compression of a message block then proceeds as follows: Given an s-bit state V and m-bit message M, compute output $W = H^E(M, V)$ by

1. Compute $(K, X) \leftarrow C^{\text{PRE}}(M, V)$.
2. Set $Y \leftarrow E(K, X)$.
3. Output $W \leftarrow C^{\text{POST}}(M, V, Y)$.

as illustrated by Figure 1. We will refer to $C^{\text{PRE}} : \{0,1\}^m \times \{0,1\}^s \to \{0,1\}^k \times \{0,1\}^n$ as preprocessing and to $C^{\text{POST}} : \{0,1\}^m \times \{0,1\}^s \times \{0,1\}^n \to \{0,1\}^s$ as postprocessing.

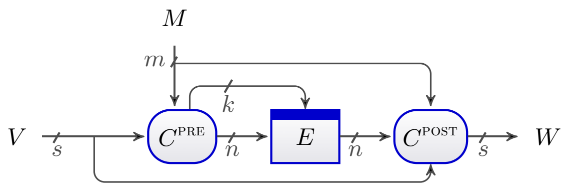

Fig. 1. General form of a $m + s$-to-s bit compression function based on a single call to the underlying blockcipher with k-bit key operating on n-bit block

Since a blockcipher is easy to invert (given its key), an adversary trying to find for instance collisions will also have access to D. To deal with inverse queries in our security analysis, we introduce the modified postprocessing $C^{\text{AUX}}(K, X, Y) = C^{\text{POST}}(C^{-\text{PRE}}(K, X), Y)$. In general, this is a function mapping triplets of strings to subsets of strings, since the result of $C^{-\text{PRE}}$ can have varying cardinality. For simplicity, when C^{PRE} is bijective, we understand C^{AUX} to have $\{0,1\}^n$ as its codomain.

A *hash function* is a mapping $\mathcal{H}$ from $\{0,1\}^*$ (the set of arbitrary length bitstrings) to $\{0,1\}^s$ for some $s > 0$. A compression function can be made into a hash function by iterating it. We briefly recall the standard Merkle-Damgård iteration [6, 13], where we assume that there is already some injective padding from $\{0,1\}^* \to (\{0,1\}^m)^* \setminus \emptyset$ in place (note that we disallow the empty message $\mathbf{M} = \emptyset$ as output of the injective padding). Given an initial vector $V_0 \in \{0,1\}^s$ define $\mathcal{H}^H : (\{0,1\}^m)^* \to \{0,1\}^s$ as follows for $\mathbf{M} = (M_1, \ldots, M_\ell)$ with $\ell > 0$:

1. Set $V_i \leftarrow H^E(M_i, V_{i-1})$ for $i = 1, \ldots, \ell$.
2. Output $\mathcal{H}^H(\mathbf{M}) = V_\ell$.

(Bearing this iteration in mind, given a compression function $H : \{0,1\}^m \times \{0,1\}^s \to \{0,1\}^s$ we will refer to the $\{0,1\}^m$ part of the input as 'message' and the $\{0,1\}^s$ part as the state or chaining variable.)

Collision Resistance. 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 or hash function. It is standard practice to consider information-theoretic adversaries only. Currently this seems to provide the only handle to get any provable results. Information-theoretic adversaries are computationally unbounded and their complexity is measured only by the number of queries made to their oracles. Without loss of generality, such adversaries are assumed not to repeat queries to oracles nor to query an oracle outside of its specified domain. We also assume that the adversary, before outputting a message, makes all calls necessary to evaluate the compressing function on that message. This does not decrease the advantage of the adversary, though it does increase its query complexity.

Despite the concept of initial vector being somewhat alien to a compression function on its own, it turns out helpful to consider a preimage to the initial vector a collision [5].

Definition 1. *Let $n, k, m, s > 0$ be integer parameters. Let $H\colon \{0,1\}^m \times \{0,1\}^s \to \{0,1\}^s$ be a compression function taking oracle $E \in \mathrm{Block}(k,n)$. The collision-finding advantage of adversary $\mathcal{A}$ is defined to be*

$$\mathsf{Adv}_H^{\mathsf{coll}}(\mathcal{A}) = \max_{V_0 \in \{0,1\}^s} \Pr\Big[E \xleftarrow{\$} \mathrm{Block}(k,n), ((M,V),(M',V')) \leftarrow \mathcal{A}^{E,D}(V_0) : \\ (M,V) \neq (M',V') \text{ and } H^E(M,V) \in \{V_0, H^E(M',V')\}\Big] \ .$$

Define $\mathsf{Adv}_H^{\mathsf{coll}}(q)$ as the maximum advantage over all adversaries making at most q queries in total.

The quantity $\mathsf{Adv}_{\mathcal{H}}^{\mathsf{coll}}(q)$ denoting collision for the iterated hash function $\mathcal{H}^H$ is defined similarly: in this case the advantage of $\mathcal{A}$ is the maximum success probability taken over the choice of possible initial values V_0, which is input to $\mathcal{A}$. It is well known that the iterated hash function $\mathcal{H}$ is at least as secure as the compression function H it is based upon, as far as collision resistance is concerned [5, Lemma 1].

Theorem 2. *Let H be a blockcipher based compression function and let $\mathcal{H}$ be the iterated hash function based on H. Then*

$$\mathsf{Adv}_{\mathcal{H}}^{\mathsf{coll}}(q) \leq \mathsf{Adv}_H^{\mathsf{coll}}(q) \ .$$

Preimage Resistance. A *preimage-finding adversary* is an algorithm with access to one or more oracles, whose goal it is to find preimages in some specified compression function. There exist several definitions depending on the distribution of the element of which a preimage needs to be found. We opt for everywhere preimage resistance [16], which intuitively states that all points are hard to invert.

Definition 3. *Let $n, k, m, s > 0$ be integer parameters. Let $H\colon \{0,1\}^m \times \{0,1\}^s \to \{0,1\}^s$ be a compression function taking oracle $E \in \mathrm{Block}(k,n)$. The everywhere preimage-finding advantage of adversary $\mathcal{A}$ is defined to be*

$$\mathsf{Adv}_H^{\mathsf{epre}}(\mathcal{A}) = \max_{W \in \{0,1\}^s} \Pr_E\big(M',V') \leftarrow \mathcal{A}^{E,D}(W) :\ W = H^E(M',V')\big] \ .$$

Define $\mathsf{Adv}_H^{\mathsf{epre}}(q)$ as the maximum advantage over all adversaries making at most q queries in total.

The quantity $\mathsf{Adv}_{\mathcal{H}}^{\mathsf{epre}}(q)$ denoting preimage resistance for the iterated hash function $\mathcal{H}^H$ is defined similarly (in this case the advantage of $\mathcal{A}$ is the maximum success probability taken over the choice of possible initial values V_0, which is input to $\mathcal{A}$). Everywhere preimage resistance is preserved in the (MD-)iteration [1], so we get:

Theorem 4. *Let H be a blockcipher based compression function and let $\mathcal{H}$ be the iterated hash function based on H. Then*

$$\mathsf{Adv}_{\mathcal{H}}^{\mathsf{epre}}(q) \leq \mathsf{Adv}_H^{\mathsf{epre}}(q) \leq \mathsf{Adv}_H^{\mathsf{coll}}(q) \ .$$

3 Classical Rate-1 Blockcipher Based Compression Functions

In this section we will deal with classical rate-1 blockcipher based compression functions, where the state size s equals the block length n of the blockcipher and the message size m matches the keysize k of the blockcipher. This includes the famous PGV hash functions [15].

Following in the footsteps of Black et al. [5], we consider Type-I and Type-II compression functions. The former give optimal collision and preimage resistance in the compression function. The second type gives optimal collision resistance in the iteration; its preimage resistance can only be proved up to the birthday bound. One of the important differences with prior art is that we specify in very broad terms the requirements on C^{PRE} and C^{POST}. Essentially our primary concern here is for the proof to go through. In Section 3.3 we will discuss what our classification of Type-I and Type-II implies for the PGV hash functions.

The proof for Type-I schemes is fairly standard and straightforward. However, for the Type-II schemes we deviate from the one by Black et al. [5]. In particular, their proof is based upon colouring a directed graph where the vertices represent queries with all possible answers and arcs are drawn according to whether the input to one query is consistent with the output of the former, given the compression function under consideration. This leads to unwieldy graphs with a complicated notion of what consitutes a collision.

This counterintuitive use of graphs was fixed by Duo and Li [7] (as well as by Lucks [10]), who consider a directed graph where vertices correspond to chaining values and edges are drawn (or coloured) whenever a query has been made that would allow to move from one chaining value to the next. Moreover, for the actual bounding of collision resistance Duo and Li dispense with the direction of the arcs (that thus become edges). Although this seemingly aids the adversary (certain patterns in the graph will be deemed a success even when the underlying event on the hash function is not), this simplification leads to a tighter bound for the Type-II schemes, mainly because there is no longer any need to distinguish between several cases (whose success probability are subsequently added). Our proof (of Theorem 9) closely follows that of Duo and Li.

Note that even for Type-I schemes our bound appears a bit tighter than the one by Black et al., which is due to their simplification based on the the inequality $2(2^n - q) > 2^n$, at least for $q < 2^{n-1}$ (and for larger q most of the bounds become vacuous anyway). We believe the choice between tightness and simplicity in this case is one mainly of taste; we have opted for the former.

3.1 Type-I: Collision Resistant Compression Functions

Definition 5. *A single call blockcipher based compression function H^E is called rate-1 Type-I iff $n = s, k = m$ and the following three hold:*

1. *The preprocessing C^{PRE} is bijective.*
2. *For all M, V the postprocessing $C^{\text{POST}}(M, V, \cdot)$ is bijective.*
3. *For all K, Y the modified postprocessing $C^{\text{AUX}}(K, \cdot, Y)$ is bijective.*

Theorem 6. *Let H^E be a rate-1 Type-I compression function (based on a blockcipher with block size n). Then the advantage of an adversary in finding a collision in H^E after q queries can be upper bounded by*

$$\mathsf{Adv}_H^{\mathsf{coll}}(q) \leq \frac{1}{2}q(q+1)/(2^n - q) .$$

Proof. Let $V_0 \in \{0,1\}^n$ be given. A collision consists of two pairs (M,V) and (M',V') satisfying $H^E(M,V) = \{V_0, H^E(M',V')\}$ yet $(M,V) \neq (M',V')$. We will maintain a list of triples (M,V,W) such that $W = H^E(M,V)$ and the adversary has made the relevant queries to E and/or D. The list is initialized with $(-,-,V_0)$. Since we require the adversary to have made all relevant queries when outputting a collision, we can upper bound the success probability of the adversary by bounding the probability of a collision occuring in this list. We show that any query, be it forward or inverse, will add at most one triple (M,V,W) to this list of computable compression functions, moreover the value W is almost completely out of the adversary's control.

Consider a forward query (K,X). By bijectivity of C^{PRE}, there is a unique pair (M,V) corresponding to this query. Thus, each forward query will add one triple (M,V,W) to the adversary's list of computable values. Since $C^{\text{POST}}(M,V,\cdot)$ is bijective for all M,V, the distribution of compression function output W is closely related to that of blockcipher output Y, which is close to being uniform. More precisely, suppose that so far t queries to E (and D) have been made involving key K, resulting in t plaintext-ciphertext pairs (X_i,Y_i) with $Y_i = E(K,X_i)$ for $i = 1,\ldots,t$. The answer to a fresh query to $E(K,\cdot)$ will therefore be $Y^* \neq Y_i, i = 1,\ldots,t$. Moreover, each of the $2^n - t$ answers is equally likely if E is an ideal cipher. Each possible answer Y^* will combine under C^{POST} with the pair (M,V) consistent with the (K,X) query being made, leading to a possible compression function outcome W^*. Because C^{POST} is bijective when (M,V) are fixed, distinct Y^* lead to distinct W^*, so there are $2^n - t$ possible outcomes W^*, all equally likely.

Similarly, consider an inverse query (K,Y). This yields a unique X and hence by bijectivity of C^{PRE}, there is a unique pair (M,V) corresponding to this query once answered. Thus, each inverse query will add one triple (M,V,W) to the adversary's list of computable values. This time bijectivity of $C^{\text{AUX}}(K,\cdot,Y)$ implies that the distribution of W is closely related to the (almost uniform) output distribution of D. Indeed, suppose that so far t queries to E have been made involving key K, resulting in t plaintext-ciphertext pairs (X_i,Y_i) with $Y_i = E(K,X_i)$ for $i = 1,\ldots,t$. The answer to a fresh query to $D(K,\cdot)$ will therefore be $X^* \neq X_i, i = 1,\ldots,t$. Moreover, each of the $2^n - t$ answers is equally likely if E is an ideal cipher. Each possible answer X^* will combine under $C^{-\text{PRE}}$ and C^{POST} with K and Y to a triple (M,V,W). Because for all K and Y the mapping from X to W is bijective (by assumption on C^{AUX}), distinct X^* lead to distinct W^*, so there are $2^n - t$ possible outcomes W^*, all equally likely.

As a result, after $i-1$ queries the list of computable values contains i triples (M,V,W). The i'th query will add one triple with W uniform over a set of size at least $2^n - i + 1$. Thus the probability that the i'th query causes a collision with any of these triples is at most $i/(2^n-i+1)$. Using a union bound, the probability of a collision after q queries can then be upper bounded by $\sum_{i=1}^{q} i/(2^n-i+1) \leq \frac{1}{2}q(q+1)/(2^n-q)$. $\square$

Theorem 7. *Let H^E be a rate-1 Type-I compression function (based on a a blockcipher with block size n). Then the advantage of an adversary in finding a preimage in H^E after q queries can be upper bounded by*

$$\mathsf{Adv}_H^{\mathsf{epre}}(q) \leq q/(2^n-q) \ .$$

Proof. Let $\mathcal{A}$ be an adversary that tries to find a preimage for its input σ. Assume that $\mathcal{A}$ asks its oracles E and D a total of q queries.

We recall the proof of Theorem 6, where we show that after $i-1$ queries (to E or D) the list of computable values $W = H^E(M,V)$ contains $i-1$ triples (M,V,W). The i'th query will add one triple with W uniform over a set of size at least 2^n-i+1. Thus the probability that the i'th query hits σ is at most $1/(2^n-i+1)$. Using a union bound, the probability of finding a preimage for σ after q queries can then be upper bounded by $\sum_{i=1}^{q} 1/(2^n-i+1) \leq q/(2^n-q)$. $\square$

3.2 Type-II: Collision Resistance in the Iteration

Definition 8. *A single call blockcipher based compression function H^E is called rate-1 Type-II iff $n = s, k = m$, and the following three hold:*

1. *The preprocessing C^{PRE} is bijective.*
2. *For all M, V the postprocessing $C^{\text{POST}}(M,V,\cdot)$ is bijective.*
3. *For all K, $C^{-\text{PRE}}(K,\cdot)$ restricted to V, its second output, is bijective.*

Theorem 9. *Let H^E be a rate-1 Type-II compression function. If E is an ideal cipher with block size n, then the advantage of an adversary in finding a collision in the iterated hash function $\mathcal{H}^H$ after q queries is upper bounded by*

$$\mathsf{Adv}_{\mathcal{H}}^{\mathsf{coll}}(q) \leq \frac{1}{2}q(q+1)/(2^n-q) \ .$$

Proof. Let $V_0 \in \{0,1\}^n$ be $\mathcal{H}$'s initial vector.

We define an undirected graph $G = (V_G, E_G)$ with vertex set $V_G = \{0,1\}^n$—corresponding to all 2^n possible chaining values—and initially an empty edge set $E_G = \emptyset$. We will dynamically add edges based on the queries to E and D. In particular, we add an edge (V,W), labelled by M, if we know a message M such that $W = H^E(M,V)$ (or $V = H^E(M,V)$) and the relevant query to either E or D has been made. We claim that to find a collision would require constructing a ρ-shape containing the initial vector V_0. Suppose that $\mathcal{H}(\mathbf{M}) = \mathcal{H}(\mathbf{M}')$ with $\mathbf{M} \neq \mathbf{M}'$. Write $\mathbf{M} = (M_1,\ldots,M_\ell)$ and $\mathbf{M}' = (M'_1,\ldots,M'_{\ell'})$ and correspondingly $V_0,\ldots,V_\ell$ respectively $V'_0,\ldots,V'_{\ell'}$ for the chaining values of the iterated hash. Note that $V_0 = V'_0$ and $V_\ell = V'_{\ell'}$. Assume $\ell \leq \ell'$. Because $\mathbf{M} \neq \mathbf{M}'$, there exists a t such that $M_i = M'_i$ for all $0 \leq i < t$ but $M_t \neq M'_t$ (or possibly

$\ell < t \leq \ell'$). As a result, the paths $(V_0, \ldots, V_t)$ and $(V'_0, \ldots, V'_t)$ are identical, but the edges (V_t, V_{t+1}) and (V'_t, V'_{t+1}) are distinct, even when V'_{t+1} happens to equal V_{t+1} (in particular, the edges are labelled differently). Since $V_\ell = V'_{\ell'}$ at some point the paths need to come together again, completing the ρ-shape. Note that due to our use of an undirected graph not every ρ-shape will lead to a collision though.

Since we are dynamically adding edges to the graph, components in the graph will also grow dynamically. Let T be the set of all nodes that are in a component containing a cycle or the initial vector V_0. The first claim is that after i queries, the set T has cardinality at most $i+1$. Indeed, the component containing V_0 has at most $i'+1$ nodes when i' edges are used. A cyclic component based on i' edges has at most i' nodes. Thus the initial vector component is the only component in T that causes the number of nodes larger than the number of edges, by at most one. Bijectivity of C^{PRE} implies that a query (either forward or inverse) will add at most one edge to the graph, so after i queries, there are at most i edges in the entire graph and at most $i+1$ nodes in T.

The second claim is that to complete a ρ-shape, either a cycle has to be completed within the V_0-component, or the V_0-component needs to be connected with a cycle. Either way, an edge has to be found of which both nodes are already part of T. The probability that on the i'th query a collision is found by a forward query is at most $i/(2^n - i)$: bijectivity of $C^{\text{POST}}(M, V, \cdot)$ ensures that W is uniformly distributed over a set of size at least $2^n - i$, so hitting a set of size i occurs at most with said probability. Similarly, for an inverse query the probability of finding a collision on the i'th query using an inverse query is at most $i/(2^n - i)$: this time bijectivity of $C^{\text{AUX}}(K, \cdot, Y)$ ensures that V is uniformly distributed over a set of size at least $2^n - i$.

We can now wrap up and conclude that the probability of finding a collision on the i'th query is at most $i/(2^n - i)$ and the probability after q queries is at most $\sum_{i=1}^{q} i/(2^n - i) \leq \frac{1}{2}q(q+1)/(2^n - q)$. □

3.3 Implications to the PGV Schemes

In this section we investigate how the 64 PGV schemes [15] fit in the general Type-I and Type-II framework. Recall that for the PGV-style schemes the blockcipher has key size equal to the block length; the compression function will look like $H^E(M, V) = E(K, X) \oplus U$ where $K, X, U \in \{C, M, V, M \oplus V\}$ and C is some fixed, publicly known bitstring. These restrictions can also be expressed in terms of C^{PRE} and C^{POST}. Our results are in line with the classification of Black et al. [5] and the tighter bounds by Duo and Li [7].

Let us first set up some notation. As is customary [8] for schemes with linear processing C^{PRE} and C^{POST}, we will represent the linear PGV schemes using matrices. We will use $\mathbb{Z}_2^2$ to express the way K,X, and U are functions of M and V: a vector $\mathbf{X} \in \mathbb{Z}_2^2$ corresponds to $X = \mathbf{X} \cdot \binom{M}{V}$, making a distinction between the linear map $\mathbf{X} \in \mathbb{Z}_2^2$ and the value $X \in \{0,1\}^n$. We will also write $\mathbf{X} = (X_M, X_V)$. We can safely ignore any affine part, so $\mathbf{U} = (00)$ can be thought of to correspond to the aforementioned $U \leftarrow C$. (This is without loss of

Table 1. The 20 Secure PGV-style schemes, writing $E_K(X)$ for $E(K,X)$ and W for $M \oplus V$. Superscripted are the $\imath$-indices from [5, Fig. 1 and 2].

$\binom{\mathbf{k}}{\mathbf{x}} \backslash \mathbf{s}$	(00)	(01)	(10)	(11)
$\begin{pmatrix}0&1\\1&0\end{pmatrix}$	insecure	insecure	$E_V(M) \oplus M^1$	$E_V(M) \oplus W^3$
$\begin{pmatrix}0&1\\1&1\end{pmatrix}$	insecure	insecure	$E_V(W) \oplus M^4$	$E_V(W) \oplus W^2$
$\begin{pmatrix}1&0\\0&1\end{pmatrix}$	$E_M(V)^{15}$	$E_M(V) \oplus V^5$	$E_M(V) \oplus M^{17}$	$E_M(V) \oplus W^7$
$\begin{pmatrix}1&0\\1&1\end{pmatrix}$	$E_M(W)^{19}$	$E_M(W) \oplus V^8$	$E_M(W) \oplus M^{20}$	$E_M(W) \oplus W^6$
$\begin{pmatrix}1&1\\0&1\end{pmatrix}$	$E_W(V)^{16}$	$E_W(V) \oplus V^{10}$	$E_W(V) \oplus M^{12}$	$E_W(V) \oplus W^{18}$
$\begin{pmatrix}1&1\\1&0\end{pmatrix}$	$E_W(M)^{13}$	$E_W(M) \oplus V^{11}$	$E_W(M) \oplus M^9$	$E_W(M) \oplus W^{14}$

generality, since translation by a constant will not affect bijectivity in either of the criteria used in Definitions 5 and 8.) Since there are 4 elements in $\mathbb{Z}_2^2$ and we have to pick 3 ($\mathbf{K}$,$\mathbf{X}$, and $\mathbf{U}$), there are 64 constructions to consider in total, corresponding to the 64 PGV schemes.

We are now ready to see what the requirements from Definitions 5 and 8 mean in terms of the vectors $\mathbf{K}, \mathbf{X}$ and $\mathbf{U}$ and hence for the classification and security of the PGV schemes. The 20 interesting schemes are listed in Table 1, where we have also included the $\imath$-indices assigned to these schemes by Black et al. [5]. When we write $H_\imath$ resp. $\mathcal{H}_\imath$ for $\imath \in \{1,\ldots,20\}$ we refer to this enumeration. Proofs are to be found in the full version [19].

Lemma 10. *A PGV scheme is Type-I iff* $\binom{\mathbf{K}}{\mathbf{X}}$ *and* $\binom{\mathbf{K}}{\mathbf{U}}$ *are both invertible matrices. In particular,* $H_{1..12}$ *are Type-I schemes.*

The requirements for the Type-II schemes turn out surprisingly simple: indeed apart from the preprocessing having full rank, the only requirement is that the key depends on the message. Consequently we end up with 16 Type-II schemes as opposed to only 8 given by Black et al. The 'additional' 8 schemes we identify are also Type-I, which explains why previously they were not classified as Type-II. Our results therefore suggest a subdivision of the PGV Type-I schemes, namely those that are also Type-II (being those with a key depending on the message) and those that are just Type-I (those whose key equals the chaining variable). The same subdivision was made by Duo and Li [7] in the context of second preimage resistance.

Lemma 11. *A PGV scheme is Type-II iff* $\binom{\mathbf{K}}{\mathbf{X}}$ *is an invertible matrix with* $K_M = 1$*. In particular,* $\mathcal{H}_{5..20}$ *are Type-II schemes.*

Combining Lemmas 10 and 11 with Theorems 6, 7, and 9 then yields Corollary 12 below. For completeness [5, 15], it is known that the given upper bounds on the advantages are tight up to a small constant factor. Moreover, for $\mathcal{H}_{13..20}$ preimage resistance is worse than desired, namely $\mathsf{Adv}^{\mathsf{epre}}_{\mathcal{H}}(q) = \Theta(q^2/2^n)$ (due to a meet-in-the-middle attack). The remaining 44 PGV schemes do not offer any collision resistance in the iteration.

Corollary 12. *(Security of the PGV schemes) For $H_{1..12}$ it holds that $\mathsf{Adv}^{\mathsf{coll}}_{H}(q) \leq \frac{1}{2}q(q+1)/(2^n - q)$, and $\mathsf{Adv}^{\mathsf{epre}}_{H}(q) \leq q/(2^n - q)$; for $\mathcal{H}_{13..20}$ it holds that $\mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}}(q) \leq \frac{1}{2}q(q+1)/(2^n - q)$.*

4 Generalized Single Call Compression Functions

In the previous section we discussed the standard (single call) case where the input and output sizes of the compression function neatly matched those of the underlying blockcipher, in particular $m = k$ and $s = n$. In this section we let go of these restrictions and consider three more general scenarios.

First we will consider what could be called chopping the output of the compression (or really the scenario where $s < n$). For instance, the Davies-Meyer construction is optimally collision and preimage resistant, but what happens if you chop the output: is the security still optimal given the new output length (it is). A welcome benefit of chopping the output is that it frees up bits for the message. More precisely, if $s < n$ then we can have a larger m while maintaining $m + s = n + k$. In particular, compression becomes feasible even for fixed permutations (corresponding to $k = 0$). In view of the recent availability of huge size permutations constructions with $s < n$ gain traction; an example is Grindahl[9]. We will refer to this scenario as compression in the postprocessing, the corresponding H^E's are called chopped compression functions.

Similarly, one might also try to improve efficiency by squeezing in more bits of input in the compression function than can be input to the primitive (this corresponds to $m + s > n + k$). We call this compression in the preprocessing and speak of overloaded compression functions. Like the previous scenario, this opens up the possibility of achieving compression based on a single fixed permutation. We suggest a general Type-I compression function and give a bound on its collision resistance and preimage resistance. Security in the iteration is more complicated here: we discuss related work and point out some challenging open problems.

Finally we deal with the problem of getting security beyond the block length of the blockcipher, that is $s > n$. Here we say that expansion in the postprocessing gives rise to supercharged compression functions. Promising results were previously given by Lucks [10] in the iteration and Stam [18] for a compression function. We develop a general theory and give two concrete examples based on the latter work.

(Any missing proofs, as well as an expanded treatment of supercharged compression functions, can be found in the full version [19].)

4.1 Chopping: Compression in the Postprocessing

Let us consider an $m+s$-to-s bit compression function based on a single call to a blockcipher with key size k and block size n. In this section we will assume that $m+s=n+k$ and $s<n$. What can we say of the collision and preimage resistance of the compression function resp. iterated hash function, under which conditions will we achieve optimal security?

If we go through the criteria from the previous section, it is clear we can no longer satisfy them all. More to the point, whereas the first condition (bijectivity of the preprocessing) still applies, the postprocessing now becomes a mapping from n to s bits, which cannot be bijective since $s<n$. The natural generalization is to replace being a bijection with being balanced: all elements in the codomain should have the same number of preimages, namely 2^{n-s}. It turns out that this fairly simple modification works quite well. Again we have two types: the first one giving optimal collision and preimage resistance for the compression function; the second one giving optimal collision resistance in the iteration only (and guaranteed preimage resistance only up to the collision resistance).

Definition 13. *A single call blockcipher based compression function H^E is called chopped single call Type-I iff $s<n, m+k=n+s$, and the following three hold:*

1. *The preprocessing C^{PRE} is bijective.*
2. *For all M,V the postprocessing $C^{\text{POST}}(M,V,\cdot)$ is balanced.*
3. *For all K,Y the modified postprocessing $C^{\text{AUX}}(K,\cdot,Y)$ is balanced.*

Definition 14. *A single call blockcipher based compression function H^E is called chopped single call Type-II iff $s<n, m+k=n+s$ and the following three hold:*

1. *The preprocessing C^{PRE} is bijective.*
2. *For all M,V the postprocessing $C^{\text{POST}}(M,V,\cdot)$ is balanced.*
3. *For all K the inverse preprocessing $C^{-\text{PRE}}(K,\cdot)$ when restricted to its V output is balanced.*

Theorem 15. *Let H^E be a chopped single call Type-I compression function. Then the advantage of an adversary in finding a collision, resp. a preimage in H^E after q queries can be upper bounded by*

$$\mathsf{Adv}_H^{\mathsf{coll}}(q) \le q(q+1)/2^s, \qquad \mathsf{Adv}_H^{\mathsf{epre}}(q) \le q/2^{s-1} \ .$$

Theorem 16. *Let H^E be a chopped single call Type-II compression function. Then the advantage of an adversary in finding a collision in the iterated hash function $\mathcal{H}^H$ after q queries is upper bounded by*

$$\mathsf{Adv}_{\mathcal{H}}^{\mathsf{coll}}(q) \le q(q+1)/2^s \ .$$

4.2 Overloading: Compression in the Preprocessing

Another way to improve efficiency it to keep $s = n$, but allow $m > k$. In this case bijectivity of the preprocessing can no longer be satisfied, which has ramifications throughout.

Firstly, for a given pair (K, X) it is now the case that $C^{-\text{PRE}}$ yields a set of 2^{m-k} pairs (M, V). Consequently, the modified postprocessing $C^{\text{AUX}}(K, \cdot, Y)$ becomes a function from n-bits to subsets of size (up to) 2^{m-k} of $\{0,1\}^n$. Our requirement on this new type of C^{AUX} is a natural generalization of balancedness.

Secondly, although the condition that $C^{\text{POST}}(M, V, \cdot)$ is bijective is still well-defined, it is no longer sufficient. For instance, if $C^{\text{PRE}}(M, V) = C^{\text{PRE}}(M', V')$ for certain values of $(M, V) \neq (M', V')$ and the bijections $C^{\text{POST}}(M, V, \cdot)$ and $C^{\text{POST}}(M', V', \cdot)$ are identical, then collisions can very easily be found. To avoid this problem we explicitly rule out collisions in the output whenever (M, V) and (M', V') already collide during preprocessing (in C^{PRE}).

Definition 17. *A single call blockcipher based compression function H^E is called overloaded single call Type-I iff $s = n, m \geq k$, and the following four hold:*

1. *The preprocessing C^{PRE} is balanced.*
2. *For all $(M, V) \neq (M', V')$ with $C^{\text{PRE}}(M, V) = C^{\text{PRE}}(M', V')$ and all Y it holds that $C^{\text{POST}}(M, V, Y) \neq C^{\text{POST}}(M', V', Y)$.*
3. *For all M, V the postprocessing $C^{\text{POST}}(M, V, \cdot)$ is bijective.*
4. *For all K, Y the modified postprocessing $C^{\text{AUX}}(K, \cdot, Y)$ is balanced in the sense that for all V the number of X such that $V \in C^{\text{AUX}}(K, X, Y)$ equals 2^{m-k}.*

Theorem 18. *Let H^E be an overloaded single call Type-I compression function. Then the advantage of an adversary in finding a collision, resp. a preimage in H^E after q queries can be upper bounded by*

$$\mathsf{Adv}_H^{\mathsf{coll}}(q) \leq q(q+1)/2^{2k+n-2m}, \qquad \mathsf{Adv}_H^{\mathsf{epre}}(q) \leq q/2^{n+k-m-1} .$$

Theorem 18 can be reinterpreted by saying that to find collisions roughly $2^{n/2+k-m}$ queries are required; to find preimages roughly 2^{n+k-m} queries should suffice. It is interesting to compare the collision resistance thus achieved with recently conjectured optimal bounds [17, 18]. A straightforward generalization of Rogaway and Steinberger's result [17] suggests the best we can achieve is collision resistance up to $2^{n/2+k-m}$ queries, neatly corresponding to our construction. However, Stam [18] conjectures collision resistance is feasible up to $2^{(n+k-m)/2}$ queries, based on an ideal state size s of $n + k - m$ bits. Using this state size actually brings us back exactly to compression in the postprocessing as discussed in the previous section: by reducing s we can increase m while maintaining $n + k = m + s$ and Theorem 15 essentially guarantees collision resistance up to $2^{(n+k-m)/2}$ queries. So here is another scenario where reducing the state size mysteriously seems to boost collision resistance.

But all is not as it seems. An example overloaded single call Type-I compression function is Davies-Meyer with the $m - k$ superfluous message bits xored directly into the output. It is not hard to show that in this case the collision finding advantage is much smaller than Theorem 18 makes believe:

$$\mathsf{Adv}_H^{\mathsf{coll}}(q) \le q(q+1)/2^{k+n-m} .$$

Iterated Case. For rate-1 and chopped compression functions, looking at the iteration gave rise to a second class of schemes that had the same collision resistance in the iteration as the main schemes, but inferior preimage resistance. For overloaded compression functions, we do not give a classificiation of Type-II schemes (also in light of our Type-I bounds' lack of tightness). However, we do point out that some non-trivial results in this setting were previously achieved for sponge functions [4], whose collision resistance (in the iteration) holds roughly up to $2^{(n-m)/2}$ queries ($k = 0$). This matches the collision resistant compression function of the previous paragraph.

However, recent developments indicate that iteration might boost collision resistance even further. In particular, the sponge construction has rate $\alpha = m/(n-m)$ achieving collision resistance up to roughly $2^{n(1-\alpha)/2}$ queries. Rogaway and Steinberger [17] have shown that for any rate-α construction after $1.9n2^{n(1-\alpha)}$ queries collisions are guaranteed. This still leaves a considerable gap.

4.3 Supercharging: Expansion in the Postprocessing

Whereas for chopped and overloaded compression functions we sacrificed security for the sake of efficiency, in this section we will attempt the exact opposite: sacrificing efficiency for the sake of security. We do this by extending the state size, so $s > n$. Not to complicate things further, we will assume that $m+s = n+k$ (and let C^{PRE} be bijective). For any fixed pair (M, V) we have that C^{POST} maps $\{0,1\}^n$ to $\{0,1\}^s$. Since $n < s$ this cannot be a bijection, but at best an injection (similar for C^{AUX}). If all these injections have exactly the same range, we are not using our codomain of 2^s values to the full; indeed we might have well been padding the state with a constant. This leads us to the following formalization.

Definition 19. *A single call blockcipher based compression function H^E is called supercharged single call Type-I with overlap γ iff $s \ge n, m + s = n + k$ and the following three hold:*

1. *The preprocessing C^{PRE} is bijective.*
2. *For all M, V the postprocessing $C^{\text{POST}}(M, V, \cdot)$ is injective, with effective range $R_{\text{POST},(M,V)}$.*
3. *For all K, Y the modified postprocessing $C^{\text{AUX}}(K, \cdot, Y)$ is injective, with effective range $R_{\text{AUX},(K,Y)}$.*

Where the overlap γ is defined as:

$$\gamma = \max\left\{ |R_Z \cap R_{Z'}| : Z, Z' \in \{\text{POST}, \text{AUX}\} \times \{0,1\}^{k+n}, Z \ne Z' \right\} .$$

Theorem 20. *Let H^E be a supercharged single call Type-I compression function with overlap γ. Then the advantage of an adversary in finding a collision after $q \leq 2^{n-1}$ queries can be upper bounded by*

$$\mathsf{Adv}_H^{\mathsf{coll}}(q) \leq q\kappa/2^{n-1} + 2^{m+s+1}\left(\frac{e\gamma q}{(\kappa-1)2^{n-1}}\right)^{\kappa-1}$$

for arbitrary positive integer $\kappa > q\gamma/2^{n-1}$.

Corollary 21. *Let H^E be a supercharged single call Type-I compression function with overlap γ. Then for $q < 2^{n-1}/\gamma^{\frac{1}{2}}$ the probability of finding a collision can be upper bounded by*

$$\mathsf{Adv}_H^{\mathsf{coll}}(q) \leq 2\max(2e\gamma^{\frac{1}{2}}, m+n+s+2)q/2^n \ .$$

In practice this means that we get good security up to q of order $2^n/\gamma^{\frac{1}{2}}$. Stam [18] suggests that finding collisions can be expected after $2^{(n+k-m)/2}$ queries. Since $n+k = m+s$ this neatly corresponds to $2^{s/2}$, in other words optimal collision resistant compression functions of this type might actually exist. Note that the rate is lower than before, arguably m/n. As we show in Lemma 22, the best we can hope for is γ of order 2^{2n-s}, giving collision resistance up to $2^{s/2}$ queries. Whether for all relevant settings of n, s, k, and m there exists a postprocessing C^{POST} with overlap γ close to 2^{2n-s} is an open problem. Below we give two examples where it does though, based on an earlier construction [18].

Lemma 22. *Let H^E be a supercharged single call Type-I compression function then overlap*

$$\gamma \geq \frac{2(2^{2n+m}-2^n)}{2^{s+m}-1} \quad (\approx 2^{2n-s+1}) \ .$$

Example I: A Double-Length Construction. We recall the construction [18] for a double length compression function based on a single ideal $3n$-to-n compression function F. Split the $2n$-bit state V in two equally sized parts V_1 and V_2. Then given an n-bit message block M, compression proceeds as follows:

1. Compute $Y \leftarrow F(M, V_1, V_2)$.
2. Output $(W_1, W_2) \leftarrow (Y, V_2Y^2 + V_1Y + M)$.

where the polynomial evaluation is over $\mathbb{F}_{2^n}$. Originally only a proof of collision resistance against non-adaptive adversaries was given, based on random functions instead of random permutations (so in particular an adversary would not have access to an inversion oracle). We would like to port the scheme to the ideal cipher model, based on a blockcipher with $k = 2n$.

1. Set $K \leftarrow (V_1, V_2)$ and $X \leftarrow M$.
2. Compute $Y \leftarrow E(K, X)$.
3. Compute $W_1 \leftarrow Y + M$ and $W_2 \leftarrow MW_1^2 + V_1W_1 + V_2$; output (W_1, W_2).

Lemma 23. *For the compression function above, $\gamma = 3$.*

Proof. To determine the overlap γ it helps to first write down the effective ranges $R_{\text{POST},(M,V)}$ and $R_{\text{AUX},(K,Y)}$ explicitly. It is easy to see that

$$R_{\text{POST},(M,V_1,V_2)} = \left\{(W, MW^2 + V_1W + V_2) | W \in \{0,1\}^n\right\}$$

and with a little bit more effort, using that $M = Y + W$ and $(K_1, K_2) = (V_1, V_2)$,

$$R_{\text{AUX},(K_1,K_2,Y)} = \left\{(W, W^3 + YW^2 + K_1W + K_2) | W \in \{0,1\}^n\right\} .$$

As a result, for (W_1, W_2) to be in the intersection of R_Z and $R_{Z'}$, we require W_1 to be a root of the difference of the two polynomials that define W_2 for Z resp. Z'. It can be readily verified that $Z \neq Z'$ implies the relevant two polynomials are distinct as well, and the resulting difference is a non-zero polynomial of degree at most three. It will therefore have at most three roots over $\mathbb{F}_{2^n}$. $\square$

Corollary 24. *For the compression function above, for $q \leq 2^{n-\frac{3}{2}}$:*

$$\mathsf{Adv}_H^{\mathsf{coll}}(q) \leq (n + \frac{1}{2})q/2^{n-3} .$$

Curiously, if we would change the computation of W_2 even slightly, for instance $W_2 \leftarrow V_2W_1^2 + V_1W_1 + M$, the impact on the overlap γ is dramatic. Suddenly $R_{\text{AUX},(K_1,K_2,Y)} = \{W, K_2W^2 + (K_1 + 1)W + Y | W \in \{0,1\}^n\}$ and consequently $R_{\text{AUX},(V_1+1,V_2,M)} = R_{\text{POST},(V_1,V_2,M)}$, so that $\gamma = 2^n$. As a result, Theorem 20 can only be used to guarantee collision resistance up to roughly $2^{n/2}$ queries.

We note that like the original [18], our double length construction has some obvious shortcomings (see the full version [19] for more details).

Example II: An Intermediate Construction. We conclude with a construction based on a $3n/2$ bit state (split into three parts of $n/2$ bits each), that compresses $n/2$ message bits.

1. $X \leftarrow (M, V_1), K \leftarrow (V_2, V_3)$;
2. $Y \leftarrow E(K, X)$;
3. $W_1 \leftarrow Y_1 + M, W_2 \leftarrow Y_2 + V_1$, and $W_3 \leftarrow MW_1^3 + V_1W_1^2 + V_2W_1 + V_3$.

Lemma 25. *For the compression function above, $\gamma = 2^{2+n/2}$.*

Corollary 26. *For the compression function above and all $q < 2^{3n/4-2}$*

$$\mathsf{Adv}_H^{\mathsf{coll}}(q) \leq eq/2^{3n/4-3} .$$

Acknowledgements

The author would like to thank John Black, Phil Rogaway and Onur Özen for useful ideas on the presentation; Elena Andreeva, Lars Knudsen and the anonymous FSE'09 referees for pointing out certain problems with preimage resistance; and Tom Shrimpton for great advice for the duration of the project.

References

1. Andreeva, E., Neven, G., Preneel, B., Shrimpton, T.: Seven-property-preserving iterated hashing: Rox. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 130–146. Springer, Heidelberg (2007)
2. Bernstein, D.J.: Cubehash specification (2.b.1). Submission to NIST (2008)
3. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sponge functions. In: Ecrypt Hash Workshop (2007)
4. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the indifferentiability of the sponge construction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 181–197. Springer, Heidelberg (2008)
5. Black, J., Rogaway, P., Shrimpton, T.: Black-box analysis of the block-cipher-based hash-function constructions from PGV. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 320–335. Springer, Heidelberg (2002)
6. Damgård, I.: A design principle for hash functions. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 416–427. Springer, Heidelberg (1990)
7. Duo, L., Li, C.: Improved collision and preimage resistance bounds on PGV schemes. IACR ePrint Archive, Report 2006/462 (2006), `http://eprint.iacr.org/2006/462`
8. Hohl, W., Lai, X., Meier, T., Waldvogel, C.: Security of iterated hash functions based on block ciphers. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 379–390. Springer, Heidelberg (1993)
9. Knudsen, L.R., Rechberger, C., Thomsen, S.S.: The Grindahl hash functions. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 39–57. Springer, Heidelberg (2007)
10. Lucks, S.: A collision-resistant rate-1 double-block-length hash function. In: Symmetric Cryptography. Dagstuhl Seminar Proceedings, IBFI, Number 07021 (2007)
11. Matyas, S., Meyer, C., Oseas, J.: Generating strong one-way functions with cryptographic algorithms. IBM Technical Disclosure Bulletin 27(10a), 5658–5659 (1985)
12. Menezes, A., van Oorschot, P., Vanstone, S.: CRC-Handbook of Applied Cryptography. CRC Press, Boca Raton (1996)
13. Merkle, R.C.: One way hash functions and DES. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 428–446. Springer, Heidelberg (1990)
14. Miyaguchi, S., Iwata, M., Ohta, K.: New 128-bit hash function. In: Proceedings 4th International Joint Workshop on Computer Communications, pp. 279–288 (1989)
15. Preneel, B., Govaerts, R., Vandewalle, J.: Hash functions based on block ciphers: A synthetic approach. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 368–378. Springer, Heidelberg (1993)
16. Rogaway, P., Shrimpton, T.: Cryptographic hash-function basics: Definitions, implications and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 371–388. Springer, Heidelberg (2004)
17. Rogaway, P., Steinberger, J.: Security/efficiency tradeoffs for permutation-based hashing. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 220–236. Springer, Heidelberg (2008)
18. Stam, M.: Beyond uniformity: Better security/efficiency tradeoffs for compression functions. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 397–412. Springer, Heidelberg (2008)
19. Stam, M.: Blockcipher based hashing revisited. IACR ePrint Archive, Report 2008/071 (2008), `http://eprint.iacr.org/2008/071`