Faster Multicollisions*

Jean-Philippe Aumasson

FHNW, Windisch, Switzerland

Abstract. Joux's multicollision attack is one of the most striking results on hash functions and also one of the simplest: it computes a k-collision on iterated hashes in time $\lceil \log_2 k \rceil \cdot 2^{n/2}$, whereas $k!^{1/k} \cdot 2^{n(k-1)/k}$ was thought to be optimal. Kelsey and Schneier improved this to $3 \cdot 2^{n/2}$ if storage $2^{n/2}$ is available and if the compression functions admits easily found fixed-points. This paper presents a simple technique that reduces this cost to $2^{n/2}$ and negligible memory, when the IV can be chosen by the attacker. Additional benefits are shorter messages than the Kelsey/Schneier attack and cost-optimality.

Keywords: hash function, collision.

1 Introduction

Cryptographic hash functions are key ingredients in numerous schemes like public-key encryption, digital signatures, message-authentication codes, or multiparty functionalities. The last past years the focus on hash functions has dramatically increased, because of new attacks on the compression algorithm of MD5 and SHA-1 and on their high-level structure, e.g. *multicollision attacks*. We introduce these attacks below.

Consider an arbitrary function $f : \{0,1\}^n \times \{0,1\}^m \mapsto \{0,1\}^n$. A classic construction [23, 24] defines the *iterated hash* of f as the function

$h_{H_0}(M_1 \dots M_\ell)$:
  for $i = 1, \dots, \ell$ **do**
    $H_i \leftarrow f(H_{i-1}, M_i)$
  return H_ℓ

where H_0 is called the *initial value* (IV), and f the *compression function*. Damgård and Merkle [6, 17] independently proved in 1989 that h is collision-resistant if f is collision-resistant when the bitlength of the message is appended at its end (a technique referred as *MD-strengthening*). This technique also prevents the *fixed-point attack*—a folklore multicollision attack—whose basic idea is that if M satisfies $f(H_0, M) = H_0$, then $h_{H_0}(M \dots M) = H_0$.

The problem we will focus on is how quickly one can compute k distinct messages mapping by h_{H_0} to the same value, when MD-strengthening is applied (call

*Article previously accepted to SECRYPT 2008, but withdrawn by the author because unable to attend the conference. This author was supported by the Swiss National Science Foundation under project no. 113329.

this a k-*collision*). An extension of the birthday attack computes k-collisions[1] within about $k!^{1/k} \cdot 2^{n(k-1)/k}$ calls to f, which was believed to be the optimal until the technique of [9] that requires only $\lceil \log_2 k \rceil \cdot 2^{n/2}$ f-calls. Kelsey and Schneier subsequently reduced this cost to $3 \cdot 2^{n/2}$ [11], provided that storage $2^{n/2}$ is available, and that f admits easily found fixed-points. Though seldom cited, this technique is more powerful than Joux's in the sense that the cost of finding a k-multicollision is independent of k, yet a drawback is the length of the colliding messages, significantly larger.

1.1 Contribution

This paper reviews the previous techniques for computing k-collisions, and presents a novel method whose main features are

- a cost independent of the number of colliding messages k (with $2^{n/2}$ trials)
- short colliding messages (with $\lceil \log_2 k \rceil$ blocks)
- negligible storage requirements

Limitations of the attack are the need for easily found fixed-points, and the IV chosen by the attacker. This means that the IV used for the multicollisions cannot be set to a predefined value, which corresponds to the model called "semi-free-start collisions" in [13], "collision with different IV" in [20], and "collision (random IV)" in [16]. Within this model, our technique is optimal, because k-collisions become as expensive as collisions.

The practical impact of this attack is limited, because it does not break the complexity barrier $2^{n/2}$. However, in terms of price/performance ratio (or "value" [20, §2.5.1]) it outperforms all the previous attacks, since for the same price as a collision, one gets k-collisions.

1.2 Related Work

Multicollisions received a steady amount of attention since Joux's attack: [18, 8] generalized them to constructions where a message block can be used multiple times; [29] revisited the birthday attack for multicollision; dedicated multicollision attacks were found for MD2 [12] and MD4 and HAVAL [30]. Finally, [10] used multicollisions for the "Nostradamus attack".

1.3 Notations

Let $f : \{0,1\}^n \times \{0,1\}^m \mapsto \{0,1\}^n$ be the compression function of the iterated hash h_{H_0}, for an arbitrary H_0, where MD-strengthening is applied. If f admits easily found fixed-points, write $\mathsf{FP}_f : \{0,1\}^m \mapsto \{0,1\}^n$ a function such that for all M, $\mathsf{FP}_f(M)$ is a fixed-point for f, i.e. $f(\mathsf{FP}_f(M), M) = \mathsf{FP}_f(M)$.

[1] Plural is used because from any k-collision we can derive many other k-collisions, by appending the same arbitrary data at the end of colliding messages.

Then, fix a unit of *time* (e.g. an integer addition, a call to f, a MIPS-year, etc.), and a unit of *space* (e.g. a bit, a 32-bit word, a n-bit chaining value, a 128 Gb hard drive, etc.), and write the cost of computing f as $\mathbf{T}_f$ time units and $\mathbf{S}_f$ space units (resp. $\mathbf{T}_{\mathsf{FP}}$ and $\mathbf{S}_{\mathsf{FP}}$ for FP_f); we assume these costs input-independent; we disregard the extra cost of auxiliary operations and memory accesses (though of certain practical relevance); we also disregard the constant factor caused by "memoryless" birthday attacks [28, 22].

Note that our goal is to find (the description of) many messages with same digest, not to effectively construct them. Hence, the time cost of finding a k-collision is not lower-bounded by k (e.g. k steps of a Turing machine), neither are the space requirements.

2 Joux Multicollisions

This method computes 2^k-collisions for k times the cost of finding a single collision: Assuming $m < n$, first compute a colliding pair (M_1, M_1'), i.e. such that $f(H_0, M_1) = f(H_0, M_1') = H_1$, then compute a second colliding pair (M_2, M_2') such that $f(H_1, M_2) = f(H_1, M_2') = H_2$, and so on until (M_k, M_k') with H_{k-1} as IV. Hence, for a symbol $X \in \{M, M'\}$, any of the 2^k messages of the form $X_1 \dots X_k$ has intermediate hash values $H_1, \dots, H_k$, and 2^k-collisions can be derived from these 2^k messages by appending extra blocks with correct padding. The cost of the operations above is time $k \cdot 2^{n/2} \cdot \mathbf{T}_f$, and negligible space.

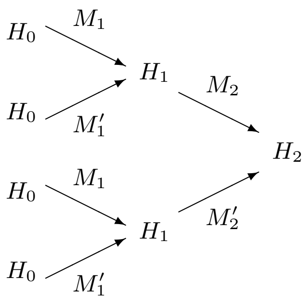

Fig. 1. Illustration of Joux's method for $k = 2$: first a collision $f(H_0, M_1) = f(H_0, M_1') = H_1$ is computed, then a second collision $f(H_1, M_2) = f(H_1, M_2') = H_2$ is found; the 4 colliding messages are M_1M_2, M_1M_2', $M_1'M_2$, and $M_1'M_2'$.

Fig. 1 gives an intuitive presentation of the attack; computing a 2^k-collision can be seen as the bottom-up construction of a binary tree, where each collision increases by one the tree depth. Note that a chosen IV does not help the attacker.

3 Kelsey/Schneier Multicollisions

As an aside in their paper on second-preimages, Kelsey and Schneier reported a method for computing k-collisions when f admits fixed-points [11, §5.1]; an advantage over Joux's attack is that the cost no longer depends on k. Here we will detail this result, which benefited of only a few informal lines in [11], and is seldom refered in literature.

3.1 Fixed-Points

A *fixed-point* for a compression function f is a pair (H, M) such that $f(H, M) = H$. For a random f finding a fixed-point requires about 2^n trials, by brute force search. Because it does not represent a security threat *per se*, neither it helps to find preimages or collisions, that property has not been perceived as an undesirable attribute: in 1993, Preneel, Govaerts and Vandewalle considered that "this attack is not very dangerous" [21], and according to Schneier in 1996, this "is not really worth worrying about" [27, p.448]; the HAC is more prudent, writing "Such attacks are of concern if it can be arranged that the chaining variable has a value for which a fixed point is known" [16, §9.102.(iii)].

The typical example is the Davies-Meyer construction for blockcipher-based compression functions, which sets $f(H, M) = E_M(H) \oplus H$. Hence, for any M a fixed point is $(E_M^{-1}(0), M)$:

$$E_M(E_M^{-1}(0)) \oplus E_M^{-1}(0) = 0 \oplus E_M^{-1}(0) = H.$$

Therefore, each message block M has a unique H that gives $f(H, M) = H$ and that is trivial to compute[2].

Note that the functions MD4/5 and SHA-0/1/2 all implicitly follow a Davies-Meyer scheme (where integer addition replaces XOR). More generally, an iterated hash may admit fixed-points for a sequences of compressions rather than a single compression—e.g. for two compressions, defining $f'(H, M, M') = f(f(H, M), M')$. Generic multicollision attacks apply as well to this type of function, up to a redefinition of f and m.

3.2 Basic Strategy

We first consider the simplest case, i.e. when any IV is allowed. Recall the fixed-point attack mentioned in §1, which exploits a fixed-point $f(H, M) = H$ to build the multicollision $h_H(M) = h_H(MM) = h_H(MMM \dots M) = H$. MD-strengthening protects against this attack, since it forces the last blocks of the messages to be distinct. The idea behind Kelsey/Schneier multicollisions is to bypass MD-strengthening using a *second fixed-point*. This fixed-point will be used to adjust the length of all messages to a similar value, to get the same padding data in all messages. Fig. 2 illustrates this attack: fix $n > 2$; if the first fixed-point is repeated k times, then the second fixed-point is repeated $n - k$ times to have n blocks in total. The last block imposed by MD-strengthening will thus be the same for all messages. Assuming one exploits the fixed-point $f(H_0, M_0) = H_0$, the second fixed-point is integrated via a meet-in-the-middle technique (MITM) that goes as follows:

[2]Similar fixed-points can be found for the constructions numbered 5 to 12 in [21].

1. Compute a list L_1:

$$(M_1, f(H_0, M_1)), \ldots, (M_{2^{n/2}}, f(H_0, M_{2^{n/2}})).$$

2. Compute a list L_2:

$$(M'_1, \mathsf{FP}(M'_1)), \ldots, (M'_{2^{n/2}}, \mathsf{FP}(M_{2^{n/2}})).$$

3. Look for a collision on the second pair element $(M_i, H_j) \in L_1$, $(M'_j, H_j) \in L_2$.
4. Construct colliding messages of the form $M_0 \ldots M_0 M_i M'_j \ldots M'_j$, such that the length of the whole message is kept constant.

The attack runs in time $2^{n/2} \cdot \mathbf{T}_f + 2^{n/2} \cdot \mathbf{T}_{\mathsf{FP}}$, and needs storage $\mathbf{S}_f + \mathbf{S}_{\mathsf{FP}} + 2^{n/2} \cdot \mathbf{S}_{(n+m)}$, with $\mathbf{S}_{(n+m)}$ the space used to store a $(n+m)$-bit string. These values are independent of the size of the multicollision. The length of messages is addressed later.

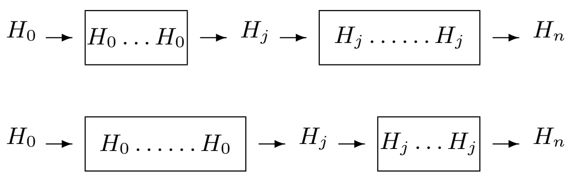

Fig. 2. Schematic view of the Kelsey/Schneier multicollision attack, for an IV chosen by the attacker: a first fixed-point allows to expand the message, while a second one adjust the lengths to a similar value.

When the IV is restricted to a specific value, the first fixed-point has to be introduced with another MITM; time cost grows to $2 \cdot 2^{n/2} \cdot \mathbf{T}_f + 2^{n/2} \cdot \mathbf{T}_{\mathsf{FP}}$, and storage is similar (the second MITM reuses the space allocated for the first one).

3.3 Multiple Fixed-Points and Message Length

In the above attack, a k-collision contains messages of about k blocks. In comparison, Joux's method produces messages of $\lceil \log_2 k \rceil$ blocks. This gap can be reduced by using more than two fixed-points: Assume that $K > 2$ fixed-points are integrated in the message. The attack now runs in time $(K-1)(2^{n/2} \cdot \mathbf{T}_f + 2^{n/2} \cdot \mathbf{T}_{\mathsf{FP}})$, counting $(K-1)$ MITM's, for a chosen IV. Also suppose a limit of ℓ blocks per message (e.g. a maximum number of blocks allowed by a design, typically 2^{64}), with $\ell > 2K$.

Given the limit ℓ, how large can be a multicollision in terms of K? The number of constructible colliding messages is equal to the number of *compositions* of ℓ having at most K non-null summands[3]. The number we are looking for is

[3] A composition (or ordered partition) of a number is a way of writing it as an ordered sum of positive integers. For example, 3 admits four compositions: 3, $2+1$, $1+2$, $1+1+1$.

$\mathcal{C}_{\ell,K} = \sum_{i=0}^{K-1} \binom{\ell}{i}$ (summing over the number of separators), so we will get a $\mathcal{C}_{\ell,K}$-collision.

For example, consider SHA-256, which admits fixed-points: with $K = 8$ one finds 2^{57}-collisions in time about $14 \cdot 2^{128}$, with 1024-block messages; in comparison Joux's method computes 2^{57}-collisions in time about $57 \cdot 2^{128}$, with 57-block messages, and if we fix the message length to 1024 it finds 2^{1024}-collisions, in time about $1024 \cdot 2^{128}$. This stresses that a small number of fixed-points leads to much longer messages. Performance becomes similar for the two attacks (in terms of time cost, message length, and k) when $K = \lfloor \ell/2 \rfloor$.

4 Faster Multicollisions

This section presents a method applicable when the compression function admits easily found fixed-points (like MD5, SHA-1, SHA-256), and when the IV can be chosen by the attacker. Despite its relative simplicity it has not mentioned in the literature, as far as we know.

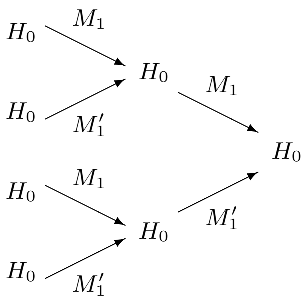

Fig. 3. Illustration of our technique for $k = 2$: a fixed-point collision $f(H_0, M_1) = f(H_0, M_1') = H_0$ is computed, then the four colliding messages are M_1M_1, M_1M_1', $M_1'M_1$, and $M_1'M_1'$. Contrary to Joux's attack, H_0 is here chosen by the attacker.

4.1 Description

The key idea of the attack is that of *fixed-point collision*, i.e. a collision for the function FP_f; since FP_f outputs n-bit this costs time $\mathbf{T}_{\mathsf{FP}} \cdot 2^{n/2}$ and space $\mathbf{S}_{\mathsf{FP}}$. A fixed-point collision is a pair (M, M') such that $\mathsf{FP}_f(M) = \mathsf{FP}_f(M') = H_0$, and thus $f(H_0, M) = f(H_0, M') = H_0$. The distribution of H_0 (as a random variable) depends on f and FP_f; e.g. for Davies-Meyer schemes based on a pseudoranom permutation (PRP), this will be uniform.

Once found a fixed-point collision (M, M'), a 2^k-collision can be constructed by considering all the k-block sequences in the set $\{M, M'\}^k$ followed by an arbitrary sequence of blocks $M^\star$ with convenient padding. For example, a 4-collision will be

$$H_0 \xrightarrow{M} H_0 \xrightarrow{M} H_0 \xrightarrow{M^\star} H$$
$$H_0 \xrightarrow{M} H_0 \xrightarrow{M'} H_0 \xrightarrow{M^\star} H$$
$$H_0 \xrightarrow{M'} H_0 \xrightarrow{M} H_0 \xrightarrow{M^\star} H$$
$$H_0 \xrightarrow{M'} H_0 \xrightarrow{M'} H_0 \xrightarrow{M^\star} H$$

The sole significant computation is for finding a fixed-point collision, hence the whole attack costs time $\mathbf{T}_{\mathsf{FP}} \cdot 2^{n/2}$ and memory $\mathbf{S}_{\mathsf{FP}}$ (with negligible overhead). For instance, for a Davies-Meyer function computing FP_f has the same cost as computing f, thus time cost is $\mathbf{T}_f \cdot 2^{n/2}$. Observe that the attack requires no call to the compression function itself, but just to the derived function FP_f.

If computing fixed-points is nontrivial but easier than expected, this attack becomes more efficient than Joux's as soon as $k > \mathbf{T}_{\mathsf{FP}}/\mathbf{T}_f$ (for computing 2^k-collisions).

4.2 Finding Fixed-Point Collisions

For a PRP-based Davies-Meyer compression function, the cost of finding a fixed-point collision (i.e. $\mathsf{FP}_f(M) = \mathsf{FP}_f(M')$) equals the cost of finding a collision (i.e. $f(H_0, M) = f(H_0, M')$); indeed in both cases the function is essentially one query to the PRP, thus the same refined birthday-based methods can be used [28, 22].

This suggests that for Davies-Meyer functions (like MD5, SHA-1, SHA-256) finding a fixed-point collision is cost-equivalent to finding a collision: indeed the goal is now to find (M, M') such that $E_M^{-1}(0) = E_{M'}^{-1}(0)$, while classical collisions need $E_M(H) = E_{M'}(H)$. Therefore, if E is a PRP then finding a fixed-point collision with fixed IV is exactly as hard a finding a collision.

For hash functions that don't have obvious fixed-points, finding a fixed-point collision is at least as hard as finding a collision. Contrary to Davies-Meyer schemes, the ability to find fixed-IV collisions does not directly allow to find fixed-point collisions.

The statements above cover other blockcipher-based schemes that allow the easy finding of fixed-points (cf. the 8 schemes in [21]). We conjecture that known techniques for finding collisions on MD5 and SHA-1 can be adapted to find fixed-point collisions within similar complexity.

4.3 Distinct-Length Multicollisions

The attacks of Joux and Kelsey/Schneier find colliding messages of same length. A variant of our technique allows to find sets of messages that collide and do not all have the same block length. The idea is to find a fixed-point collision $f(H, M) = f(H, M') = H$ such that M and M' contain valid padding bits, that is, are of the form $\ldots 10\ldots 0\|\ell$. The chosen message bitlength ℓ should be different for M and M', and be consistent with the number of zeros added. Finding a fixed-point collision with these restrictions is not more expensive than in the general case as soon as at least $n/2$ bits in the message blocks are not padding bits.

Once a pair (M, M') with the above conditions is found, we can directly describe multicollisions. Suppose for example that $M = \ldots 10\ldots 0\|\ell$ and $M' = \ldots 10\ldots 0\|\ell'$, where ℓ encodes the length of a 2-block message, and ℓ' encodes the length of a 3-block message. Then the messages $M\|M$, $M'\|M$, $M\|M\|M'$,..., $M'\|M'\|M'$ all have the same hash value by h_H, and have suitable message length encoding.

4.4 Comparison to Joux and Kelsey/Schneier

Compared to Joux's technique, ours has the advantage of a cost independent of k; optimality of the algorithm follows (with respect to the assumption that a single collision costs at least $2^{n/2}$ f-calls). Compared to Kelsey/Schneier, our technique benefits of short messages ($\lceil \log_2 k \rceil$ for a k-collision), and no storage requirement. However, our attack is limited by the chosen IV, which makes it irrelevant for many applications of hash functions.

Consider for example an attacker with $2^{130} \cdot \mathbf{T}_f$ power to attack SHA-256: with Joux's technique he finds 4-collisions, with Kelsey/Schneier's he finds k-collisions with k-block messages if memory $2^{128} \cdot \mathbf{S}_{(768)}$ is available, and with our method he finds k-collisions of length $\lceil \log_2 k \rceil$ for 4 different IV's, for any k.

4.5 Application to Concatenated Hash Functions

Let the hash function $\mathcal{H}(M) = h_{H_0}(M)\|h'_{H'_0}(M)$, where h is an iterated hash whose compression function f admits fixed-points, and h' and ideal hash function (in practice, h and h' might be the same function, and use different IV's). Suppose further that both hash to n-bit digests.

A basic birthday attack finds collisions on $\mathcal{H}$ within 2^n calls to h, and as many to h'; Joux reduced this cost to $n/2 \cdot 2^{n/2} \cdot \mathbf{T}_f + 2^{n/2} \cdot \mathbf{T}_{h'}$. Our multicollision technique applies similarly, if the IV of h can be chosen by the attacker: first compute a $2^{n/2}$-collision for h, in time $2^{n/2} \cdot \mathbf{T}_{\mathsf{FP}}$, then look for a collision on h' among these messages, in time $2^{n/2} \cdot \mathbf{T}_{h'}$. Assuming $\mathbf{T}_{h'} = \mathbf{T}_f$, we get an overall cost $2^{n/2+1} \cdot \mathbf{T}_f$, instead of $(n+1) \cdot 2^{n/2} \cdot \mathbf{T}_f$ with Joux's technique. Our method is almost optimal, since it almost reaches the cost of computing a collision on h or h' (up to a factor 2).

4.6 Countermeasures

The foremost question is "do we really need countermeasures?" A pragmatic answer would be negative, arguing that the barrier $2^{n/2}$ remains intact thus the security level is not reduced; however, from a price/performance perspective, security is clearly damaged. So if cheap countermeasures exist there seems to be really few reasons to ignore them.

The first obvious measure against our attacks and Kelsey/Schneier's is to avoid easy-to-find fixed-points. For example by using one of the four blockcipher-based constructions in [21] that have no fixed-points. Another choice is to "dither" the hash function, i.e. adding a stage-dependent input to the compression function, cf. $[2, 25, 5, 11, 1, 4]$). For example by adding a counter to the input of f, such that $H_i = f(H_{i-1}, M_i, i)$. Dithering however doesn't protect against Joux's method, since this computes a new collision for every dither value.

Joux's attack can be prevented by a technique like the "wide-pipe" and "double-pipe" of [14] or the similar chop-MD [5] construction, which enlarge the chain values compared to the hash value. This trick also makes our attack unapplicable, because it increases the cost of finding fixed-point collisions. Kelsey/Schneier attacks are applicable when fixed-points are easily found.

Another construction proposed in [15] prevents from all multicollision attacks presented here, including ours. Generally, our attack will work for some hash construction when both Joux's and Kelsey/Schneier do, hence won't work when at least one does not apply.

A construction published in Dean's thesis [7, §5.6.3, credited to Lipton] consists in hashing M as $\tilde{M}\|\tilde{M}$, with $\tilde{M}$ the padded message, to simulate a "variable IV". This prevents all nontrivial multicollision attacks, but is unreasonably inefficient.

5 Conclusions

We presented a multicollision attack applicable to iterated hashes when the IV can be chosen by the attacker, and when fixed-points for the compression function are easy to find. This can be seen as a variant of Joux's attack when some restrictions are put on the hash function (Joux's attack works for any IV and doesn't need fixed-points).

Our attack leaves open two related issues:

1. Can we find other generic attacks on iterated hashes that exploit easily-found fixed-points?
2. How to find fixed-point collisions for dedicated hash functions?

Current known generic attacks using fixed-points are those of Dean for second-preimages [7, 5.3.1], Kelsey/Schneier for multicollision [11], and ours in this paper. Fixed-point collisions are likely to be found using similar techniques as collisions, for blockcipher-based functions. Positive results to those two issues would lead to new generic attacks (finding collisions or preimages) and new dedicated attacks (finding fixed-points).

Acknowledgements

I wish to thank the referees of ICALP 2008 for many helpful comments, and John Kelsey for suggesting the attack of §4.3.

References

1. Elena Andreeva, Charles Bouillaguet, Pierre-Alain Fouque, Jonathan J. Hoch, John Kelsey, Adi Shamir, and Sébastien Zimmer. Second preimage attacks on dithered hash functions. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *LNCS*, pages 270–288. Springer, 2008.
2. Jean-Philippe Aumasson and Raphael C.-W. Phan. How (not) to efficiently dither blockcipher-based hash functions? In Serge Vaudenay, editor, *AFRICACRYPT*, volume 5023 of *LNCS*, pages 308–324. Springer, 2008.
3. Eli Biham and Orr Dunkelman. A framework for iterative hash functions - HAIFA. Second NIST Cryptographic Hash Workshop, 2006.
4. Eli Biham and Orr Dunkelman. A framework for iterative hash functions - HAIFA. Cryptology ePrint Archive, Report 2007/278, 2007. Extended version of [3].
5. Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård revisited: How to construct a hash function. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *LNCS*, pages 430–448. Springer, 2005.
6. Ivan Damgård. A design principle for hash functions. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *LNCS*, pages 416–427. Springer, 1989.
7. Richard Drews Dean. *Formal Aspects of Mobile Code Security*. PhD thesis, Princeton University, 1999.
8. Jonathan Hoch and Adi Shamir. Breaking the ICE - finding multicollisions in iterated concatenated and expanded (ICE) hash functions. In Matthew J. B. Robshaw, editor, *FSE*, volume 4047 of *LNCS*, pages 179–194. Springer, 2006.
9. Antoine Joux. Multicollisions in iterated hash functions. application to cascaded constructions. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *LNCS*, pages 306–316. Springer, 2004.
10. John Kelsey and Tadayoshi Kohno. Herding hash functions and the Nostradamus attack. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *LNCS*, pages 183–200. Springer, 2006.
11. John Kelsey and Bruce Schneier. Second preimages on n-bit hash functions for much less than 2^n work. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 474–490. Springer, 2005.
12. Lars R. Knudsen and John Erik Mathiassen. Preimage and collision attacks on MD2. In Henri Gilbert and Helena Handschuh, editors, *FSE*, volume 3557 of *LNCS*, pages 255–267. Springer, 2005.
13. Xuejia Lai and James Massey. Hash function based on block ciphers. In Rainer A. Rueppel, editor, *EUROCRYPT*, volume 658 of *LNCS*, pages 55–70. Springer, 1992.
14. Stefan Lucks. Design principles for iterated hash functions. Cryptology ePrint Archive, Report 2004/253, 2004.
15. Ueli M. Maurer and Stefano Tessaro. Domain extension of public random functions: Beyond the birthday barrier. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *LNCS*, pages 187–204. Springer, 2007.
16. Alfred Menezes, Paul van Oorschot, and Scott Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
17. Ralph Merkle. One way hash functions and DES. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *LNCS*, pages 428–446. Springer, 1989.
18. Mridul Nandi and Douglas Stinson. Multicollision attacks on generalized hash functions. Cryptology ePrint Archive, Report 2004/330, 2004. Later published in [19].

19. Mridul Nandi and Douglas Stinson. Multicollision attacks on a class of hash functions. *IEEE Transactions on Information Theory*, 53:759–767, 2007.
20. Bart Preneel. *Analysis and Design of Cryptographic Hash Functions.* PhD thesis, Katholieke Universiteit Leuven, 1993.
21. Bart Preneel, René Govaerts, and Joos Vandewalle. Hash functions based on block ciphers: A synthetic approach. In Douglas R. Stinson, editor, *CRYPTO*, volume 773 of *LNCS*, pages 368–378. Springer, 1993.
22. Jean-Jacques Quisquater and Jean-Paul Delescaille. How easy is collision search? Application to DES (extended summary). In Jean-Jacques Quisquater and Joos Vandewalle, editors, *EUROCRYPT*, volume 434 of *LNCS*, pages 429–434. Springer, 1989.
23. Michael Rabin. Digitalized signatures. In Richard Lipton and Richard DeMillo, editors, *Foundations of Secure Computation*, pages 155–166. Academic Press, 1978.
24. Michael Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT, 1979.
25. Ronald Rivest. Abelian square-free dithering for iterated hash functions. ECRYPT Conference on Hash Functions, 2005. Also presented in [26].
26. Ronald Rivest. Abelian square-free dithering for iterated hash functions. First NIST Cryptographic Hash Workshop, 2005.
27. Bruce Schneier. *Applied Cryptography.* John Wiley & Sons, second edition, 1996.
28. Robert Sedgewick, Thomas G. Szymanski, and Andrew Chi-Chih Yao. The complexity of finding cycles in periodic functions. *SIAM Journal of Computing*, 11(2):376–390, 1982.
29. Kazuhiro Suzuki, Dongvu Tonien, Kaoru Kurosawa, and Koji Toyota. Birthday paradox for multi-collisions. In Min Surp Rhee and Byoungcheon Lee, editors, *ICISC*, volume 4296 of *LNCS*, pages 29–40. Springer, 2006.
30. Hongbo Yu and Xiaoyun Wang. Multi-collision attack on the compression functions of MD4 and 3-pass HAVAL. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *ICISC*, volume 4817 of *LNCS*, pages 206–226. Springer, 2007.