J. Cryptol. (2010) 23: 519–545
DOI: 10.1007/s00145-010-9071-0

Journal of
CRYPTOLOGY

An Analysis of the Blockcipher-Based Hash Functions from PGV

J. Black
Department of Computer Science, University of Colorado, Boulder, CO 80309, USA
jrblack@cs.colorado.edu

P. Rogaway
Department of Computer Science, University of California, Davis, CA 95616, USA
rogaway@cs.ucdavis.edu

T. Shrimpton
Department of Computer Science, Portland State University, Portland, OR 97201, USA
teshrim@ucdavis.edu

M. Stam
LACAL, School of Computer and Communication Sciences, EPFL, Station 14, Lausanne 1015, Switzerland
martijn.stam@epfl.ch

Received 3 January 2005
Online publication 16 July 2010

Abstract. Preneel, Govaerts, and Vandewalle (1993) considered the 64 most basic ways to construct a hash function $H\colon \{0,1\}^* \to \{0,1\}^n$ from a blockcipher $E\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. They regarded 12 of these 64 schemes as secure, though no proofs or formal claims were given. Here we provide a proof-based treatment of the PGV schemes. We show that, in the ideal-cipher model, the 12 schemes considered secure by PGV really *are* secure: we give tight upper and lower bounds on their collision resistance. Furthermore, by stepping outside of the Merkle–Damgård approach to analysis, we show that an additional 8 of the PGV schemes are just as collision resistant (up to a constant). Nonetheless, we are able to differentiate among the 20 collision-resistant schemes by considering their preimage resistance: only the 12 initial schemes enjoy optimal preimage resistance. Our work demonstrates that proving ideal-cipher-model bounds is a feasible and useful step for understanding the security of blockcipher-based hash-function constructions.

Key words. Blockcipher, Collision-resistant hash function, Cryptographic hash function, Ideal-cipher model, Modes of operation.

1. Introduction

Background The idea of building a cryptographic hash function from a blockcipher goes back more than 30 years [31], when Michael Rabin suggested to hash a message $M = m_1 m_2 \cdots m_\ell$ by setting $H(M) = \mathrm{DES}_{m_\ell}(\mathrm{DES}_{m_{\ell-1}}(\cdots(\mathrm{DES}_{m_1}(h_0))\cdots))$ for

some constant h_0. This can be viewed as the Merkle–Damgård construction [10,27] with compression function $f(h_{i-1}, m_i) = \mathrm{DES}_{m_i}(h_{i-1})$. Other blockcipher-based constructions soon followed, like those of Davies–Meyer [26], Matyas–Meyer–Oseas [24], and Miyaguchi–Preneel [29]. When dedicated hash functions began to emerge and become popular, beginning with MD4 [32], these too were built from (often implicit) blockciphers.

Generalizing from many blockcipher-based hash-function constructions, Preneel, Govaerts, and Vandewalle [30], henceforth "PGV", systematically considered ways to turn a blockcipher $E\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ into a hash function $H\colon (\{0,1\}^n)^* \to \{0,1\}^n$. The authors assume that E is first used to define a compression function $f\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ and that the hash function is then given by iterating f with Merkle–Damgård chaining, meaning:

function $H(m_1 \cdots m_\ell)$
for $i \leftarrow 1$ **to** ℓ **do** $h_i \leftarrow f(h_{i-1}, m_i)$
return h_ℓ

Here the initial chaining value h_0 is a fixed n-bit constant, and the message $M = m_1 \cdots m_\ell$, where $|m_i| = n$, is assumed to be a multiple of n bits. To make the compression function $f(h, m)$ from the blockcipher E, the PGV paper considers all functions

$$f(h, m) = E_k(x) \oplus s \quad \text{where } k, x, s \in \{c,\ h,\ m,\ h \oplus m\}.$$

Here c is a fixed but arbitrary n-bit constant. Of the 64 such schemes, PGV regard 12 as secure. Another 13 schemes they classify as *backward-attackable*, which means they are subject to an identified (but not very severe) potential attack. The remaining 39 schemes are subject to damaging attacks identified by PGV and others.

PGV focused on attacks, not on proofs. All the same, it seems to have been a commonly held belief that it should be possible to produce proofs for the schemes that PGV regarded as secure. Indeed PGV go so far as to say that *For each of these schemes, it is possible to write a 'security proof' based on a black box model of the encryption algorithm, as was done for the Davies–Meyer scheme in* [43]. The latter paper shows that the scheme we will later call H_5 is preimage resistant in the following (rather weak) sense. Consider an algorithm A with E and E^{-1} oracles, the first realizing a family of uniform independent permutations and the second being the inverse of the first. Assume that A, given $y \in \{0,1\}^n$, always outputs a preimage x for it under H_5^E. Then the expected number of queries A must make is at least 2^{n-1}.

The model above is called the *ideal-cipher model* (more on it shortly). It is the same model later used by Merkle in his own investigation of the security of hash functions [27]. Still, prior to the proceedings version of the current paper [6], there had been no detailed analysis in the literature for the collision resistance or preimage resistance of *any* blockcipher-based hash function—neither in the ideal-cipher model nor any other.

Our Results This paper takes a proof-centric look at the construction of blockcipher-based hash functions. We again consider the 64 hash functions of PGV, but we do so from the point of view of security upper and lower bounds—proofs and attacks—in a

PGV category	our category	collisions	preimages
✓or FP (12 schemes)	group-1: $H_1, \ldots, H_{12}$ (12 schemes)	$\Theta(q^2/2^n)$	$\Theta(q/2^n)$
B (13 schemes)	group-2: $H_{13}, \ldots, H_{20}$ (8 schemes)	$\Theta(q^2/2^n)$	$\Theta(q^2/2^n)$
F, P, or D (39 schemes)	group-3: $H_{21}, \ldots, H_{64}$ (44 schemes)	$\Theta(1)$	—

Fig. 1. Summary of our results. In the ideal-cipher model, the 64 schemes analyzed by PGV fall into three natural categories, based on their collision resistance and their preimage resistance. The rightmost two columns entail both upper and lower bounds on the schemes' security.

formally specified model. What falls out is a taxonomy somewhat different from that provided by PGV. Twelve of the PGV hash functions, the *group-1* schemes $H_1, \ldots, H_{12}$, are shown to have essentially optimal collision resistance and optimal preimage resistance (optimal, up to a constant, with respect to just these two properties). Eight more PGV hash functions, the *group-2* schemes $H_{13}, \ldots, H_{20}$, are shown to have optimal collision-resistance but highly suboptimal preimage resistance. The remaining $44 = 64 - 12 - 8$ PGV hash functions have terrible collision resistance and are therefore lumped together into a third, *group-3* set of functions. PGV had already provided constant-query collision-finding attacks on 39 of these 44 functions; in Sect. 9 we provide constant-query collision-finding attacks on the five remaining group-3 schemes. Our taxonomy is summarized in Fig. 1, where it is also compared to the taxonomy of PGV. The precise model, including the meaning of q, will momentarily be described.

The definition of the 20 group-1 and group-2 schemes is given in Fig. 2, which also contains a forward pointer to our main results. No attempt has been made to optimize the constants, though none are excessively large. Figures 3 and 4 provide a pictorial view of the compression functions of $H_1, \ldots, H_{20}$. We comment that several of the PGV hash functions have well-known inventors or names: H_1 (Matyas–Meyer–Oseas), H_2 (Miyaguchi–Preneel), H_5 (Davies–Meyer), H_{13} (Rabin), and H_{17} (Bitzer); see [26,30].

The Ideal-Cipher Model Our results are in the model dating to Shannon [36] and used for works like [14,20,43]. An adversary A is given access to oracles E and E^{-1}, where E is a random blockcipher $E\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$, and E^{-1} is its inverse (we are restricting our attention to blockciphers with both key-length and block-length equal to some integer $n \geq 1$). Thus each key $k \in \{0,1\}^n$ names a uniformly selected permutation $E_k = E(k, \cdot)$ on $\{0,1\}^n$, and the adversary is given oracles for E and E^{-1}. The latter, on input (k, y), returns the point x such that $E_k(x) = y$.

For a hash function H that depends on E, the adversary's job in attacking the collision resistance of H is to find distinct M, M' such that $H(M) = H(M')$. One measures the optimal adversary's chance of doing this as a function of the number of E or E^{-1} queries it makes. Similarly, the adversary's job in inverting H is to find an inverse under H for a random range point $Y \in \{0,1\}^n$. (See Sect. 10 for a justification of this definition.) Again one measures the optimal adversary's chance of doing this as a function of the total number of E or E^{-1} queries it makes. This model has variously been called the *Shannon model* or the *ideal-cipher model.*

s	r	$f(h,m)$	$f'(h,m)$	$f''(h,m,y)$	$f^*(k,x,y)$	coll atk	coll sec	pre atk	pre sec
1	3	$E_h(m) \oplus m$	(h,m)	$m \oplus y$	$x \oplus y$				
2	10	$E_h(m) \oplus w$	(h,m)	$h \oplus m \oplus y$	$k \oplus x \oplus y$				
3	30	$E_h(w) \oplus m$	$(h, h \oplus m)$	$m \oplus y$	$k \oplus x \oplus y$				
4	38	$E_h(w) \oplus w$	$(h, h \oplus m)$	$h \oplus m \oplus y$	$x \oplus y$	at least	at most	at least	at most
5	19	$E_m(h) \oplus h$	(m,h)	$h \oplus y$	$x \oplus y$	$\frac{1}{4e}\, q^2/2^n$	$q^2/2^n$	$q/2^n$	$2\,q/2^n$
6	21	$E_m(h) \oplus w$	(m,h)	$h \oplus m \oplus y$	$k \oplus x \oplus y$	by	by	by	by
7	33	$E_m(w) \oplus h$	$(m, h \oplus m)$	$h \oplus y$	$k \oplus x \oplus y$	Th. 15	Th. 7	Th. 16	Th. 11
8	37	$E_m(w) \oplus w$	$(m, h \oplus m)$	$h \oplus m \oplus y$	$x \oplus y$				
9	20	$E_w(h) \oplus h$	$(h \oplus m, h)$	$h \oplus y$	$x \oplus y$				
10	17	$E_w(h) \oplus m$	$(h \oplus m, h)$	$m \oplus y$	$k \oplus x \oplus y$				
11	7	$E_w(m) \oplus h$	$(h \oplus m, m)$	$h \oplus y$	$k \oplus x \oplus y$				
12	4	$E_w(m) \oplus m$	$(h \oplus m, m)$	$m \oplus y$	$x \oplus y$				
13	13	$E_m(h) \oplus c$	(m,h)	$c \oplus y$	$c \oplus y$				
14	15	$E_m(h) \oplus m$	(m,h)	$m \oplus y$	$k \oplus y$				
15	25	$E_m(w) \oplus c$	$(m, h \oplus m)$	$c \oplus y$	$c \oplus y$	at least	at most	at least	at most
16	29	$E_m(w) \oplus m$	$(m, h \oplus m)$	$m \oplus y$	$k \oplus y$	$\frac{1}{4e}\, q^2/2^n$	$q^2/2^n$	$\frac{1}{2e} q^2/2^n$	$q^2/2^n$
17	14	$E_w(h) \oplus c$	$(h \oplus m, h)$	$c \oplus y$	$c \oplus y$	by	by	by	by
18	23	$E_w(h) \oplus w$	$(h \oplus m, h)$	$h \oplus m \oplus y$	$k \oplus y$	Th. 15	Th. 10	Th. 18	Th. 14
19	2	$E_w(m) \oplus c$	$(h \oplus m, m)$	$c \oplus y$	$c \oplus y$				
20	11	$E_w(m) \oplus w$	$(h \oplus m, m)$	$h \oplus m \oplus y$	$k \oplus y$				

Fig. 2. Definitions of group-1 and group-2 schemes, and results on them. Column 1 is our index s (we write f_s for the compression function and H_s for the iterated construction). Column 2 is the index used by PGV [30]. Column 3 defines the compression functions where, for compactness, $w = h \oplus m$. Columns 4–6 are the f', f'', f^* functions described in Sect. 3. Columns 7–10 summarize our collision-resistance and preimage-resistance bounds and where they can be found. We omit mention of the domain of validity for the bounds as well as ± 1 or ± 2 addends on q.

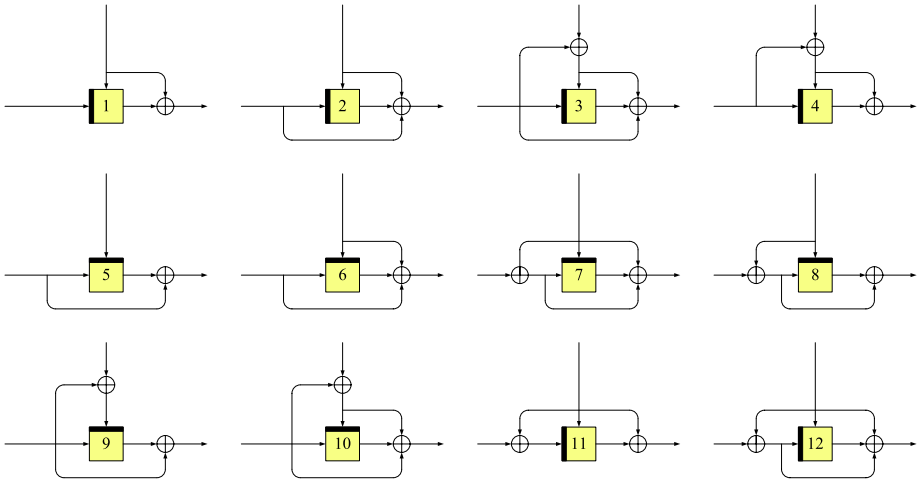


Fig. 3. Compression functions $f_1, \ldots, f_{12}$ for the group-1 hash functions $H_1, \ldots, H_{12}$. The blockcipher's key k comes in at the darkened side of the box, and the plaintext block x comes in at the other edge with an incoming arc. Both are n bits. The incoming chaining value $h = h_{i-1}$ enters on the left, and the incoming message block $m = m_i$ enters from the top. The outgoing chaining value $h' = h_i$ exits out the right.

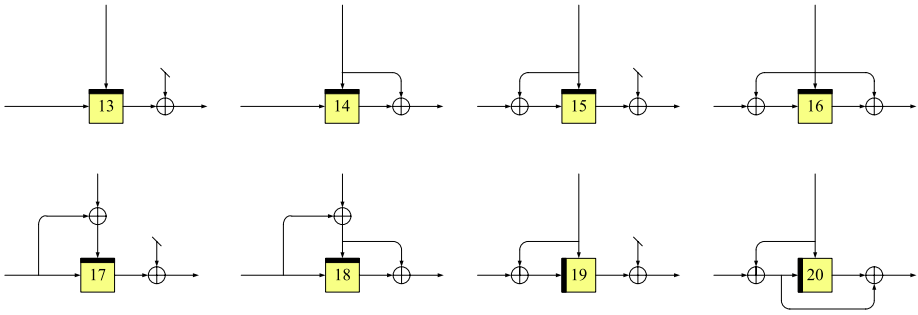


Fig. 4. Compression functions $f_{13}, \ldots, f_{20}$ for the group-2 hash functions $H_{13}, \ldots, H_{20}$. Notation is as in Fig. 3. The slanted hatch mark in compression functions 13, 15, 17, and 19 represents an arbitrary constant c.

It is important not to read too much or too little into ideal-cipher-model proofs. On one hand, attacks on (explicitly) blockcipher-based hash functions have traditionally treated the blockcipher as a black box: they do not exploit the structure of the blockcipher at all. Such attacks are doomed when one has strong results in the ideal-cipher model. On the other hand, the only structural aspect of a blockcipher captured by the model is its invertibility, so one must be skeptical about what an ideal-cipher-model proof means as soon as one employs a blockcipher with significant structural properties: perhaps there is a way for the adversary to exploit these properties. Properties like the existence of "weak keys" are problematic in this setting. Still, the ideal-cipher model is considerably sharper than modeling the blockcipher as a random function $E\colon \{0,1\}^{2n} \to \{0,1\}^n$, and such a model should be avoided because many attacks on

blockcipher-based hash functions *do* use the adversary's ability to compute E_k^{-1}. Overall, we see the ideal-cipher model as an appropriate first step in understanding the security of blockcipher-based hash functions.

Message Padding As with PGV [30], we do not concern ourselves with messages that are not multiples of the block length n; one can always deal with such messages by padding techniques. One common method, known as *MD-strengthening* [10,27], is to 0-pad the message to a multiple of the block length n and then to add in one more block that specifies the length of the original message (modulo 2^n, say). Simple results establish the security for such techniques in defining a hash function H^* with domain $\{0,1\}^*$ given a hash function H with domain $(\{0,1\}^n)^*$. All of our attacks likewise work in the presence of MD-strengthening.

Publication History The proceedings version of this paper appeared in *CRYPTO 2002* [6]. Many of the proofs in the current version have been simplified or otherwise cleaned up. In particular, one of the deficiencies in the proceedings version was that, in proofs, we would take an example function from a class of hash functions, prove a claim for it, and then say that the other functions worked the same way. Such claims were hard to verify. The current version instead describes proofs in terms of basic properties of the hash functions, an approach employed by Stam [40]. We have imported the coauthor as well as the approach.

Subsequent Work There has been considerable follow-on work since the proceedings version of this paper [6], some of which we sketch in Sect. 11.

2. Preliminaries

Basic Notions Given a finite set S, we write $x \xleftarrow{\$} S$ for the experiment of uniformly selecting a random element from S and calling it x. An *adversary* is an algorithm with access to one or more oracles. We write these as superscripts.

We restrict attention to blockciphers whose key-length and block-length are the same number $n \geq 1$. A *blockcipher* is then a map $E\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$, where, for each $k \in \{0,1\}^n$, the function $E_k(\cdot) = E(k,\cdot)$ is a permutation on $\{0,1\}^n$. If E is a blockcipher, then E^{-1} is its inverse, where $E_k^{-1}(y)$ is the string x such that $E_k(x) = y$. Let $\mathrm{Bloc}(n)$ be the set of all blockciphers $E\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. Choosing a random element of $\mathrm{Bloc}(n)$ means that, for each $k \in \{0,1\}^n$, one chooses a uniformly random permutation $E_k(\cdot)$.

A (blockcipher-based) *hash function* is a map $H\colon \mathrm{Bloc}(n) \times D \to R$ where $n \geq 1$, $D \subseteq \{0,1\}^*$, and $R = \{0,1\}^n$. We stress that a hash function is not allowed to depend arbitrarily on its first argument; access to the specified element $E \in \mathrm{Bloc}(n)$ is black-box. In particular the function H must be given by a program that, given M, computes $H^E(M) = H(E,M)$ using an E-oracle. When hash function $f\colon \mathrm{Bloc}(n) \times D \to R$ has $D = \{0,1\}^n \times \{0,1\}^n$, we call it a *compression function.* (We do not distinguish between elements in $\{0,1\}^n \times \{0,1\}^n$ and those in $\{0,1\}^{2n}$, assuming some canonical way such as concatenation to shift between representations. In this way we consider $\{0,1\}^n \times \{0,1\}^n$ a subset of $\{0,1\}^*$.) Fix $h_0 \in \{0,1\}^n$. The *iterated hash* of

compression function $f\colon \mathrm{Bloc}(n) \times (\{0,1\}^n \times \{0,1\}^n) \to \{0,1\}^n$ is the hash function $H\colon \mathrm{Bloc}(n) \times (\{0,1\}^n)^* \to \{0,1\}^n$ defined by $H^E(m_1 \cdots m_\ell) = h_\ell$, where $h_i = f^E(h_{i-1}, m_i)$. Set $H^E(\varepsilon) = h_0$, where ε denotes the empty string. If the program for f uses a single query $E(k,x)$ to compute $f^E(h,m)$, then f (and its iterated hash H) is said to be *rate-1*. We often omit the superscript E to f and H. Sometimes we wish to stress the parameter n (the block-length), and then we write f^n or H^n.

Definition 1 (Group-1 and group-2 PGV hash functions). Fix $n \geq 1$. For any $s \in [1..20]$ (our numbering convention), Fig. 2 serves to define the group-1 ($s \in [1..12]$) respectively group-2 ($s \in [13..20]$) compression functions

$$f_s^n : \mathrm{Bloc}(n) \times \big(\{0,1\}^n \times \{0,1\}^n\big) \to \{0,1\}^n.$$

Given a fixed $h_0 \in \{0,1\}^n$, the compression functions above induce iterated hash functions H_s^n by way of $H_s^n(\varepsilon) = h_0$ and $H_s^n(m_1 \ldots m_\ell) = f_s^n(H_s^n(m_1 \ldots m_{\ell-1}), m_\ell)$.

Often we omit writing the subscript s or superscript n from a hash function or compression function. Sometimes we will talk of the iterated hash function without explicitly fixing h_0 first; suffice it to say our results hold for all fixed $h_0 \in \{0,1\}^n$.

Security Notions A systematic exploration of different notions of hash-function security is beyond the scope of this paper; see [33] for some work in this direction. In the sequel, we specify the notions for collision resistance and preimage resistance that we will use.

To quantify the collision resistance of a blockcipher-based hash function H, we instantiate the blockcipher E by uniformly sampling it from $\mathrm{Bloc}(n)$. An adversary A is given oracles for $E(\cdot,\cdot)$ and $E^{-1}(\cdot,\cdot)$ and wants to find a *collision* for H^E—that is, M, M' where $M \neq M'$ but $H^E(M) = H^E(M')$. We look at the number of queries that the adversary makes and compare this to the probability that the adversary finds a collision.

Definition 2 (Collision resistance of a hash function). Let H be a blockcipher-based hash function, $H\colon \mathrm{Bloc}(n) \times D \to R$, and let A be an adversary. Then the advantage of A in finding collisions in H is the real number

$$\mathbf{Adv}_H^{\mathrm{coll}}(A) = \Pr\big[E \xleftarrow{\$} \mathrm{Bloc}(n);\ (M, M') \xleftarrow{\$} A^{E,E^{-1}} :\ M \neq M' \text{ and } H^E(M) = H^E(M')\big].$$

For $q \geq 1$, we write $\mathbf{Adv}_H^{\mathrm{coll}}(q)$ for $\max_A\{\mathbf{Adv}_H^{\mathrm{coll}}(A)\}$, where the maximum is taken over all adversaries that ask at most q oracle queries (E-queries plus E^{-1}-queries). Other advantage measures are silently extended in the same way.

We also define the advantage of an adversary in finding collisions in a compression function $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. Naturally (h,m) and (h',m') collide under f if they are distinct and $f^E(h,m) = f^E(h',m')$, but we also give credit for finding an (h,m) such that $f^E(h,m) = h_0$ for a fixed $h_0 \in \{0,1\}^n$. If one treats the hash of the empty string as the constant h_0, then $f^E(h,m) = h_0$ amounts to having found a collision between (h,m) and the empty string.

Definition 3 (Collision resistance of a compression function). Let f be a blockcipher-based compression function, $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. Fix a constant $h_0 \in \{0,1\}^n$ and an adversary A. Then the advantage of A in finding collisions in f is the real number

$$\mathbf{Adv}_f^{\mathrm{comp}}(A) = \Pr\big[E \xleftarrow{\$} \mathrm{Bloc}(n);\ \big((h,m),(h',m')\big) \xleftarrow{\$} A^{E,E^{-1}} :$$
$$\big((h,m) \neq (h',m') \ \wedge\ f^E(h,m) = f^E(h',m')\big) \ \vee\ f^E(h,m) = h_0\big].$$

Though we focus on collision resistance, we are also interested in preimage resistance. We consider defining this in two natural ways.

Definition 4 (Preimage resistance). Let $H\colon \mathrm{Bloc}(n) \times D \to R$ be a blockcipher-based hash function, and let A be an adversary. Let λ be a number such that $D_\lambda = D \cap \{0,1\}^\lambda$ is nonempty. Then the advantage of A in finding preimages for H is measured by the real numbers

$$\mathbf{Adv}_H^{\mathrm{pre1}}(A) = \Pr\big[E \xleftarrow{\$} \mathrm{Bloc}(n);\ \sigma \xleftarrow{\$} R;\ M \leftarrow A^{E,E^{-1}}(\sigma) :\ H^E(M) = \sigma\big] \quad \text{and}$$
$$\mathbf{Adv}_H^{\mathrm{pre2}}(A,\lambda) = \Pr\big[E \xleftarrow{\$} \mathrm{Bloc}(n);\ M \xleftarrow{\$} D_\lambda;\ \sigma \leftarrow H^E(M);\ M \xleftarrow{\$} A^{E,E^{-1}}(\sigma):$$
$$H^E(M) = \sigma\big].$$

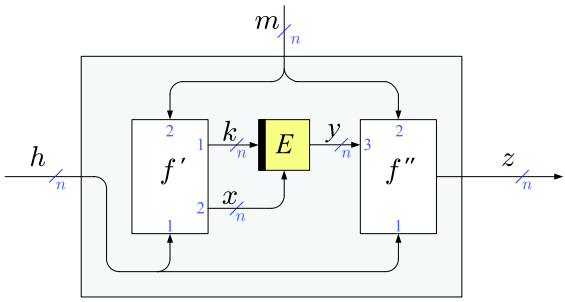
The first formalization speaks to the difficulty of finding a preimage for a random range point, while the second speaks to the difficulty of finding a preimage for the image of a random domain point. The latter is better aligned with the traditional notion of a one-way function, but it is more cumbersome to work with because one must introduce an extra parameter (the value λ) and use it to split the domain into finite sets. We therefore almost always prefer to use the "pre1" notion. We will justify this choice for our particular context in Sect. 10, showing that, for the 20 PGV hash functions we consider, the two measures almost coincide.

Conventions For the remainder of this paper, we assume the following significant conventions. First, an adversary does not ask any oracle query in which the response is already known; namely, if A asks a query $E_k(x)$ and this returns y, then A does not ask a subsequent query of $E_k(x)$ or $E_k^{-1}(y)$; and if A asks $E_k^{-1}(y)$ and this returns x, then A does not ask a subsequent query of $E_k^{-1}(y)$ or $E_k(x)$. Second, when a (collision-finding) adversary A for H outputs M and M', adversary A must have already computed $H^E(M)$ and $H^E(M')$, in the sense that A has made the necessary E or E^{-1} queries to compute $H^E(M)$ and $H^E(M')$ according to the program that defines H. Similarly, we assume that a (collision-finding) adversary A for the compression function f computes $f^E(h,m)$ and $f^E(h',m')$ prior to outputting (h,m) and (h',m'). Similarly, when a (preimage-finding) adversary A for H outputs a message M, we assume that A has already computed $H^E(M)$, in the sense that A has made the necessary E or E^{-1} queries to compute this value. These assumptions are all without loss of generality, in that an adversary A not obeying these conventions can easily be modified to give an adversary A' having similar computational complexity that obeys these conventions and has the same advantage as A.

3. Properties of Group-1 and Group-2 Compression Functions

To simplify proving collision-resistance and preimage-resistance for the *twelve* group-1 hash functions and the *eight* group-2 hash functions—conceptually, 40 different theorems—it is convenient to define a combinatorial property shared by each hash-function member in the group. This allows one to simultaneously prove security properties enjoyed by every group-1 hash function—and any *other* function satisfying the stated combinatorial property—and similarly for the group-2 schemes. This section defines the needed property.

Functions Associated to a Compression Function We begin by defining some functions that are naturally associated to a rate-1 compression function. If $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is a rate-1 compression function, then it can be "factored" into functions f' and f'', where $f' : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^n$ is the *preprocessing function* that prepares the input to be fed to the blockcipher, and where $f'' : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is the *postprocessing function* that determines the final output by way of $z = f^E(h,m) = f''(h,m,y)$ for $(k,x) = f'(h,m)$ and $y = E_k(x)$. See the inset figure. If f' is bijective, we further define $f^* : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ by $f^*(k,x,y) = f''(h,m,y)$, where $(h,m) = f'^{-1}(k,x)$. This function maps the input and output of the blockcipher, (k,x,y), to the corresponding output of the compression function, $z = f(h,m)$.



Compression-Function Properties Let $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be the compression function of a rate-1 hash function. Then we define the following four properties that may or may not hold on f:

- P1 The function f' is bijective. Recall that this function maps $(h,m) \in \{0,1\}^n \times \{0,1\}^n$ to $(k,x) \in \{0,1\}^n \times \{0,1\}^n$.
- P2 The function $f''(h,m,\cdot)$ is bijective for all $(h,m) \in \{0,1\}^n \times \{0,1\}^n$. Note that the specified function maps $y \in \{0,1\}^n$ to $z \in \{0,1\}^n$.
- P3 The function $f^*(k,\cdot,y)$ is bijective for all $(k,y) \in \{0,1\}^n \times \{0,1\}^n$. Note that the function maps $x \in \{0,1\}^n$ to $z \in \{0,1\}^n$ and that mention of f^* already assumes property P1.
- P4 The function $f'(h,\cdot)|_1$, the projection of f' to its first component of output, is bijective for all $h \in \{0,1\}^n$. Note that the specified projection maps $m \in \{0,1\}^n$ to $k \in \{0,1\}^n$.

We also name the following two properties, which are just collections of the properties of above: we say that $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ has property:

T1 The function f has properties P1, P2, and P3.
T2 The function f has properties P1, P2, and P4, but not property P3.

Let us take a moment to build some intuition for these definitions. Property P1 ensures that blockcipher inputs (k, x) are in one-to-one correspondence with compression-function inputs (h, m). Property P2 ensures that there is a one-to-one correspondence between blockcipher outputs and compression-function outputs for any compression-function inputs. In particular, this ensures that every bit of the blockcipher output affects the output of the compression function. Property P3 ensures that every bit of the inverse blockcipher output affects the output of the compression function. Finally, assuming that property P1 holds, property P4 ensures that every bit of the inverse blockcipher output affects the part of the input of the compression function that serves as chaining variable in the iterated construction. This last interpretation relies on an alternative formulation of property P4 in the case property P1 holds.

P4′ The function $f'^{-1}(k,\cdot)|_1$, the projection of f'^{-1} to its first component of output, is bijective for all $k \in \{0,1\}^n$. Note that the specified projection maps $x \in \{0,1\}^n$ to $h \in \{0,1\}^n$.

Lemma 5. *Let f' be given having property* P1. *Then it has property* P4 *iff it has property* P4′.

Proof. We prove only that having property P4 implies having P4′ (by contradiction), the other implication follows by a symmetrical argument. Suppose that property P4′ does not hold; then $f'^{-1}(k,\cdot)|_1$ is not injective for some $k \in \{0,1\}^n$, and there exist $x, x' \in \{0,1\}^n$, $x \neq x'$ such that $f'^{-1}(k,x)|_1 = f'^{-1}(k,x')|_1$. Let $h, m, h', m' \in \{0,1\}^n$ be such that $(h,m) = f'^{-1}(k,x)$ and $(h',m') = f'^{-1}(k,x')$. Then in particular $h = h', m \neq m'$, yet $f'(h,m)|_1 = k = f'(h,m')|_1$, violating property P4. □

Properties of the Group-1 and Group-2 Hash Functions The table of Fig. 2 not only defines the compression function f for each group-1 and group-2 scheme, but, in addition, it specifies the corresponding functions f' (column 4) and f'' (column 5) and, since all f' are bijective, function f^* (column 6). From this information it is immediate to verify that properties P1 and P2 hold for functions $f_1, \ldots, f_{20}$, property P3 holds for $f_1, \ldots, f_{12}$ but not for $f_{13}, \ldots, f_{20}$, and property P4 holds for $f_5, \ldots, f_{20}$. In particular then, group-1 compression functions have the property we named T1, while group-2 compression functions have the property we named T2. We note this below.

Proposition 6. *The group-*1 *compression functions each have property* T1, *while the group-*2 *compression functions have property* T2.

It will follow from results of this paper that the above statement is actually an if and only if: the remaining 44 PGV compression functions have neither property T1 nor T2.

The group-1 compression functions could be further subdivided into those that additionally have property P4 (namely $f_5, \ldots, f_{12}$) and those that do not ($f_1, \ldots, f_4$). In the context of preimage and collision resistance we will not consider this subdivision, but when other security properties are considered, it can surface. For instance, the eight schemes for which PGV found fixed-point attacks are precisely those having property P4.

4. Collision Resistance of the Group-1 Schemes

The group-1 hash functions can all be analyzed using the Merkle–Damgård paradigm. Our security bound is identical for all of these schemes.

Theorem 7 (Collision resistance of the group-1 hash functions). *Fix $n \geq 1$ and let H be a group-1 hash function. Then $\mathbf{Adv}^{\mathrm{coll}}_{H^n}(q) \leq q(q+1)/2^n$ for any $q \geq 1$.*

The proof combines a lemma showing the collision resistance of a group-1 compression-function with the classical result, stated for the ideal-cipher model, showing that a hash function is collision resistant if its compression function is.

Lemma 8 (Merkle–Damgård [10,27] in the ideal-cipher model). *Let f be a compression-function $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$, and let H be the iterated hash of f. Then $\mathbf{Adv}^{\mathrm{coll}}_{H}(q) \leq \mathbf{Adv}^{\mathrm{comp}}_{f}(q)$ for all $q \geq 1$.*

Proof. Let E be the blockcipher upon which f is based. Let A be a collision-finding adversary for H that takes two oracles, E, E^{-1}. We construct from A a collision-finding adversary B for f. Adversary B also takes oracles E, E^{-1}. Let B run A. When A makes an E (resp., E^{-1}) query, adversary B forwards it to E (resp., E^{-1}) and returns the result to A. For $i \in [1..q]$, we say that the ith triple is (x_i, k_i, y_i) if A's ith oracle query was an E-query of (k_i, x_i) and this returned y_i, or else A's ith oracle query was an E^{-1}-query (k_i, y_i) and this returned x_i. Algorithm B records the list of triples. Eventually A halts with an output $(M, M') = (m_1 \cdots m_\ell, m'_1 \cdots m'_{\ell'})$.

Have B compute $H(M)$ and $H(M')$. According to our conventions, all of the necessary queries for B to use in this computation are already recorded in B's list of triples, so no new oracle calls are needed to compute $H(M)$ and $H(M')$. We want to show that whenever A is successful in finding a collision (in the hash function), B will be successful in finding a collision in the compression function. Hence, suppose $M \neq M'$ and $H(M) = H(M')$, and, by symmetry, we can assume without loss of generality that $\ell \geq \ell'$. If $\ell' = 0$ (and necessarily $\ell > 0$), it holds that $f(h_{\ell-1}, m_\ell) = h_0$ and B can claim success by outputting $(h_{\ell-1}, m_\ell)$. Otherwise, if $\ell' > 0$, we can write $h_\ell = f(h_{\ell-1}, m_\ell) = f(h'_{\ell'-1}, m'_{\ell'}) = h'_{\ell'}$. (Primed variables are understood to be associated to $H(M')$.) If $h_{\ell-1} \neq h'_{\ell'-1}$ or $m_\ell \neq m'_{\ell'}$, then B returns $(h_{\ell-1}, m_\ell), (h'_{\ell'-1}, m'_{\ell'})$. Otherwise, $h_{\ell-1} = f(h_{\ell-2}, m_{\ell-1}) = f(h'_{\ell'-2}, m'_{\ell'-1}) = h'_{\ell'-1}$. Again, if $h_{\ell-2} \neq h'_{\ell'-2}$ or $m_{\ell-1} \neq m_{\ell'-1}$, then B wins by returning $(h_{\ell-2}, m_{\ell-1}), (h'_{\ell'-2}, m'_{\ell'-1})$. Proceeding in this way, we find values $\alpha \in [1..\ell]$ and $\beta \in [1..\ell']$ such that either $h_\alpha = f(h_{\alpha-1}, m_\alpha) = f(h'_{\beta-1}, m'_\beta) = h'_\beta$ (yet $(h_{\alpha-1}, m_\alpha) \neq h'_{\beta-1}, m'_\beta)$) or $f(h_{\alpha-1}, m_\alpha) = h_0$, so B will be able to output a collision for f. □

Lemma 9 (Collision resistance of the group-1 compression functions). *Fix $n \geq 1$ and a constant $h_0 \in \{0,1\}^n$, and let f be a group-1 compression function. Then* $\mathbf{Adv}^{\mathrm{comp}}_{f^n}(q) \leq q(q+1)/2^n$ *for any* $q \geq 1$.

Proof. Let E be the blockcipher upon which f is based. Let A be an adversary with oracle access to E, E^{-1} that attacks the compression function f. Assume that A asks its oracles a total of q queries. A collision consists either of two pairs (h,m) and (h',m') satisfying $f^E(h,m) = f^E(h',m')$ yet $(h,m) \neq (h',m')$ or of a single pair (h,m) satisfying $f^E(h,m) = h_0$. We will maintain a list of triples (h,m,z) such that $z = f^E(h,m)$ and the adversary has made the relevant queries to E and/or E^{-1}. 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 occurring in this list. We show that any query, be it to E or E^{-1}, will add at most one triple (h,m,z) to this list of computable compression function evaluations; moreover the value z is almost completely out of the adversary's control.

Consider a forward query (k,x). By bijectivity of f' (property P1), there is a unique pair (h,m) corresponding to this query. Thus, each forward query will add one triple (h,m,z) to the adversary's list of computable values. Since $f''(h,m,\cdot)$ is bijective for all h,m, the distribution of compression function output z 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 E^{-1}) 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 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 f'' with the pair (h,m) consistent with the (k,x) query being made, leading to a possible compression function outcome h^*. Because f'' is bijective when (h,m) are fixed (property P2), distinct y^* lead to distinct z^*, so there are $2^n - t$ possible outcomes z^*, all equally likely.

Similarly, consider an inverse query (k,y). This yields a unique x, and hence by bijectivity of f' (property P1), there is a unique pair (h,m) corresponding to this query once answered. Thus, each inverse query will add one triple (h,m,z) to the adversary's list of computable values. This time bijectivity of $f^*(k,\cdot,y)$ implies that the distribution of z is closely related to the (almost uniform) output distribution of E^{-1}. 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 E_k^{-1} 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 f'^{-1} and f'' with k and y to a triple (h,m,z). Because for all k and y, the mapping from x to z is bijective (property P3), distinct x^* lead to distinct z^*, so there are $2^n - t$ possible outcomes z^*, all equally likely.

As a result, after $i-1$ queries the list of computable values contains $i-1$ triples (h,m,z). The ith query will add one triple with z uniform over a set of size at least $2^n - (i-1)$. Thus the probability that the ith query causes a collision with any of these triples or h_0 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 $\sum_{i=1}^{q} i/(2^n - (i-1)) \leq \frac{1}{2}q(q+1)/(2^n - q)$.

If $q \le 2^{n-1}$, our expression is at most $\frac{1}{2^{n-1}} \frac{q(q+1)}{2} = \frac{q(q+1)}{2^n}$. Since the bound is vacuous when $q > 2^{n-1}$ we can drop the assumption that $q \le 2^{n-1}$ and conclude $\mathbf{Adv}^{\mathrm{comp}}_{f^n}(q) \le q(q+1)/2^n$. □

5. Collision Resistance of the Group-2 Schemes

We cannot use the Merkle–Damgård paradigm for proving the security of group-2 schemes because their compression functions are *not* collision-resistant. Indeed, compression functions that belong to group-2 are easily invertible thus also allowing quick collision finding. For example, one can break $f_{14}(h, m) = E_m(h) \oplus m$ as a compression function by choosing any two distinct $m, m' \in \{0,1\}^n$, computing $h = E_m^{-1}(m)$ and $h' = E_{m'}^{-1}(m')$, and outputting (h, m) and (h', m'). All the same, group-2 hash functions enjoy almost the same upper bound on collision resistance as group-1 schemes.

The proof we give differs significantly from the original given in [6]. In particular, the original proof was based upon coloring a directed graph. The 3^n vertices represent queries with all possible answers and have outdegree 2^n where the 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. Coloring is added dynamically depending on the actual query responses. This leads to unwieldy graphs with a complicated notion of what constitutes a collision.

Here we consider an undirected graph where vertices correspond to chaining values and edges are dynamically added whenever a query (to E or E^{-1}) has been made that would allow one to move from one chaining value to the next. This simplification, originally by Duo and Li [13], leads to a tighter bound than was originally presented, mainly because we no longer need to distinguish several cases whose success probabilities are subsequently added.

A similar approach was taken by Lucks [23], who considers a directed graph where vertices correspond to chaining values and arcs are drawn (or colored) whenever a query has been made that would allow one to move from one chaining value to the next. However, the fact that Lucks's graph is directed complicates his proof forcing some extra case analysis. Dispensing with the direction of the arcs (that thus become edges) leads to simpler proofs, despite seemingly aiding the adversary (certain patterns in the graph will be deemed a success even when the underlying event on the hash function is not).

Theorem 10 (Collision resistance of the group-2 hash functions). *Fix $n \ge 1$ and let H be a group-2 hash function. Then $\mathbf{Adv}^{\mathrm{coll}}_{H^n}(q) \le q(q+1)/2^n$ for all $q \ge 1$.*

Proof. Let $h_0 = H(\epsilon) \in \{0,1\}^n$ be the initial value of H. Consider 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 where the edge set E_G is initially empty. We will dynamically add edges based on the queries to E and E^{-1}. In particular, an edge (h, z) labeled by m is added if there is a message m such that $z = f^E(h, m)$ (or $h = f^E(z, m)$) for which relevant query to either E or E^{-1} has been made. We claim that to find a collision would require constructing a "ρ-shape" in the graph that contains the (fixed)

initial value h_0. Suppose that $H(M) = H(M')$ with $M \neq M'$. Write $M = (m_1, \ldots, m_\ell)$ and $M' = (m'_1, \ldots, m'_{\ell'})$ and correspondingly $h_0, \ldots, h_\ell$ respectively $h'_0, \ldots, h'_{\ell'}$ for the chaining values of the iterated hash. Note that $h'_0 = h_0$ and $h_\ell = h'_{\ell'}$. Assume without loss of generality that $\ell \geq \ell'$. Because $M \neq M'$, there exists a t such that $m_i = m'_i$ for all $0 \leq i < t$ but $m_t \neq m'_t$ (or possibly $\ell \geq t > \ell'$). As a result, the paths $(h_0, \ldots, h_t)$ and $(h'_0, \ldots, h'_t)$ are identical, but the edges (h_t, h_{t+1}) and (h'_t, h'_{t+1}) are distinct, even when h_{t+1} happens to equal h'_{t+1} (in particular, the edges are labeled differently). Since $h_\ell = h'_{\ell'}$ 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 actually lead to a collision, but for analysis it will be sufficient to allow the adversary this extra credit.

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 either a cycle or the initial value h_0. The first claim is that after i oracle queries, the set T has cardinality at most $i + 1$. Indeed, the component containing h_0 has at most $i' + 1$ nodes when it has i' edges. A component with i' edges that contains a cycle has at most i' nodes. Thus the component of G that contains h_0 is the only one that causes the number of nodes (in T) to be larger than the number of edges (in G), and this by at most one. Bijectivity of f' (property P1) implies that an oracle 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 hence at most $i + 1$ nodes in the set T. This justifies our first claim.

Our second claim is that to complete a ρ-shape, either (1) a cycle has to be completed within the h_0-component, or (2) the h_0-component needs to become 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 ith query a collision is found by a forward query is at most $i/(2^n - i)$: bijectivity of $f''(h, m, \cdot)$ (property P2) ensures that z 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 ith query using an inverse query is at most $i/(2^n - i)$: this time bijectivity of $f^*(k, \cdot, y)$ (property P4′) ensures that h is uniformly distributed over a set of size at least $2^n - i$.

We can now conclude that the probability of finding a collision on the ith query is at most $i/(2^n - i)$ and the probability after q queries is at most $\sum_{i=1}^{q} i/(2^n - i) \leq \frac{1}{2}q(q+1)/(2^n - q)$. If $q \leq 2^{n-1}$, this expression is at most $\frac{1}{2^{n-1}} \frac{q(q+1)}{2} = \frac{q(q+1)}{2^n}$. Since the bound is vacuous when $q > 2^{n-1}$, we can drop the assumption that $q \leq 2^{n-1}$ and conclude that $\mathbf{Adv}^{\text{coll}}_{H^n}(q) \leq q(q+1)/2^n$. □

6. Preimage Resistance of the Group-1 Schemes

From the perspective of collision resistance there is no reason to favor any particular scheme, be it group-1 or just group-2. However, the schemes *can* be separated based on their preimage resistance. In particular, for an n-bit blockcipher, an adversary attacking a group-1 hash function requires nearly 2^n oracle queries to do well at finding a preimage for a random range point. In contrast, an adversary attacking a group-2 hash function—with invertible compression function—requires roughly $2^{n/2}$ oracle queries to do the same job. This renders the group-2 schemes $H_{13..20}$ less secure than their group-1 siblings $H_{1..12}$.

In Sects. 7 and 8 we will address the preimage resistance of the group-2 schemes (Theorems 14 and 18), but we begin with the theorem establishing good preimage-resistance for the group-1 schemes. The theorem is immediate from the two lemmas that follow it. The first lemma is analogous to Lemma 8, and the second shows that group-1 compression functions have good preimage resistance.

Theorem 11 (Preimage resistance of the group-1 hash functions). *Fix $n \geq 1$ and let H be a group-1 hash function. Then $\mathbf{Adv}^{\mathrm{pre1}}_{H^n}(q) \leq q/2^{n-1}$ for any $q \geq 1$.*

Lemma 12 (Merkle–Damgård for pre1-security). *Let $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a compression function, and H be its iterated hash. Then $\mathbf{Adv}^{\mathrm{pre1}}_{H}(q) \leq \mathbf{Adv}^{\mathrm{pre1}}_{f}(q) + 2^{-n}$ for all $q \geq 1$.*

Proof. Let A be an adversary for H: adversary A takes oracles E, E^{-1} and an input σ, and, when successful, it outputs M such that $H^E(M) = \sigma$. We construct an adversary B for f: adversary B takes oracles E, E^{-1} and an input σ, and, when successful, it outputs (h, m) such that $f^E(h, m) = \sigma$. Adversary B works as follows. It runs A on σ. When A makes an E (resp., E^{-1}) query, adversary B forwards the query to its E (resp., E^{-1}) oracle and returns to A the result. During this process, for each $i \in [1..q]$, we say that the ith triple is (x_i, k_i, y_i) if A's ith oracle query was an E-query of (k_i, x_i) and this returned y_i, or else A's ith oracle query was an E^{-1}-query (k_i, y_i) and this returned x_i. Adversary B records the list of triples. Eventually A halts with an output $M = m_1 \cdots m_\ell$. Assume for the moment that M is indeed a preimage of σ and that $\ell \geq 1$. Let B compute $H^E(M)$, i.e., for $i \leftarrow 1$ to ℓ, it sets $h_i \leftarrow f^E(h_{i-1}, m_i)$. According to our conventions, all of the $E_k(x)$ values that B requires for this computation are already in its list of triples, so no new oracle calls are required. Since we assume that A's output M is a preimage of σ, it must be that $h_\ell = f^E(h_{\ell-1}, m_\ell) = \sigma$, and so we have B output $(h_{\ell-1}, m_\ell)$ as its (valid) preimage. Now if A fails to return a preimage for σ or $\ell = 0$, we have B abort its execution thereby failing to return a preimage itself. We see then that B succeeds to find a preimage for σ whenever A does, unless A's preimage is the empty string. Since $H(\varepsilon) = h_0$, this would require $h_0 = \sigma$, which happens with probability 2^{-n} over the random choice of σ.

The statement $\mathbf{Adv}^{\mathrm{pre1}}_{H}(q) \leq \mathbf{Adv}^{\mathrm{pre1}}_{f}(q) + 2^{-n}$ follows by conditioning on aborting (or not) and by observing that (for the pre1-advantage notion) the distribution of the challenge σ is the same for the experiments over H and f. □

Lemma 13 (Preimage resistance of the group-1 compression functions). *Fix $n \geq 1$ and let f be a group-1 compression function. Then $\mathbf{Adv}^{\mathrm{pre1}}_{f^n}(q) \leq q/2^{n-1} - 2^{-n}$ for any $q \geq 1$.*

Proof. Let A be an adversary with oracles E, E^{-1} and input σ. Assume that A asks its oracles a total of q queries. We are interested in A's behavior when its left oracle is instantiated by $E \xleftarrow{\$} \mathrm{Bloc}(n)$ and its right oracle is instantiated by E^{-1}.

We recall the proof of Theorem 9, where we show that after $i-1$ queries (to E or E^{-1}) the list of computable values $z = f(h,m)$ contains $i-1$ triples (h,m,z). The ith query will add one triple with z uniform over a set of size at least $2^n-(i-1)$. Thus the probability that the ith 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 $\sum_{i=1}^{q} 1/(2^n-(i-1)) \le q/(2^n-q)$. This is smaller than $q/2^{n-1}-2^{-n}$ for $q < 2^{n-1}$ and vacuous for $q \ge 2^{n-1}$. □

7. Preimage Resistance of the Group-2 Schemes

We cannot use Lemma 12 to prove the security of the group-2 schemes because the associated compression functions are *not* preimage-resistant. For example, consider $f_{19}(h,m) = E(h \oplus m, m) \oplus c$. For any point σ, the adversary fixes $k=0$, computes $m = E_0^{-1}(\sigma \oplus c)$, and returns (m,m), which is always a correct inverse to σ. Still, despite these compression functions being invertible with a single oracle query, there is a reasonable security bound for the group-2 schemes.

Theorem 14 (Preimage resistance of the group-2 hash functions). *Fix $n \ge 1$ and let H be a group-2 hash function. Then $\mathbf{Adv}^{\mathrm{pre1}}_{H^n}(q) \le q(q+1)/2^n$ for any $q > 1$.*

Proof. Let $h_0 = H(\epsilon) \in \{0,1\}^n$. Let A be an adversary with oracles E, E^{-1} and input σ. Assume that A asks its oracles q queries in total. We are interested in A's behavior when its left oracle is instantiated by $E \xleftarrow{\$} \mathrm{Bloc}(n)$ and its right oracle is instantiated by E^{-1}.

As in the proof of Theorem 10, 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 E^{-1}. In particular, we add an edge (h,z), labeled by m, if we know a message m such that $z = f^E(h,m)$ (or $h = f^E(z,m)$) and the relevant query to either E or E^{-1} has been made. We claim that to find a preimage would require finding a path between the (fixed) initial vector and the (random) target σ: suppose that $H(M) = \sigma$, where $M = (m_1, \ldots, m_\ell)$ and correspondingly $h_1, \ldots, h_\ell$ for the chaining values of the iterated hash. Noting that $h_\ell = \sigma$, we see that $h_{0..\ell}$ is a (possibly empty) path from initial vector to target.

Since we are dynamically adding edges to the graph, components in the graph will also grow dynamically. Let T_0 be the set of all nodes that are in the component containing h_0, and similarly let T_σ be the set of all nodes connected to σ. Unless $h_0 = \sigma$, which happens with probability 2^{-n}, T_0 and T_σ are initially disjoint. However, when a path between h_0 and σ is present, we have $T_0 = T_\sigma$.

The first claim is that after i queries, the sets T_0 and T_σ have combined cardinality at most $i+2$. Indeed, either component has at most $i'+1$ nodes when i' edges are used. Bijectivity of f' (property P1) 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 $i+2$ nodes in T_0 plus T_σ.

The second claim is that to complete the path between h_0 and σ, an edge needs to be added with one end (node) in T_0 and the other in T_σ. Writing $T = T_0 \cup T_\sigma$, we need to find an edge with both nodes already part of T (cf. the proof of Theorem 10). Consider the ith query. For a forward query, bijectivity of $f''(h, m, \cdot)$ (property P2) ensures that the distribution of z is uniform over a set of size at least $2^n - i$. For an inverse query, bijectivity of $f^*(k, \cdot, y)$ (property P4′) ensures that the distribution of h is uniform over a set of size at least $2^n - i$. Consequently, the probability that on the ith query a preimage is found is upper bounded by $i/(2^n - i)$. (The set T contains $i + 1$ elements before the ith query, the query itself also needs to specify a node in T, but a self-loop cannot possibly connect T_0 and T_σ.)

With a union bound we can bound the probability of finding a preimage within q or fewer queries: $\mathbf{Adv}^{\mathrm{pre1}}_{H^n}(q) \le 1/2^n + \sum_{i=1}^{q} i/(2^n - i)$ leading to the stated upper bound. □

8. Matching Collision-Finding and Preimage-Finding Attacks

Matching Attacks on Collision Resistance In this section we show that the security bounds given in Sects. 4 and 5 are tight. We show a very general result for any rate-1 compression function. For concreteness sake, we only consider the scenario with block lengths corresponding to the previous section. In this case a standard 2-block birthday attack achieves advantage within a constant factor to the earlier upper bounds. The adversary's advantage below is bounded for any fixed blockcipher taking only randomization over the adversary's coins into account. We note that for specific constructions, the constant we get can be improved upon.

Theorem 15 (Finding collisions). *Let H be a rate-1 hash function based on compression function f: $\mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. Then $\mathbf{Adv}^{\mathrm{coll}}_{H^n}(q) \ge \frac{1}{4e}\frac{q(q-2)}{2^n}$ for any even number of queries $q \in [0..2^{(n+3)/2}]$.*

Proof. Given the hash function H, consider that hash function restricted to domain $\{0,1\}^{2n}$, that is, 2-block messages. For each message M, evaluation of $H(M)$ takes 2 calls to the underlying compression function f and hence 2 calls to blockcipher E. Thus, for q calls to E, we can evaluate H for $q' = q/2$ 2-block messages (assuming q even). If we pick the messages at random, the probability of finding a collision is governed by the birthday bound, in particular,

$$\mathbf{Adv}^{\mathrm{coll}}_{H^n}(q) \ge (1/e)q'(q'-1)/2^n = (1/4e)q(q-2)/2^n\,,$$

provided that $q' \le 2^{(n+1)/2}$ or, equivalently, that $q \le 2^{(n+3)/2}$. □

Matching Attacks on Preimage Resistance Finally, we prove that the security bounds given in Theorems 11 and 14 are tight, by describing adversaries that achieve advantage very close to the upper bounds. We begin with a general bound on the preimage resistance of rate-1 hash functions; thus it applies to both the group-1 and group-2 schemes. For this result, we find the pre2 notion most natural, but by virtue of Lemma 19 one also has a corresponding result for the pre1 notion.

Theorem 16 (Attacking the preimage-resistance of a general rate-1 hash function). *Let $n \geq 1$, and let $H\colon \mathrm{Bloc}(n) \times (\{0,1\}^n)^* \to \{0,1\}^n$ be the iterated hash function based on a rate-1 compression function $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. Then $\mathbf{Adv}^{\mathrm{pre2}}_{H^n}(q,n) \geq q/2^n$ for any $q \leq 2^n$.*

Proof. We describe an adversary A asking at most $q \geq 0$ queries and achieving the claimed advantage. Since $\{0,1\}^n \subset (\{0,1\}^n)^*$, by the definition of pre2-advantage the challenge digest $\sigma \in \{0,1\}^n$ for A is created by randomly selecting $m^* \in \{0,1\}^n$ and setting $\sigma = H^n(m^*) = f(h_0, m^*)$, where h_0 is the initial value for H^n.

Now on input σ, adversary A arbitrarily picks q distinct points $m_1, \ldots, m_q$ from $\{0,1\}^n$ (hence $q \leq 2^n$). Since f is rate-1, at most one query to E is required in order to evaluate $y_i = f(h_0, m_i) = H^n(m_i)$ for each $i \in [1..q]$; thus at most q queries are asked. If $y_i = \sigma$, we have A output (h_0, m_i) as valid preimage. As m^* was uniformly selected, the probability that $m^* = m_i$ for some i is $q/2^n$, and so the advantage of A is at least this. (Note that the advantage may be larger if σ has multiple preimages.) □

For the group-2 schemes, we begin with a lemma that lowerbounds the probability of finding a collision between two equal-sized multisets consisting of uniformly random samples from $\{0,1\}^n$. This lemma will allow us to likewise lowerbound the success of meet-in-the-middle preimage-finding attacks on the group-2 schemes.

Lemma 17. *Fix $n \geq 1$ and even $q \in [2..2^{n/2}]$. Let L_1 and L_2 be multisets of uniform random samples from $\{0,1\}^n$, where $|L_1| = |L_2| = q/2$. Then the probability that there exists an x such that $x \in L_1$ and $x \in L_2$ is at least $\frac{1}{2e}\frac{q^2}{2^n}$.*

Proof. We consider the elements of L_1 as being selected by throwing $q/2$ red balls uniformly into 2^n bins, these bins corresponding to the elements of $\{0,1\}^n$ in some understood way. Similarly, the elements of L_2 are selected by uniformly throwing $q/2$ blue balls. Let C denote the event that after throwing all q balls we have a bin containing at least two balls (regardless of color). Let $\mathsf{C_M}$ denote the event that there are only monochromatic collisions after throwing q balls (in other words, $\mathsf{C_M}$ denotes the event that C has occurred but for any pair of balls sharing a bin, they are either both blue or both red). Finally, let $\mathsf{C_B}$ denote a bichromatic collision (i.e., the event that there is some red ball and some blue ball sharing the same bin). So C is the disjoint union of $\mathsf{C_M}$ and $\mathsf{C_B}$, or $\Pr[\mathsf{C}] = \Pr[\mathsf{C_M}] + \Pr[\mathsf{C_B}]$.

Next we claim that, for any even $q > 1$, we have $\Pr[\mathsf{C_B}] > \Pr[\mathsf{C_M}]$. This claim is justified by a straightforward combinatorial argument: consider any configuration of q uncolored balls lying in 2^n bins where two balls share a bin. If we color these two balls blue, then there are $\binom{q-2}{q/2-2}$ ways to color the remaining balls. However, if we color one ball red and the other blue, there are $\binom{q-2}{q/2-1}$ ways to color the remaining balls. The latter number is larger than the former since $(q/2-2)!(q/2)! > (q/2-1)!(q/2-1)!$. This immediately implies our claim.

Since $q \leq 2^{n/2}$, we know that $\Pr[\mathsf{C}] \geq (1/e)\,q^2/2^n$ (for a proof see, for example, Appendix A of [2]). Therefore $\Pr[\mathsf{C_B}] > \Pr[\mathsf{C_M}]$ implies $\Pr[\mathsf{C_B}] > (1/2e)\,q^2/2^n$, completing the proof. □

With this lemma in hand, we proceed to the matching preimage-finding attacks on the group-2 schemes.

Theorem 18 (Attacking the preimage resistance of group-2 schemes). *Let $n \geq 1$, and let $H\colon \mathrm{Bloc}(n) \times (\{0,1\}^*)^n \to \{0,1\}^n$ be the iterated hash function based on a group-2 compression function $f\colon \mathrm{Bloc}(n) \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. Then $\mathbf{Adv}^{\mathrm{pre1}}_{H^n_s}(q) \geq \frac{1}{2e}\frac{q^2}{2^n}$ for any even $q \in [0..2^{n/2}]$.*

Proof. We construct an adversary A with two oracles E, E^{-1} that, on input $\sigma \in \{0,1\}^n$, will mount a meet-in-the-middle attack on H and return (with high probability) a two-block message $M = (m_1, m_2)$ for which $H(M) = \sigma$.

To begin, we observe that properties P1 and P4 together imply a bijection between pairs (h,k) and pairs (x,m) (where $\{h,m,k,x\}$ are the inputs to f and the inputs to E, respectively). Thus $h_0 = H^n(\varepsilon) \in \{0,1\}^n$ and arbitrary $k \in \{0,1\}^n$ uniquely determine the corresponding x and m. The first stage of the adversary's attack builds a set S_1 of size $q/2$ as follows: for $i \in [1..q/2]$, it sets $k^{(i)} = \langle i \rangle$ (where $\langle i \rangle$ means to represent i as an n-bit string in some canonical way), determines $x^{(i)}$ and $m^{(i)}$ from h_0 and $k^{(i)}$, queries $y^{(i)} \xleftarrow{\$} E_{k^{(i)}}(x^{(i)})$, computes $h^{(i)} = f''(h_0, m^{(i)}, y^{(i)})$, and adds $(m^{(i)}, h^{(i)})$ to S_1. Since each E-query uses a distinct key, the strings $y^{(i)}$ are uniformly and independently distributed; as $f''(h_0, m^{(i)}, \cdot)$ is a bijection (property P1), the $h^{(i)}$ are likewise distributed.

For the second stage, we observe from Fig. 4 that for any group-2 compression function, fixing any σ' in the range of f and any blockcipher key k uniquely determines an E^{-1}-query (k, y) (essentially because for PGV schemes, having properties P1 and P2 yet not P3 implies that $f^*(k, \cdot, y)$ is a constant function). Additionally, it is easy to see from Figs. 2 and 4 that for any particular key k, the distribution of $x \xleftarrow{\$} E_k^{-1}(y)$ is imparted to h (and possibly to m); this is by property P4′. Thus adversary A builds a set S_2 of size $q/2$ as follows: for $i \in [q/2+1..q]$, it sets $k^{(i)} = \langle i \rangle$, determines $y^{(i)}$ from $k^{(i)}$ and σ, queries $x^{(i)} \xleftarrow{\$} E^{-1}_{k^{(i)}}(y^{(i)})$, determines $m^{(i)}$ and $h^{(i)}$ from $k^{(i)}$ and $x^{(i)}$, and adds $(m^{(i)}, h^{(i)})$ to S_2. Since each E^{-1} query uses a distinct key (and distinct from those used in the first stage), the strings $x^{(i)}$ are uniformly and independently distributed. By our observation above, the $h^{(i)}$ are likewise distributed.

Now we let L_1 be the multiset consisting of the second components of the entries in S_1 (i.e., the $h^{(i)}$), and let L_2 be the multiset consisting of the second components of S_2 (again, the $h^{(i)}$). Then by Lemma 17 there exists an h such that $h \in L_1$ and $h \in L_2$ with probability at least $(1/2e)\,q^2/2^n$, and if such an h exists, then adversary A outputs preimage $M = (m_1, m_2)$ for z where $(m_1, h) \in S_1$ and $(m_2, h) \in S_2$. If no such h exists, adversary A outputs $M = m_1$ (or simply aborts). The bound claimed in the theorem statement follows. □

9. Fatal Attacks on Five of PGV's B-Labeled Schemes

In [30] there are a total of 13 schemes labeled as "backward attackable." We have already shown that eight of these, $H_{13}, \ldots, H_{20}$, are collision resistant. But the remaining

r	$f(h,m) =$	M_c
39	$E_w(w) \oplus m$	$h_0 \oplus c \;\|\; E_c(c) \oplus h_0$
40	$E_c(w) \oplus m$	$h_0 \oplus c \;\|\; E_c(c) \oplus h_0$
43	$E_w(w) \oplus h$	$h_0 \oplus c \;\|\; E_c(c) \oplus h_0 \oplus c$
55	$E_w(c) \oplus m$	$h_0 \oplus c \;\|\; E_c(c) \oplus h_0$
59	$E_w(c) \oplus h$	$h_0 \oplus c \;\|\; E_c(c) \oplus h_0 \oplus c$

Fig. 5. Messages that collide under the five weak "backward attackable" schemes in [30]. For any $c \in \{0,1\}^n$, $\hat{H}_r(M_c) = h_0$ for all five schemes listed.

five schemes are completely insecure; each can be broken with two queries. Consider, for example, $H = \hat{H}_{39}$, constructed by iterating the compression function $f = \hat{f}_{39}$ defined by $f^E(h_{i-1}, m_i) = E_{m_i \oplus h_{i-1}}(m_i \oplus h_{i-1}) \oplus m_i$. For any $c \in \{0,1\}^n$, the strings $(h_0 \oplus c) \,\|\, (E_c(c) \oplus h_0)$ hashes to h_0, and so it takes only two queries to produce a collision. Variants of this attack break the schemes $\hat{H}_{40}$, $\hat{H}_{43}$, $\hat{H}_{55}$, and $\hat{H}_{59}$. See Fig. 5 for the definition of these schemes and the corresponding collision-finding attacks.

10. Relating the Two Notions of Preimage Resistance

In general, the $\mathbf{Adv}^{\mathrm{pre1}}_H$ and $\mathbf{Adv}^{\mathrm{pre2}}_H$ measures can be far apart; it is easy to construct an example hash function to separate them. But for a "reasonable" hash function H, like those in group-1 and group-2, one might expect the two notions to be close, if not identical. In particular, it is natural to suspect that the random variable $H^E(M)$ is uniformly distributed in $\{0,1\}^n$ if $M \xleftarrow{\$} \{0,1\}^{n\ell}$ and $E \xleftarrow{\$} \mathrm{Bloc}(n)$. Interestingly, this is not true. For example, a computer-aided experiment shows that when $E \xleftarrow{\$} \mathrm{Bloc}(2)$ and $M \xleftarrow{\$} \{0,1\}^4$, the string $H^E_1(M)$ takes on the value 00 more than a quarter of the time (in fact, 31.25% of the time), while each of the remaining three possible outputs (01, 10, 11) occur less than a quarter of the time (each occurs $22.91\overline{6}$% of the time). Still, for group-1 and group-2 schemes, the two notions are close enough that $\mathbf{Adv}^{\mathrm{pre1}}_H$ makes a good surrogate for $\mathbf{Adv}^{\mathrm{pre2}}_H$. We note that the result is proved only in the ideal cipher setting. If one would consider the adversary's advantage for arbitrary E, the two notions can be far apart. For example, $f_1 = f(h,m) = E_h(m) \oplus m$ instantiated with the (disastrous) $E_k(x) = x$ for all k has $\mathbf{Adv}^{\mathrm{pre1}}_{H^n}(q) = 2^{-n}$ yet $\mathbf{Adv}^{\mathrm{pre2}}_{H^n}(q, \ell n) = 1$ for $q > 1$.

Lemma 19 (The "pre1" and "pre2" notions are close for group-1 and group-2 schemes). *Fix $n \geq 1$, let H be a rate-1 hash function whose corresponding compression function has properties P1 and P2, and let A be a preimage-finding adversary. Then* $|\mathbf{Adv}^{\mathrm{pre1}}_{H^n}(A) - \mathbf{Adv}^{\mathrm{pre2}}_{H^n}(A, \ell n)| \leq \ell / 2^{n-1}$.

Proof. The proof is a code-based game-playing argument [1,20]. Given that $f''(h, m, \cdot)$ is bijective for all $(h, m) \in \{0,1\}^{2n}$ (property P2), we write $y = f''^{-1}(h, m, z)$ to denote the (partial) inverse of f'', i.e., the unique y for which $z = f''(h, m, y)$. Let $h_0 \in \{0,1\}^n$

Game D: as written (include everything)
Game R: omit the framed statements at lines 09 and 11

procedure INITIALIZE
01 $h \leftarrow h_0$
02 **for** $i \leftarrow 1$ **to** ℓ **do**
03 $\quad m \xleftarrow{\$} \{0,1\}^n,\ (k,x) \leftarrow f'(h,m)$
04 $\quad$ **if** $i = \ell$ **then continue**
05 $\quad$ **if** $\pi(k,x) = \text{undefined}$ **then** $\pi(k,x) \xleftarrow{\$} \overline{\text{Range}}(\pi(k,\cdot))$
06 $\quad y \leftarrow \pi(k,x),\ h \leftarrow f''(h,m,y)$
07 $z \xleftarrow{\$} \{0,1\}^n,\ y \leftarrow f''^{-1}(h,m,z),\ h \leftarrow z$
08 **if** $\pi(k,x) \neq \text{undefined}$ **then**
09 $\quad bad \leftarrow \text{true},\ \boxed{y \leftarrow \pi(k,x),\ h \leftarrow f''(h,m,y)}$
10 **else if** there is an x' such that $\pi(k,x') = y$ **then**
11 $\quad bad \leftarrow \text{true},\ \boxed{y \xleftarrow{\$} \overline{\text{Range}}(\pi(x,\cdot)),\ h \leftarrow f''(h,m,y)}$
12 $\pi(k,x) \leftarrow y$
13 **return** $\sigma \leftarrow h$

procedure $E(k,x)$
20 **if** $\pi(k,x) \neq \text{undefined}$ **then** $y \leftarrow \pi(k,x)$ **else** $y \xleftarrow{\$} \overline{\text{Range}}(\pi(k,\cdot)),\ \pi(k,x) \leftarrow y$
21 **return** y

procedure $E^{-1}(k,y)$
30 **if** $\exists\, x'$ such that $\pi(k,x') = y$ **then** $x \leftarrow x'$ **else** $x \xleftarrow{\$} \overline{\text{Domain}}(\pi(k,\cdot)),\ \pi(k,x) \leftarrow y$;
31 **return** x

procedure FINALIZE$(m_1 m_2 \cdots m_{\ell'})$
40 $h \leftarrow h_0$
41 **for** $i \leftarrow 1$ **to** ℓ' **do**
42 $\quad m \leftarrow m_i,\ (k,x) \leftarrow f'(h,m),\ y \leftarrow \pi(k,x),\ h \leftarrow f''(h,m,y)$
43 **if** $h = \sigma$ **then return** 1 **else return** 0

Fig. 6. Games used in the proof of Lemma 19.

be the initial chaining vector associated with H. Consider Games D and R (for "domain" and "range") defined in Fig. 6. (Also see [1] for an explanation of the semantics.) Note first that the probability that adversary A outputs 1 in game R is $\mathbf{Adv}^{\mathrm{pre1}}_{H}(A)$, as the value h sampled at line 07 is always returned at line 13 (in addition, the permutation π is grown in the manner of a uniform random permutation, and the test in Finalize conforms with the notion of success associated to $\mathbf{Adv}^{\mathrm{pre1}}_{H}$). Similarly, the probability that A outputs 1 in game D is exactly $\mathbf{Adv}^{\mathrm{pre2}}_{H}(A, \ell n)$. Furthermore, Games D and R are identical-until-*bad*, meaning that they are syntactically identical until the sequels to lines that set the flag *bad* to true. Thus $|\mathbf{Adv}^{\mathrm{pre1}}_{H}(A) - \mathbf{Adv}^{\mathrm{pre2}}_{H}(A, \ell n)|$ is at most the probability that *bad* gets set to true in Game R. We now upperbound this quantity.

The value of (k, x) at line 08 is defined (line 03) by applying f' to a given h and a uniformly selected (line 03) value of m. Since f' is bijective (property P1), each h defines a set of 2^n possible pairs (k, x). The uniform selection of m implies a pair is chosen uniformly among these, and this is independent of the set of points at which π has already been defined. As a consequence, the flag *bad* will get set at line 09 with probability at most the number of points at which π is already defined, divided by 2^n. This is upper bounded by $(\ell - 1)/2^n$.

We must similarly bound the probability that *bad* gets set at line 11. At line 10 the value of y, which was defined at line 07, is uniform, due to z being uniform at line 07 and $f''(h, m, \cdot)$ being bijective for fixed (h, m) (property P2), and it is independent of the values that are currently in the range of π. So the probability that *bad* gets set at line 11 is again upper bounded by $(\ell - 1)/2^n$. Summing, the probability that *bad* gets set in the game is at most $(2\ell - 2)/2^n < 2\ell/2^n$. The result for H follows. □

11. Subsequent Work

Following the publication of the proceedings version of this paper in 2002 [6], the ideal-cipher model has been employed in a multitude of papers that explore security properties of blockcipher-based hash functions. Most directly related is work by Stam [40], which considers generalizations of PGV functions [40]. The paper shows, for example, that all 20 group-1 and -2 schemes can be generalized, preserving security, to the setting where the blockcipher has a key length unequal to the block length. Moreover, it shows that chopping the output provides the expected preimage and collision resistance given the new output length. Some of Stam's work [40] is now included here.

Most of the other related work is in one of the following directions: (1) looking at "double-length" constructions (for better security bounds); (2) exploiting the possibility of an iterated hash function being *more* secure than its underlying compression function; (3) considering the security of hash functions built from *non*-compressing primitives, such as an ideal cipher with a fixed key; (4) exploring properties beyond collision resistance and preimage resistance; and (5) investigating the foundations of the ideal-cipher model used here. We now describe each of these lines.

1. *Double-length constructions*. For double-length constructions, the hash function returns a value whose length is twice the block length of the underlying blockcipher, thus enabling collision resistance up to the block length of the blockcipher

(meaning that, asymptotically, for an n-bit blockcipher, some $\Omega(2^n)$ queries are needed to gain significant advantage). The first good bounds were shown by Hirose [18]. Using a blockcipher with key length twice the block length, he exhibits a rate-1/2 compression function (two blockcipher calls per block of message digested) that achieves essentially optimal collision resistance. Fleischmann et al. [15,16] likewise address the collision resistance of two double-length constructions by Lai and Massey [21] known as Abreast Davies–Meyer and Tandem Davies–Meyer. These constructions also have keys whose length is twice that of the blocks. Özen and Stam [28] address the security of a large class of double-length constructions with similarly long keys. Stam [40] gives a rate-1 compression function that is optimally collision resistant up to a logarithmic factor.

2. *Security through iteration*. The theme of building collision-resistant hash functions by iterating a weak compression function has surfaced a number of times in the recent literature. As we do in this work, the Özen–Stam paper establishes a class of constructions whose security can be proven only in the iteration. In a result of particular importance for practice, Steinberger [41] proves a strong security bound for the double-length construction known as MDC-2 [19]. A trivial bound on collision resistance (to $2^{n/2}$ queries) follows by the analyses given here. But by arguing about the iterated hash function directly, rather than indirectly via the compression function, Steinberger proves collision resistance to about $2^{3n/5}$ queries; concretely, when $n = 128$, he shows that about 2^{75} queries are necessary until the adversary has a good chance to find a collision.
3. *Noncompressing primitives*. Hash functions can be designed from fixed-key blockciphers (random permutations). The first systematic work in this direction is by Black, Cochran, and Shrimpton [7,8], who consider the general case of a $2n$-bit to n-bit compression function that makes a single call to an n-bit random permutation. They show that such a scheme *cannot* be collision resistant, even in the iteration. That said, permutation-based compression functions (slower than rate-1) were later provided by Rogaway and Steinberger [34], Shrimpton and Stam [37], and Stam [39]. For some of these constructions, Dodis and Steinberger [11] and then Lee and Steinberger [22] establish security results beyond collision resistance. Lowerbounds on what is achievable by permutation-based constructions of specified efficiency—that is, generic attacks—are provided by Rogaway and Steinberger [35], Stam [39], and Steinberger [42].
4. *Beyond collision/preimage resistance*. Useful security properties for hash functions go far beyond collision and preimage resistance. The original PGV paper [30] distinguished among our twelve group-1 schemes, regarding four of them (our $H_1, \ldots, H_4$) as better than the other eight (our $H_5, \ldots, H_{12}$) based on there being easily-found preimages in the compression functions associated to the latter group. While this is not by itself a defect, Duo and Li [13] consider second-preimage resistance and show that, in fact, the same eight schemes are worse in this regard.

   To capture these properties, and much more, one would like to show that a particular hash function delivers the functionalities of a random oracle. Towards this goal, Coron et al. [9] adapt the indifferentiability framework by Maurer, Renner, and Holenstein [25] to the setting of hash functions. This allows one to formally quantify the extent to which a hash function (based on some underlying

ideal primitive) behaves like a random oracle. Within this framework, they present compression function modes-of-operation that provably turn an ideal compression function (i.e., a fixed-input-length random oracle) into a hash function that is indifferentiable from a (variable-input-length) random oracle. Said another way, these modes are good domain extenders for fixed-input-length random oracles. This is in contrast to the Merkle–Damgård iteration, which is a good domain extender for collision-resistant compression functions but fails to turn a fixed-input-length random oracle into a variable-input-length one. (The well-known length-extension attack makes this failure obvious.) Coron et al. also show that their modes of operation retain indifferentiability when Davies–Meyers (based on an ideal cipher) is used in place of an ideal compression function. This holds despite the fact that the Davies–Meyer construction is *not* itself indifferentiable from an ideal compression function. (For example, it has fixed points.)

Even though Davies–Meyer fails to be indifferentiable from an ideal compression function, Dodis, Ristenpart, and Shrimpton [12] show that all group-1 PGV schemes (including Davies–Meyer) are *preimage-aware* in the ideal-cipher model. This is a property stronger than collision resistance. They also show that the group-2 schemes yield preimage-aware hash functions when MD-iterated.

5. *Investigating the ideal-cipher model*. Finally, some of the related work has directly addressed the use of the ideal-cipher model. Black [5] gave an (admittedly pathological) blockcipher-based hash function that is provably secure in the ideal-cipher model, and yet collisions can be efficiently found for any standard-model instantiation of the blockcipher. Hirose [17] shows that for each group-1 or -2 scheme, it is possible to construct a secure pseudorandom permutation (PRP) for which the scheme admits easy attacks. Simon had earlier provided a result pointing to the insufficiency of one-way functions for proving collision resistance [38]. Biryukov, Khovratovich, and Nikolić [4] attack the Davies–Meyer construction (scheme H_5) when instantiated with AES, thereby concretizing the gap between the ideal-cipher model and real-world instantiations.

We conclude by noting that many of the recent NIST SHA-3 competition entrants are blockcipher or permutation-based and that the authors routinely provide a security analysis in the ideal-cipher model, either for the compression function or in the iteration. A good example here is the sponge construction [3], which is permutation-based and achieves its security properties only in the iteration (collisions can be found in the compression function in a constant number of queries). The approach presented in this paper has, in short, become routine.

Acknowledgements

The authors gratefully acknowledge the helpful and insightful comments of the anonymous referees.

John Black received support from NSF CAREER award CCR-0240000; Phil Rogaway and Tom Shrimpton received support from NSF award CCR-0085961 and a gift from CISCO Systems. For the journal revision, Phil received additional support from NSF award CNS-0904380 and Tom from CNS-0627752 and

NSF CAREER award CNS-845610. The work described in this paper has been further supported by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II.

References

[1] M. Bellare, P. Rogaway, The security of triple encryption and a framework for code-based game-playing proofs, in *Advances in Cryptology—Proceedings of EUROCRYPT 2006*. Lecture Notes in Computer Science, vol. 4004 (Springer, Berlin, 2006), pp. 409–426

[2] M. Bellare, J. Kilian, P. Rogaway, The security of cipher block chaining message authentication code. *J. Comput. Syst. Sci.* **61**(3), 362–399 (2000)

[3] G. Bertoni, J. Daemen, M. Peeters, G. Assche, On the indifferentiability of the sponge construction, in *Advances in Cryptology—Proceedings of EUROCRYPT 2008*. Lecture Notes in Computer Science, vol. 4965 (Springer, Berlin, 2008), pp. 181–197

[4] A. Biryukov, D. Khovratovich, I. Nikolić, Distinguisher and related-key attack on the full AES-256, in *Advances in Cryptology—Proceedings of CRYPTO 2009*. Lecture Notes in Computer Science, vol. 5677 (Springer, Berlin, 2009), pp. 229–247

[5] J. Black, The ideal-cipher model, revisited: an uninstantiable blockcipher-based hash function, in *Fast Software Encryption, 13th International Workshop, FSE 2006*. Lecture Notes in Computer Science, vol. 4047 (Springer, Berlin, 2006), pp. 328–340

[6] J. Black, P. Rogaway, T. Shrimpton, Black-box analysis of the block-cipher-based hash-function constructions from PGV, in *Advances in Cryptology—CRYPTO 2002*. Lecture Notes in Computer Science, vol. 2442 (Springer, Berlin, 2002), pp. 320–335. Proceedings version of this paper

[7] J. Black, M. Cochran, T. Shrimpton, On the impossibility of highly efficient blockcipher-based hash functions, in *Advances in Cryptology—EUROCRYPT 2005*. Lecture Notes in Computer Science, vol. 3494 (Springer, Berlin, 2005), pp. 526–541

[8] J. Black, M. Cochran, T. Shrimpton, On the impossibility of highly-efficient blockcipher-based hash functions. *J. Cryptol.* **22**(3), 311–329 (2009)

[9] J. Coron, Y. Dodis, C. Malinaud, P. Puniya, Merkle-Damgård revisited: how to construct a hash function, in *Advances in Cryptology—CRYPTO 2005*. Lecture Notes in Computer Science, vol. 3621 (Springer, Berlin, 2005), pp. 430–448

[10] I. Damgård, A design principle for hash functions, in *Advances in Cryptology—CRYPTO 1989*. Lecture Notes in Computer Science, vol. 435 (Springer, Berlin, 1990), pp. 416–427

[11] Y. Dodis, J. Steinberger, Message authentication codes from unpredictable block ciphers, in *Advances in Cryptology—Proceedings of CRYPTO 2009*. Lecture Notes in Computer Science, vol. 5677 (Springer, Berlin, 2009), pp. 267–285

[12] Y. Dodis, T. Ristenpart, T. Shrimpton, Salvaging Merkle–Damgård for practical applications, in *Advances in Cryptology—Proceedings of EUROCRYPT 2009*. Lecture Notes in Computer Science, vol. 5479 (Springer, Berlin, 2009), pp. 371–388

[13] L. Duo, C. Li, Improved collision and preimage resistance bounds on PGV schemes. Technical Report 2006/462, IACR's ePrint Archive, 2006

[14] S. Even, Y. Mansour, A construction of a cipher from a single pseudorandom permutation, in *Advances in Cryptology—ASIACRYPT 1991*. Lecture Notes in Computer Science, vol. 739 (Springer, Berlin, 1992), pp. 210–224

[15] E. Fleischmann, M. Gorski, S. Lucks, On the security of tandem-DM, in *Fast Software Encryption, 16th International Workshop, FSE 2009*. Lecture Notes in Computer Science, vol. 5665 (Springer, Berlin, 2009), pp. 84–103

[16] E. Fleischmann, M. Gorski, S. Lucks, Security of cyclic double block length hash functions, in *Cryptography and Coding, 12th IMA International Conference, Cryptography and Coding 2009*. Lecture Notes in Computer Science, vol. 5921 (Springer, Berlin, 2009), pp. 153–175

[17] S. Hirose, Secure block ciphers are not sufficient for one-way hash functions in the Preneel-Govaerts-Vandewalle model, in *Selected Areas in Cryptography 2002*. Lecture Notes in Computer Science, vol. 2595 (Springer, Berlin, 2003), pp. 339–352

[18] S. Hirose, Provably secure double-block-length hash functions in a black-box model, in *Information Security and Cryptology—ICISC 2004*. Lecture Notes in Computer Science, vol. 3506 (Springer, Berlin, 2005), pp. 330–342
[19] ISO/IEC 10118-2. Information technology—Security techniques—Hash functions—Hash functions using an n-bit block cipher algorithm. International Organization for Standardization, Geneva, Switzerland, 1994
[20] J. Kilian, P. Rogaway, How to protect DES against exhaustive key search. *J. Cryptol.* **14**(1), 17–35 (2001). Earlier version in *CRYPTO 1996*
[21] X. Lai, J. Massey, Hash function based on block ciphers, in *Advances in Cryptology—Proceedings of EUROCRYPT 1992*. Lecture Notes in Computer Science, vol. 658 (Springer, Berlin, 1992), pp. 55–70
[22] J. Lee, J. Steinberger, Multi-property-preserving domain extension using polynomial-based modes of operation, in *Advances in Cryptology—Proceedings of EUROCRYPT 2010*. Lecture Notes in Computer Science (Springer, Berlin, 2010)
[23] S. Lucks, A collision-resistant rate-1 double-block-length hash function, in *Symmetric Cryptography, Dagstuhl Seminar Proceedings*, no. 07021, Dagstuhl, Germany, 2007. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany
[24] S. Matyas, C. Meyer, J. Oseas, Generating strong one-way functions with cryptographic algorithms. *IBM Tech. Dis. Bull.* **27**(10a), 5658–5659 (1985)
[25] U. Maurer, R. Renner, C. Holenstein, Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology, in *Theory of Cryptography Conference (TCC '04)*. Lecture Notes in Computer Science, vol. 2951 (Springer, Berlin, 2004), pp. 21–39
[26] A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography* (CRC Press, Boca Raton, 1996)
[27] R. Merkle, One way hash functions and DES, in *Advances in Cryptology—CRYPTO 1989*. Lecture Notes in Computer Science, vol. 435 (Springer, Berlin, 1990), pp. 428–446
[28] O. Özen, M. Stam, Another glance at double-length hashing, in *Cryptography and Coding, 12th IMA International Conference, Cryptography and Coding 2009*. Lecture Notes in Computer Science, vol. 5921 (Springer, Berlin, 2009), pp. 176–201
[29] B. Preneel, Analysis and design of hash functions. PhD thesis, Katholike Universiteit Leuven (Belgium), 1993. Available from Preneel's homepage
[30] B. Preneel, R. Govaerts, J. Vandewalle, Hash functions based on block ciphers: a synthetic approach, in *Advances in Cryptology—Proceedings of CRYPTO 1993*. Lecture Notes in Computer Science, vol. 773 (Springer, Berlin, 1994), pp. 368–378
[31] M. Rabin, Digitalized signatures, in *Foundations of Secure Computation* (Academic Press, New York, 1978), pp. 155–168
[32] R. Rivest, The MD4 message digest algorithm, in *Advances in Cryptology—Proceedings of CRYPTO 1900*. Lecture Notes in Computer Science, vol. 2442 (Springer, Berlin, 1991), pp. 303–311
[33] P. Rogaway, T. Shrimpton, Cryptographic hash-function basics: definitions, implications and separations for preimage resistance, second-preimage resistance, and collision resistance, in *Fast Software Encryption, 11th International Workshop, FSE 2004*. Lecture Notes in Computer Science (Springer, Berlin, 2004), pp. 371–388
[34] P. Rogaway, J. Steinberger, Constructing cryptographic hash functions from fixed-key blockciphers, in *Advances in Cryptology—Proceedings of CRYPTO 2008*. Lecture Notes in Computer Science, vol. 5157 (Springer, Berlin, 2008), pp. 433–450
[35] P. Rogaway, J. Steinberger, Security/efficiency tradeoffs for permutation-based hashing, in *Advances in Cryptology—Proceedings of EUROCRYPT 2008*. Lecture Notes in Computer Science, vol. 4965 (Springer, Berlin, 2008), pp. 220–236
[36] C. Shannon, Communication theory of secrecy systems. *Bell Syst. Tech. J.* **28**(4), 656–715 (1949)
[37] T. Shrimpton, M. Stam, Building a collision-resistant compression function from non-compressing primitives, in *ICALP 2008, Part II*, vol. 5126 (Springer, Berlin, 2008), pp. 643–654
[38] D. Simon, Finding collisions on a one-way street: can secure hash functions be based on general assumptions? in *Advances in Cryptology—Proceedings of EUROCRYPT 1998*, vol. 1403. Lecture Notes in Computer Science (Springer, Berlin, 1998), pp. 334–345
[39] M. Stam, Beyond uniformity: better security/efficiency tradeoffs for compression functions, in *Advances in Cryptology—Proceedings of CRYPTO 2008*. Lecture Notes in Computer Science, vol. 5157 (Springer, Berlin, 2008), pp. 397–412

[40] M. Stam, Block cipher based hashing revisited, in *Fast Software Encryption 2009*. Lecture Notes in Computer Science, vol. 5665 (Springer, Berlin, 2009), pp. 67–83

[41] J. Steinberger, The collision intractability of MDC-2 in the ideal-cipher model, in *Advances in Cryptology—Proceedings of EUROCRYPT 2007*. Lecture Notes in Computer Science, vol. 4515 (Springer, Berlin, 2007), pp. 34–51

[42] J. Steinberger, Stam's collision resistance conjecture, in *Advances in Cryptology—Proceedings of EUROCRYPT 2010*. Lecture Notes in Computer Science, vol. 6110 (Springer, Berlin, 2010), pp. 597–615

[43] R. Winternitz, A secure one-way hash function built from DES, in *Proceedings of the IEEE Symposium on Information Security and Privacy* (IEEE Press, New York, 1984), pp. 88–90