Cryptanalysis of hash functions with structures

Dmitry Khovratovich

University of Luxembourg

Abstract. Hash function cryptanalysis has acquired many methods, tools and tricks from other areas, mostly block ciphers. In this paper another trick from block cipher cryptanalysis, the structures, is used for speeding up the search for collisions for hash functions. We investigate the memory and time complexity of this approach under different assumptions on round functions. The power of the new attack is illustrated with the cryptanalysis of the hash functions Grindahl (256 and 512 bit versions) and RadioGatun (all versions). The collision attack on Grindahl-512 is the first collision attack on this function.

1 Introduction

Since 1990 the MD family of hash functions and its successor SHA family have been most widely used data integrity primitives. In contrast to few cryptanalytic results in 90s last attacks on MD5 [28], SHA-0 [24], and SHA-1 [8] make us to look for more reliable components. A higher level standard, the Merkle-Damgard approach [13, 25] to build hash functions from compression functions, has lost a part of credit due to such generic attacks as multicollisions [17], second-preimage search with expandable messages [20], and the herding attack [19].

In order to counteract the latter attacks several new designs and frameworks were recently proposed by Lucks (wide-pipe hash [23]), Biham&Dunkelman (HAIFA [5]), Bertoni, Daemen et al. (sponge functions [3]), Coron et al. (the chop solution [10]).

In parallel to generic constructions several new hash functions were designed: based on some well-known hard problems (LASH [1], DaKoTa [14], VSH [9]) or the security of some block and stream cipher primitives (Grindahl [22], RadioGatun [2]). Some hash functions also fit one of designs described above: RadioGatun as a sponge function, LAKE [16] as a HAIFA hash function.

In contrast to generic attacks like multicollisions, which are applicable to hash functions with Merkle-Damgard strengthening, attacks on lower level *compression functions* are highly dependable on a particular proposal and can be rarely extended to other functions. Common ideas are mostly related to the notion of differentials since the fact that two different messages produce the same hash value (a *collision*) can be expressed as zero difference in the output.

The idea of differentials comes from block cipher cryptanalysis and pioneering papers by Biham and Shamir [6]. As high probability differential characteristics were exploited in attacks on block ciphers as high probability zero-ending differential paths, or *trails*, are used to find collisions for compression functions.

Since block cipher cryptanalysis is highly developed many cryptanalysts tried to use the most efficient methods and tools in attacks on hash functions. However, due to stronger requirements on the results of an attack only few of them were applied. The amplified boomerang attacks [18] and use of truncated differentials [26] are the examples.

In this paper we investigate another tool from block cipher cryptanalysis: *structures*. A structure is originally a group of the plaintexts (or that ciphertexts) that pairwise have some property (e.g. zero difference in particular bytes). Since the number of pairs with desired properties in a structure is much bigger than the structure size, such constructions are widely used in order to save memory and time in attacks on block ciphers [7, 11, 21].

Intuitively, structures might have been used in attacks on the hash functions built on block ciphers [27]. However, the authors are aware of only one such attack: a recent attack on Snefru [4], though Snefru does not directly fit the constructions from [27].

This paper is organized as follows. First we briefly explain how structures reduce the cost of collision search providing some differential trail. In Section 3 we make some observations on

how such trails can be found. Then we investigate how structures follow the differential trail and collapse to pairs in some step (Section 4) so that the standard differential approach can be applied afterwards. We also derive the memory complexity of the attack. In Section 5 the time complexity of the attack is estimated under different assumptions.

Then we do the cryptanalysis of Grindahl and RadioGatún with structures(Section 6). The cost of collision for Grindahl-256 is reduced compared to the recent attack by Peyrin [26]. We present the first collision attack on Grindahl-512.

RadioGatún being one of fastest hash functions so far [2] is a tasty object for a cryptanalyst. We give the best known attack on RadioGatún, though the claimed security level is not broken. Finally, we summarize our results and give directions for future development of our ideas.

2 Idea in brief

In many attacks on compression functions a cryptanalytic deals with a set of pairs that are to follow a particular differential trail. Here the *trail* is a sequence of differences in internal states, which are measured before and after the application of the round function (see [2] for a more formal approach). With respect to the message scheduling, an adversary has some freedom in first steps thus increasing the number of pairs that follow the trail (the attack by De Canniere and Rechberger on SHA-1 [8] is an example). After some step, when there is no more degree of freedom that can save a pair or this freedom is little, the number of pairs tends to decrease. We show how this effect can be postponed if a differential trail allows to incorporate pairs into structures as follows.

In order to distinguish between the approach when a cryptanalytic deals with pairs and our approach we call the former one the *standard differential attack*. It is also known as the *trail backtracking* [2]. Our attack is later called *the structural approach*, or the *structural attack*.

Now assume that the trail deals with truncated differentials, and the possible differences form a linear space R of differences. Then if a pair of states (S_1, S_2) fits the trail, and a pair (S_2, S_3) fits the trail, then the pair (S_1, S_3) fits the trail too. Such a group of states is called a *structure*.

Suppose at some step a structure $(S_1, S_2, \ldots, S_Q)$ of size Q enters the round, and for a randomly chosen pair of states (S_i, S_j) the probability to come out of the round with one of allowed differences is P. Then every state S_i will have a desired difference with $P(Q-1)$ states $S_{i_1}, S_{i_2}, \ldots, S_{i_{P(Q-1)}}$ thus composing a smaller structure. The other states are not left out but grouped into structures as well. Thus the initial structure divides into $\frac{Q}{P(Q-1)+1} \approx \frac{1}{P}$ structures of size $\sim PQ$. If $P(Q-1) = 1$ then $Q/2$ pairs come out, and the differential attack is launched.

Now consider a more complicated case. Suppose there is some additional freedom from message injection. More precisely, to a given pair of states a pair of messages (M, M') is injected such tthat there are V possibilities for M and D possibilities[1] for M' for each M (VD pairs at all). So if T pairs enter a round, which has probability P to be passed, then about $V \cdot D \cdot P \cdot T$ pairs survive. See Figure 1 for the outline of the situation.

When we deal with structures, the value of the injected message can be freely chosen only for the first state of a new structure, or the *leader state*; the messages injected to the other states should have a desired difference with a first one. However, if there is no message such that the resulting state fit the structure then the state becomes a leader state for a new structure and may be recalculated with another message value.

Consequently, the message freedom results in structures of size $\approx D \cdot P \cdot Q$. The number of states remains the same; however, it can be increased if we take other states as leading ones.

3 How to get a trail

The reader may ask the question: how to get a trail for a particular hash function. Unfortunately, the best attacks presented so far use the trails that are found ad-hoc and are specific for the hash

[1] We distinguish D from V because the differential trail may impose restrictions on injected differences

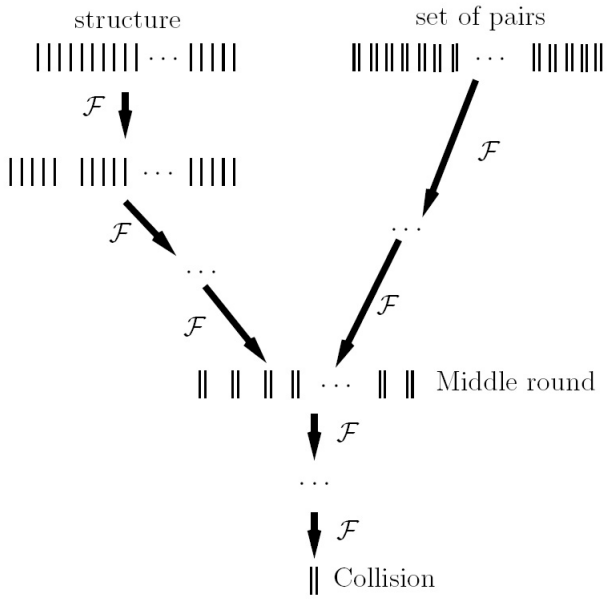

Fig. 1. Comparative view on the structural and the differential approach. The number of states in the structural approach does not change till structures collapse to pairs (*middle round*).

function. For SHA family this process was partly automatized by De Canniere and Rechberger in [8]. Peyrin in the attack on Grindahl sought the trail that starts with the full-difference state and has the best conditions/(control bytes) ratio.

The trails used in our attacks on Grindahl and RadioGatún (Section 6) have been obtained by a simple backtracking process. The idea is to start with zero-difference state and step back with introducing differences by all message injections. These difference spread to the internal state till every byte (or another building block) contains the difference. The number of steps is subject to the diffusion properties of the internal transformations.

One may argue that some round functions provide diffusion close to ideal in means of the fact that each bit of an internal state is affected by the message injection after one round. However, the size of a message block to be injected is usually much smaller than the internal state, which might help to distinguish some specific classes of differences.

4 Analysis of structure fission

In order to benefit from the number of pairs in a structure an adversary should keep the size of the structure as big as possible. Let us estimate the size as a function of the round probability and the freedom given by message injections. Here and later the attack follow a chosen differential trail as described in Section 2. We assume that at each step due to the trail either the values of injected messages are defined in early steps, or the values can be varied but the difference is fixed, or the difference can be varied as well.

Denote by $\sim$ the desired binary relation between two states, which can be also interpreted as the fact that the difference in the stated fits the trail requirements. Here and later we use this notion for all rounds covered by the differential trail, and the particular relation is usually clear from the context. A formal definition of a structure with respect to the binary relation follows.

Definition 1. *A* structure *is a set of internal states such that any two states of the set satisfy the binary relation* $\sim$.

In our attacks on GRINDAHL and RADIOGATÚN(Section 6) the relation is of form "bytes (respectively, words) $i_1, i_2, \ldots, i_k$ are equal".

No freedom in message injection. Suppose a structure of size $Q = 2^q + 1$ states enters a round with the message injection in the beginning. First consider the case when there is no freedom in message injection due to the differential trail or message scheduling. States $S_1, \ldots, S_Q$ are transformed to states $S'_1, \ldots, S'_Q$. The probability that arbitrary chosen states S'_i and S'_j have the desired difference is the conditional probability $\mathbb{P}(S'_i \sim S'_j \mid S_i \sim S_j) = \frac{\mathbb{P}(S'_i \sim S'_j, S_i \sim S_j)}{\mathbb{P}(S_i \sim S_j)}$. If we additionally assume the independence of the relations in adjacent rounds or the positive correlation between the corresponding events[2] then $\mathbb{P}(S'_i \sim S'_j)$ is the lower bound for the desired probability.

Let us introduce $p = -\log_2 \mathbb{P}(S'_i \sim S'_j \mid S_i \sim S_j)$. A randomly chosen state S' has the desired difference with about $2^q \cdot 2^{-p}$ states. Thus the original structure divides into about 2^{-p} smaller structures of average size 2^{q-p}.

The reader may argue that the concrete partition may give us so different values of structure size that it may badly affect the number of pairs to be composed. However, providing the constant number of structures in the partition it is easy to prove that the structures of the same size give the lowest number of pairs (see Appendix B).

On the other hand, this might be not the case when p is so high that a number of states does not have the desired difference with any other one. In that case we assume that these states are just removed from the consideration while the others are composed into pairs. One may compose 2^{2q-1} pairs from a structure of size $2^q + 1$. Thus under some reasonable assumptions on independency of the events about 2^{2q-1-p} pairs comes out of one round. Then the pairs are processed by the standard differential attack.

Value freedom in message injection. Suppose now there is some freedom in message injections but the differential trail does not allow to introduce a difference, or the value of the difference is fixed. Then every state can be transformed to at max one element of a new structure, so the structures do not grow in this case. However, one may increase the number of structures and thus the number of considered states.

The latter approach increase the memory complexity so we do not use it except for the round when all structures collapse to pairs. Given $V = 2^v$ possibilities for an injected message one obtains $2^{2q-1+v-p}$ pairs after the round.

Difference freedom in message injection. Here we assume that the trail allows to inject 2^d possible differences[3] by the message block. This fact provides structures of bigger size because we have more freedom in steering a state into the structure.

Suppose that state S_i has already transformed to state S'_i by message v_i: $S_i[v_i] = S'_i$. Let us compute the probability that a randomly chosen state S_j can be transformed to some state with the desired difference with S'_i by some message m', which follows the trail too. The probability can be expressed as $\mathbb{P}(\exists m' : S_j[m'] \sim S'_i, m' \sim m \mid S_i \sim S_j)$.

Assuming that the events for particular messages are independent we obtain the following expression:

$$\mathbb{P}(\exists m' : S_j[m'] \sim S'_i, m' \sim m \mid S_i \sim S_j) = \\ = 1 - \prod_{m' \sim m} \mathbb{P}(S_j[m'] \nsim S'_i \mid S_i \sim S_j) = 1 - (1 - 2^{-p})^{2^d} \approx 2^{d-p}. \quad (1)$$

Remark 1. We assume that the probability p does not depend on a particular pair. A sufficient condition for this is the uniform distribution in every word (byte) among the states in the structure. This is usually the case if the compression function is first iterated with random messages for several rounds.

[2] It can be provided by the particular trail.

[3] Recall that they form a linear space.

Consequently, one structure divides into structures of average size $Q' = 2^{q+d-p}$. Analogously, if $q + d - p < 0$ the structure collapse to pairs. Since $2^{2q-1+v+d}$ pairs can be composed about $2^{2q-1+v+d-p}$ come out of the round.

Remark 2. If the parameter d is close to p the approximation of (1) does not hold. The exact value of $\mathbb{P}$ can be estimated as $2^{d-p} + O(2^{2d-2p})$. As a result, the probability is actually equal to $1 - 1/e$ when $d = p$. Thus the size of structure is roughly halved. We take this fact into account in the attack on RadioGatún (Section 6.3). However, this effect can be compensated if we allow more than one state to be derived from an initial one (which by default causes non-uniformity).

Remark 3. If a state is steered to only one structure (due to restrictions on the complexity of the attack) then the resulting structures differ in size. The reason is that there are fewer states to form the structure. If the probability to be composed into the structure is α then the first structure consists of $1/\alpha$ of all states, the second one consists of $(1-\alpha)\alpha$ and so on. The resulting number of pairs is about $2^{2q+2d-2p-2}$ which is equivalent to 2^{d-p} structures of average size $2^{q-1/2}$. Thus in the attacks the structure is roughly halved each two rounds thus slightly increasing the requirements for the size of the initial structure. The attacks presented in this paper use structures for at max three rounds so we do not take this effect into account.

Size of the initial structure. Now we can answer the question, what the size of the initial structure should be to maintain the collision search. As we pointed out, at some step the structures collapse to pairs and afterwards we use a standard differential attack.

Consider the differential trail that we follow. At round i there are v_i degrees of freedom[4] in the values of the injected message, d_i degrees of freedom in the differences in injected messages, and p_i (bit) conditions to be satisfied. In the standard differential attack we start with 2^c pairs and leave with one pair in the end. Thus we obtain the following equation:

$$c + \sum_{i=1}^{T} (v_i + d_i - p_i) = 0.$$

Here T stands for the number of rounds covered by the trail. We also denote by $c(t)$ the logarithm of the number of pairs after t-th round:

$$c(t) = c + \sum_{i=1}^{t} (v_i + d_i - p_i) = c + \underbrace{\sum_{i=1}^{t} v_i}_{v(t)} + \underbrace{\sum_{i=1}^{t} d_i}_{d(t)} - \underbrace{\sum_{i=1}^{t} p_i}_{p(t)}.$$

Suppose we start with a structure of size 2^q, which collapse to pairs after $l+1$ rounds, $l < T$. The structure divides into $2^{p(l)-d(l)}$ smaller structures after l rounds. Each structure is of size about $2^{q+d(l)-p(l)}$. Thus about $2^{2q+d(l)-p(l)-1}$ pairs come out of round l.

In order to continue the collision search and obtain one pair in the end the following equation should hold:

$$2q + d(l) - p(l) - 1 = c(l) \;\Leftrightarrow\; 2q = c + v(l) + 1 \;\Leftrightarrow\; q = \frac{c + v(l) + 1}{2}.$$

The memory complexity is thus determined by the maximum of 2^q and $2^{c(l+1)+1}$. It can be finally expressed as

$$\min_{0 \le l < T} \max(2^{\frac{c+v(l)+1}{2}}, 2^{c(l+1)+1}). \tag{2}$$

The plot of the memory complexity of the attack with structures compared to a standard differential attack is drawn in Figure 2. There c stands for the logarithm of the number of pairs required by the differential attack, q stands for the logarithm of the size of the structure that is used for the structural attack.

[4] The notion of *degrees of freedom* stands for base 2 logarithms of the number of admissible values.

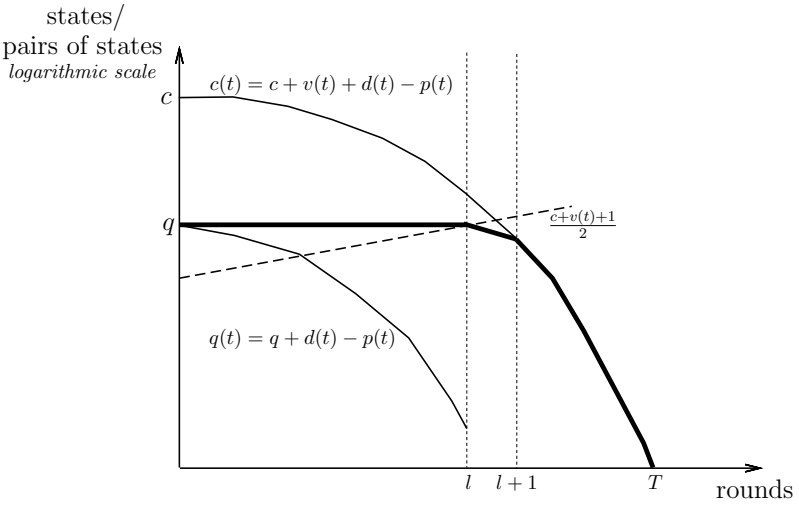

Fig. 2. Memory complexity of the collision search with structures.

5 Analysis of complexity

The amount of processed states determines the memory complexity of the collision search. The time complexity analysis is much harder due to the fact that we have to arrange states into structures as fast as possible.

We consider the complexity of the fission of only one structure, because this process is independently applied for all structures. To obtain the whole complexity one should multiply the derived values by the current number of structures.

No freedom in message injection. States S'_i and S'_j belong to the same structure[5] if $S'_i \sim S'_j$. On the other hand, if $\sim$ defines set R of linear differences then the condition can be expressed in terms of projections to space $R^\perp$ that is orthogonal to R:

$$S'_i \sim S'_j \;\Leftrightarrow\; \mathrm{pr}_{R^\perp} S'_i = \mathrm{pr}_{R^\perp} S'_j.$$

As a result, we compute the ordered set of projections of structures and use binary search to derive the structure a state belongs to. Assuming the sorting and search costs are negligible comparing to the round iteration we derive the complexity roughly equal to the number of states.

Value freedom in message injection. The fact that there is the value freedom in message injection does not affect the complexity of the attack if structures do not collapse into pairs yet. The exact value of the complexity in this case is just the number of states after the round iteration.

Consider the case when structures are collapsed to pairs and single states. As mentioned in Section 4, the structure contains 2^{2q-1+v} pairs. The further steps depend on whether we can exploit properties of the round function.

- If we consider the round function as a black box, we just derive pairs for each possible message m. The complexity is about 2^{q+v}.
- If we can fast find solutions m for the equation

$$\mathrm{pr}_{R^\perp} S_i[m] = \mathrm{pr}_{R^\perp} S_j[m] \tag{3}$$

  then it is solved for all possible pairs about 2^{2q-1} times.

[5] Whether structures actually collapse into pairs does not matter.

– If there exist not only a fast algorithm for solution (3) but also function f such that (3) has a solution iff $f(S_i) = f(S_j)$. Then we compose the ordered set of $f(S)$ and for each new state look for a pair with negligible cost. The complexity would be equal to the maximum of the size of the initial structure (2^q) and the number of resulting pairs ($2^{2q-1+v-p}$).

Difference freedom in message injection. Again, first, we investigate the case when structures do not collapse to pairs. Suppose states $S'_1, \ldots, S'_i$ have been already distributed into just created structures. We also require that every leader state is obtained by the same injected message m_0. A state S_{i+1} can be distributed to the structure with the leader state S' if there exist a message m_{i+1} such that $m_0 \sim m_{i+1}$ and $\mathrm{pr}_{R^\perp} S' = \mathrm{pr}_{R^\perp} S_{i+1}[m_{i+1}]$. Denote by S the set of all such states $S_{i+1}[m_{i+1}]$. Then the question is whether $\mathrm{pr}_{R^\perp} S'$ belongs to $\mathrm{pr}_{R^\perp} \mathsf{S}$.

If S is an affine space, and the linear space does not depend on S_{i+1} then we can easily compute the projection and find the corresponding structure using the ordered set approach. The complexity would be equal to the number of states. If S is not an affine space but can be represented as a union of affine spaces then we compute the projection for each space. In the worst case the complexity is equal to 2^d multiplied by the number of states.

Now consider the case when a structure collapse to pairs. This is actually the most complicated case and can be considered as a bottleneck. Indeed, about $2^{2q-1+v+d}$ pairs are composed from a structure with 2^q states. About $2^{2q-1+v+d-p}$ pairs come out of the round iteration. The possible approaches are similar to the case when there is no freedom in difference:

– If the round function is a black box we first compute the states that are to be iterated with the default message m_0. Then we compute for each initial state the set S as discussed before. According to division of S to affine spaces we obtain the complexity from 2^{q+v} to 2^{q+v+d}.
– If we can fast solve the equation

$$\mathrm{pr}_{R^\perp} S_i[m] = \mathrm{pr}_{R^\perp} S_j[m + \Delta m]. \tag{4}$$

Here Δm stands for a admissible difference, then it is solved for all possible pairs (about 2^{2q-1} times).
– If there exist not only a fast algorithm for solution (4) but also function f such that (4) has a solution iff $f(S_i) = f(S_j)$. Similarly, we obtain the complexity about the maximum of the size of the initial structure (2^q) and the number of resulting pairs ($2^{2q-1+v+d-p}$).

In particular attacks some ad-hoc optimizations can be maintained (Section 6).

6 Practical attacks

6.1 Grindahl-256

Description. Grindahl is a family of hash functions proposed by Knudsen, Rechberger and Thomsen at FSE 2007 [22]. It can be also considered as a sponge-like function, though message injection is a bit different. The round function of Grindahl uses the design components of AES [12]: SubBytes and the MixColumns operation. Since the internal state of Grindahl is wider than that of AES (Grindahl-256 can be viewed as a byte matrix of 4 rows and 13 columns) it uses a modified ShiftRows transformation in order to obtain better diffusion. The other message-independent transformation is AddConstant, which adds a constant to a particular byte.

In Grindahl-256 the message injection is just the substitution of the first column with 4-byte message block. The round function is defined as the following composition of transformations:

$$P(\alpha, M) = \textsc{MixColumns} \circ \textsc{ShiftRows} \circ \textsc{SubBytes} \circ \\ \circ \textsc{AddConstant} \circ \textsc{InjectMessage}(\alpha, M).$$

Here α denotes the state to be iterated, and M the message block to be injected. Every message block is used only once.

In order to obtain a hash value the state filled with zeros is iterated till the message is ended. Then 8 blank rounds are applied and the resulting state is truncated to 256 bits, which is the hash value.

Security. The designers of Grindahl-256 claimed the security level of 2^{128} operations against both collision and second-preimage attack. Peyrin in [26] found a differential trail, which leads to a full collision in an internal state before the blank rounds are applied. The trail deals with two values of byte differences: non-zero and zero. It starts with a pair of states that differ in all bytes and after 9 message injections leads to a collision. Following our notation, he had 55 byte conditions, 21 byte degrees of value freedom[6], and 20 byte degrees of difference freedom thus obtaining complexity $2^{(55-21-20)*8} = 2^{112}$ message pairs. There is also an attack on the prefix-MAC built on Grindahl [15].

Although Grindahl-256 is already broken, the goal of our attack is not only the illustration of structural technique. Peyrin provided some ad-hoc observations on the fact that his attack is one of the best dealing with truncated-differential approach, and 2^{104} is the lower bound on the complexity of such attack. Our attack breaks this bound.

Attack. In order to apply the structural approach we first have to a bit modify the class of truncated differentials. Here and later we consider two-valued byte-difference: $*$ (random difference, including 0) and 0 (bytes coincide). They are marked as grey and white cells in Figure 3, respectively. One can easily check that this not only barely affect the probability of the trail and the complexity of the collision search but also simplify computations.

The second barrier is that the trail used by Peyrin for collision search is badly suited for the structural approach due to the distribution of probabilities among the iterations, which helps the standard differential attack but does not provide the best results for the structural attack. Table 1 (a) shows that we would have to start with a structure if $2^{12.5*8} = 2^{100}$ states, which is too small advantage against Peyrin's attack.

The better complexity is provided by the second trail from [26], which was proposed for the second-preimage search. However, there is a mistake in Peyrin's paper: the byte C inserted before the k-th iteration does not affect column 11 in the $k+1$-th iteration. As a result, the complexity of a simple truncated differential attack is $2^{21*8} = 2^{168}$ pairs. However, the structural attack needs only a set of $2^{10.5*8+0.5} = 2^{84.5}$ states (Table 1 (b)).

The attack works as follows. Iterate Grindahl-256 for 10 rounds with randomly chosen messages and obtain a structure with $2^{84.5}$ states. Then the first iteration is passed without a fission due to 4-byte difference freedom. After the second round the structure collapses to 2^{72} pairs, and only one pair comes out of the next iteration.

i	v_i	d_i	p_i	$c(i)$	$\frac{c+v(i)}{2}$	$q(i)$
Start	–	–	–	14	7	12.5
1	0	2	2	14	7	12.5
2	3	4	7	14	8.5	9.5
3	4	3	7	14	10.5	5.5
4	4	2	7	13	**12.5**	1.5
5	4	3	9	11	14.5	–
6	4	4	14	5	16.5	–
7	2	2	9	0	17.5	–
$8-9$	0	0	0	0	17.5	–

(a)

i	v_i	d_i	p_i	$c(i)$	$\frac{c+v(i)}{2}$	$q(i)$
Start	–	–	–	21	10.5	10.5
1	0	4	4	21	**10.5**	10.5
2	4	4	20	9	12	–
3	4	3	15	0	14	–
$4-5$	0	0	0	0	14	–

(b)

Table 1. Parameters of differential trails for Grindahl-256. Measurement in bytes.

[6] In early steps there was more freedom that is required by the trail so there was no clear difference between the value freedom and the difference freedom. However, the structural approach benefits from the difference freedom so we first exploit the latter one.

Time complexity of the attack. Since some message bytes pass several SubBytes transformations it is not clear how costly the steps when we deal with structures are. We propose some optimizations, which lead to the complexity about 2^{100} operations though the technique can probably be improved. The reader may refer to Table 5 for better understanding.

In the first step there are 4 bytes of difference freedom and 4 bytes where the difference should be deleted. The leader state of a new structure is defined by iterating the round function with a random message block. For each next state S in the structure we must find the message bytes (A, B, C, D) to be injected[7], which lead to a state colliding in particular 4 bytes with the leader state. First consider column 7 before the MixColumns transformation in the second iteration[8] Three bytes of column 7 are not affected by the message injection and can be derived explicitly. On the other hand, one byte after the MixColumns transformation is known because a collision there is needed. Thus compute both the input and the output of the MixColumns transformation of column 7 and thus derive the value of D and the value of second byte in column 9 in the next iteration.

Then try all the values of C. For each value derive one more byte in column 9 in the third iteration. Thus two bytes in column 9 are known before the MixColumns transformation and two bytes are known due to the fact of collision. As a result, derive the values of A and B and check the MixColumns transformation in column 3 of the second iteration with the latter two values. On average, 2^7 trials of C are required.

The second step is actually the bottleneck of our attack, though we believe that the complexity may be reduced. First vary B and C for each state thus obtaining $2^{100.5}$ states. Then the 16 bytes in the third iteration where zero difference is desired are fully determined by 6 bytes that are affected by A and D. This gives us $16 - 6 = 10$ byte conditions, which can be used to divide the set of states into structures. One more condition we get from the second iteration, where the byte was affected by just fixed B. Thus we obtain $2^{(10+1)*8} = 2^{88}$ blocks each of size $2^{12.5}$. In every block we have 6 variables and 6 conditions; the other conditions are provided by constants. Since we process the blocks independently, the memory complexity is not increased.

Then consider the unknown bytes in columns 3, 5 and 11 that are affected only by A. Consider two random states in a block and denote by x_A and x'_A the message byte A after the SubBytes transformation. Then the fact of zero difference in those columns can be expressed as the following system of equations:

$$\begin{cases} a_{12}S(a_{21}x_A + c_1) + c_2 = a_{12}S(a_{21}x'_A + c'_1) + c'_2; \\ a_{11}S(a_{31}x_A + c_3) + c_4 = a_{11}S(a_{31}x'_A + c'_3) + c'_4; \\ a_{14}S(a_{41}x_A + c_5) + c_6 = a_{14}S(a_{41}x'_A + c'_5) + c'_6. \end{cases}$$

Here a_{ij} are coefficients of the MixColumns matrix and c_i are state-dependent constants. Due to properties of the AES S-box x_A and x'_A are uniquely determined (if there is a solution) by constants $\overline{c} = (c_1, c_2, c_3, c_4, c_5, c_6)$ and $\overline{c}' = (c'_1, c'_2, c'_3, c'_4, c_5, c_6)$. Furthermore, this property is transitive, so that we precompute the function $f : \overline{c} \to x_A$.

As a result, a block of 2^{12} states divides into 2^8 blocks with 2^4 states each where A, B and C are fixed. In order to obtain the value of D repeat the same trick in columns 7, 11, and 12 thus getting one pair per 2^4 states, or 2^{96} pairs at all. Only 2^{72} pairs of them pass through 3 conditions in column 9 in the fourth iteration.

In the last third step we have to pass 15 byte conditions given 6 byte degrees of freedom. Since we deal with separate pairs, the filtering process be maintained with precomputations (see [26]).

6.2 Grindahl-512

The hash function Grindahl-512 is defined similarly to Grindahl-256, but the internal state is twice as big as that of Grindahl-512: it has 8 rows and 13 columns. Each injection of a message

[7] We keep this notation for future steps as well.

[8] In each step we deal with several consecutive iterations because the message injection does not affect all the columns during one round.

block substitutes the first column with 8 bytes of a message. The row offset values are defined by the following expression:

$$c_i = i + 1;\ 0 \leq i \leq 7.$$

The MixColumns matrix is also redefined but the exact coefficients are irrelevant to our attack. The only property we use is that this matrix is MDS with branch number 9.

Security So far there is no collision attack on GRINDAHL-512 though a weakness of using GRINDAHL-512 as the base of prefix-MAC was shown [15].

Attack

Differential trail. We use a 3-round differential trail, which is shown at Figure 4. The trail is obtained by iterating GRINDAHL-512 backwards from the zero-difference state. It is assumed that the last truncation (before the injection) deletes a column with 6 byte differences, while the first two truncations delete the full-difference column.

A direct application of the complexity measurement approach described in Section 5 gives the following memory complexity table (values in bytes).

i	v_i	d_i	p_i	$c(i)$	$q(i)$
	–	–	–	48	**28**
1	8	8	21	43	15
2	8	8	49	**10**	–
3	2	2	14	**0**	–

However, the second step becomes so time-consumable that the resulting complexity overcomes the brute-force one. The reason is that structures are too large to be quickly recomposed into pairs. Testing all possible injections increase time complexity as well.

We managed to decompose the second round into two sub-rounds with only slight increase of the complexity. The idea is as follows. We first process the zeros that are the result of the second MixColumns transformation and that are affected by the second message injection. These are 21 zeros in columns 1–8. For any two states that follow the trail before the second injection the condition of having zero difference in these positions is equivalent to 21 linear equations with the differences in the internal state after the S-box application as variables. Since the message injection can be equivalently swapped with the S-box transformation we obtain that the 21 equations are 21 linear conditions on 8 differences in the message block.

Thus 2^{13*8} structures of size 2^{15*8} split into $2^{(13+21-8)*8} = 2^{26*8}$ structures of size 2^{2*8}. These structures collapse to pairs and are partly filtered out due to the remaining 28 byte conditions though 8 byte degrees of freedom are still available. Then we compose all possible 2^{30*8} pairs and filter them out. The desired values to be injected can be derived from pre-computed tables, which are applicable since we already deal with pairs. The resulting complexity is 2^{240} computations and still 2^{224} memory. The complexity of the last step is negligible. We also modify the memory complexity table taking into account the considerations discussed above (Table 2).

6.3 RadioGatún

Description. RADIOGATÚN [2] is a family of hash functions proposed by Bertoni, Daemen, van Assche, and Peeters at the Second Cryptographic Hash Workshop in 2006. It can be considered as an example of the sponge construction [3].

RADIOGATÚN operates on small message blocks, which are used only once, and a round function. After a message is fully processed RADIOGATÚN generates output of infinite length by just consecutively applying the round function and taking a block in an internal state as a new output block. There is no message scheduling: each message word is used only in one round.

i	v_i	d_i	p_i	$c(i)$	$q(i)$
	–	–	–	48	**28**
1	8	8	21	43	15
2 - I	0	8	21	36	2
2 - II	8	0	28	**10**	–
3	2	2	14	**0**	–

Table 2. Differential trail for GRINDAHL-512.

RADIOGATÚN operates on words of some integer length l_w (the parameter is fixed for a concrete hash function). An internal state of RADIOGATÚN consists of two substates also called *belt* (B) and *mill* (A) of size $39l_w$ and $19l_w$ respectively. The round function treats them differently. The belt is updated by a simple linear transformation and fed with 12 words of the mill and with 3 words of the message block in a linear way. The mill is fed with 3 words of the belt and 3 words of the message block in a linear way and afterwards is updated by a nonlinear function. The resulting round function is invertible.

Denote the injected block by $M = (M_1, M_2, M_3)$. Following this notation, the round function of RADIOGATÚN transforms a state $S = (A, B)$ to a new state $S' = (A', B')$ as follows:

- $B \xleftarrow{\text{Message injection } (M) \text{ and shift}} B$;
- $A \xleftarrow{\text{Message injection } (M)} A$;
- $B' \xleftarrow{\text{Mill2Belt feedforward } (A)} B$;
- $A \xleftarrow{\text{Mill function}} A$;
- $A' \xleftarrow{\text{Belt2Mill feedforward } (B')} A$.

The belt is represented as a rectangle of height 3 and width 13. A message block is added modulo 2 to the column 0: $B[i, 0] \leftarrow B[i, 0] \oplus M[i]$, then the belt is rotated one column to the right: $A_{new}[i, j] = A[i, j + 1]$ (here and later all indices are taken modulo 19). The same message words are added to three words of the mill: $A[i + 16] = A[i + 16] \oplus M[i]$. Then 12 words of the mill are added to the belt:

$$B[i + 1, i \bmod 3] = B[i + 1, i \bmod 3] \oplus A[i],\ 0 \le i \le 11; \quad B' = B;$$

The Mill function is a composition of transformations: $\iota \circ \theta \circ \pi \circ \gamma$, and the only non-linear transformation is γ:

$$\gamma: \qquad a[i] = A[i] \oplus \overline{A[i + 1]}\,\overline{A[i + 2]}.$$

Here $A[i]$ denotes the i-th word of A. Let us note that γ is a bitwise transformation.

The π, θ, and ι transformations are defined by the following expressions:

$$\begin{aligned} \pi: &\qquad A[i] = a[7i] \ggg i(i + 1)/2; \\ \theta: &\qquad a[i] = A[i] \oplus A[i + 1] \oplus A[i + 4]; \\ \iota: &\qquad A[0] = a[0] \oplus 1; \quad A[i] = a[i],\ 1 \le i \le 18. \end{aligned}$$

Here $\ggg$ stands for rotation.

Security. The designers of RADIOGATÚN claim the security level of $2^{9.5l_w}$ operations for both the collision and the second-preimage attack. On the other hand, the collision in an internal state can be found with $2^{27.5l_w}$ hash queries using the birthday paradox. Although our attack does not break the security level, it is significantly faster than the birthday attack.

Attack. Consider a differential trail with truncated differentials, which is exploited similarly to our (Section 6.1) and Peyrin's attacks on GRINDAHL. The word difference is two-valued: $*$ (random difference, including 0) and 0 (words coincide). These values are marked as grey and white cells in Figure 5, respectively.

The trail starts with two states with no restrictions on difference in the state (grey cells in all positions). After six message injections the states has zero difference. At each step convert to zero some differences in the belt, and each word is affected only once. We also require zero difference in all the mill words after the last message injection. Overall, there are 58 word (or $58l_w$ bit) conditions in the trail.

Careful analysis shows that the Mill function does not provide ideal diffusion. More precisely, words 0, 3, 6, 10, 11, 14 and 18 of the mill are not affected by the message injection. The distribution of differences in the other words is not completely random either, however we assume that the contents of the belt is distributed randomly, which allows us to maintain the attack.

Then vary the message injection words in order to obtain particular differences in words 1, 2, 4, 5, 7, 8, 9, 12 in the mill in the current iteration and in words 3, 6, 10, 11 in the next iteration. Under our assumption the conditional probability at each step (Section 4) is equal to $2^{-(\text{number of bit conditions at the step})}$. A condition corresponds to the message injection that affects it last. The indices of injections are inserted to the cell, where zero difference has been just obtained (Figure 5). The distribution of the probabilities and degrees of freedom are provided in Table 3.

i	v_i	d_i	p_i	$c(i)$	$\frac{c+v(i)}{2}$	$q(i)$
Start	–	–	–	28	14	17
1	0	3	3	28	14	17
2	3	3	10	24	15.5	10
3	3	3	11	19	**17**	2
4	3	3	15	10	18.5	–
5	3	3	13	3	20	–
6	0	3	6	0	21.5	–

Table 3. Parameters of the differential trail for RADIOGATÚN (l_w is omitted).

Thus a standard differential attack would require 2^{28l_w} pairs[9] while we start only with a structure of size 2^{17l_w+1} (we later omit this 1 for simplicity). These states can be taken randomly because no restrictions on difference are imposed in the beginning.

Time complexity. The main question is the time complexity of the attack. The analysis depends on either we divides a structure into structures, or a structure collapse to a pair, or pairs are filtered. We ask the reader to refer to Figure 5 for better understanding.

We begin with a structure of 2^{17l_w+2} states. In the first step we have 3 word degrees of difference freedom and 3 word differences that should be erased in the third iteration in the belt. However, the resulting differences are non-linear functions of the injected differences so that some states can not be steered into the new structure no matter what difference is injected. As a result, only $1/e$ of all states survive. The filtering is maintained as follows. We fix the value of the second injected word and mark the other injected values as $2l_w$ variables. The output of the γ transformation is thus linear with respect to these variables. As a result, the mill words in the output of the round function form an affine space $\overline{c} + L$ of dimension $2l_w$. We also remark that the coefficients of the variables are fully determined by only 3 mill words (with indices 15, 17, and 0). This implies that there are only 2^{3l_w} possible subspaces L.

On the other hand, the desired values of three mill words in the third iteration are fully determined by the 13 mill words entering the γ transformation in the second round. Thus about

[9] Bertoni et al. considered trails with fixed differences and found a trail with $c = 46l_w$, but they did not take into account difference freedom, which is $18l_w$ in our trail.

2^{10l_w} 13-word vectors produced the desired values. If the projection of a vector to $L^\perp$ is equal to $\overline{c}^{L^\perp}$ then there exists the injection that satisfies the trail condition. Since there are only 2^{3l_w} possible projections they can be precomputed, sorted and stored in the table of size $2^{13l_w+3l_w} = 2^{16l_w}$. The search in this table is of negligible cost. Thus the complexity of the first step is 2^{18l_w+2}. The resulting structure is of size 2^{17l_w+1}.

In the second step we do not have value freedom either. We first take only 3 conditions in the third iteration and find the difference in the message injection that leads to collision in these three words. We try all the possible values for the second word of the message injection so that the collision conditions are provided by the linear equations. As a result, we have the structure of size 2^{17l_w+1}. The other 7 conditions of the second step divide the structure into 2^{7l_w} smaller structures. The same trick holds for the third step.

In the fourth step 2^{15l_w} structures with 2^{2l_w+1} states each collapse to 2^{10l_w} pairs. Here there is the value freedom, which should be exploited, so the previous approach does not work. For each structure we try all possible C for all states in the structure and compute the part of the internal state that is not affected by M_1 and M_1. This area contains the three message words, where zero difference is desired. These conditions divide the set of 2^{3l_w+1} states to average 2^{3l_w} pairs. In order to define M_1 and M_2 for each pair we note that belt words $(1,2)$, $(2,4)$, and $(3,5)$ in the fifth iteration are linear functions of M_1 and M_2 (with fixed M_3). Then the requirement on zero difference in these words can be expressed as a system of $3l_w$ linear equations, and l_w more equations are provided by the simple condition in word $(0,1)$ in the belt in the fourth iteration. These $4l_w$ equations fully determine[10] M_1 and M_2. The other 8 conditions reduce the number of pairs to 2^{10l_w}. This step is actually the bottleneck and, the complexity is about 2^{18l_w} operations.

In the last two steps we deal with significantly smaller number of pairs, which allows to perform exhaustive search on message words to be injected.

The overall time complexity is 2^{18l_w} operations.

Implementation aspects

Due to high complexity of both attacks we did not fully launch them. The reduced version of, say, RadioGatún (with the smaller belt and mill) does not directly fit the framework of the attack because the output of the resulting round function are highly non-uniform and requires much more careful and complicated attack in spite of smaller prospective complexity.

However, the basic assumptions on the behavior of a structure and the possibility that an appropriate difference can be chosen were checked on RadioGatún with small l_w. For example, the theoretical implication that the size of the structure after the first round of the attack on RadioGatún decreased $(e/(e-1))$-fold was precisely confirmed by the calculations.

7 Conclusions and future work

We showed how the organization of internal states into structures can drastically reduce the complexity of collision search providing an appropriate differential trail. The exact formulas for memory complexity and estimates on time complexity of the attack with structures have been provided. We successfully combined our approach with simply obtained differential trails and presented the best known attacks on Grindahl and RadioGatún. The results are summarized in Table 4.

We believe that our attacks can be further improved with other differential trails or better optimization of the maintenance of structures. The complexity of the attack is now determined by the bottleneck step when structures collapse to pairs. It is likely that the plot of the complexity function can be significantly smoothed for some hash functions.

Another application of this technique may be bit-oriented hash functions with SHA-2 being the hardest and most famous among not broken ones. Although current differential trails for SHA

[10] Some matrices may be singular, which leads to either zero solutions or many solutions. We believe that the average number of solutions is 1.

Hash function	Attack	Memory complexity	Time complexity
Grindahl-256	Truncated differential [26]	2^{32}*	2^{112}
	Structural	2^{84}	2^{100}
Grindahl-512	**Structural**	2^{224}	2^{240}
RadioGatún	Differential [2]	2^{19l_w}*	2^{46l_w}
	Truncated differential	2^{10l_w}	2^{28l_w}
	Structural	2^{17l_w}	2^{18l_w}

* — approximate value, was not given in the paper.

Table 4. Summary of our attacks on concrete hash functions.

hash functions deal with fixed values of differences, some new results [29] give us the hope that this approach can be generalized.

References

1. Kamel Bentahar, Dan Page, Markku-Juhani O. Saarinen, Joseph H. Silverman, and Nigel Smart. Lash. Technical report, NIST Cryptographic Hash Workshop, 2006.
2. Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. Radiogatun, a belt-and-mill hash function, 2006, available at http://radiogatun.noekeon.org/.
3. Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. Sponge functions, 2007, available at http://sponge.noekeon.org/.
4. Eli Biham. New techniques for cryptanalysis of hash functions and improved attacks on Snefru. In *FSE'08*. Springer, to appear, 2008.
5. Eli Biham and Orr Dunkelman. A framework for iterative hash functions — HAIFA. *Cryptology ePrint Archive: Report 2007/278*, 2007.
6. Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems. In *CRYPTO'90*, volume 537 of *LNCS*, pages 2–21. Springer, 1991.
7. Eli Biham and Adi Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer, 1993.
8. Christophe De Cannière and Christian Rechberger. Finding SHA-1 characteristics: General results and applications. In *ASIACRYPT'06*, volume 4284 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2006.
9. Scott Contini, Arjen K. Lenstra, and Ron Steinfeld. Vsh, an efficient and provable collision-resistant hash function. In *EUROCRYPT'06*, volume 4004 of *LNCS*, pages 165–182. Springer, 2006.
10. Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-damgård revisited: How to construct a hash function. In *CRYPTO'05*, volume 3621 of *LNCS*, pages 430–448. Springer, 2005.
11. Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The block cipher square. In *FSE'97*, volume 1267 of *LNCS*, pages 149–165. Springer, 1997.
12. Joan Daemen and Vincent Rijmen. *The Design of Rijndael. AES — the Advanced Encryption Standard*. Springer, 2002.
13. Ivan Damgård. A design principle for hash functions. In *CRYPTO'89*, volume 435 of *LNCS*, pages 416–427. Springer, 1989.
14. Ivan Damgard, Lars Knudsen, and Soren S. Thomsen. Dakota — hashing from a combination of modular arithmetic and symmetric cryptography. Technical report, Echternach Symmetric Cryptography Seminar, available at http://wiki.uni.lu/esc/, 2008.
15. Michael Gorski, Stefan Lucks, and Thomas Peyrin. Slide attacks on a class of hash functions. In *ASIACRYPT'08, to appear*.
16. Willi Meier Jean-Philippe Aumasson and Raphael C.-W. Phan. The hash function family LAKE. In *FSE'08*. Springer, to appear, 2008.
17. Antoine Joux. Multicollisions in iterated hash functions. Application to cascaded constructions. In *CRYPTO'04*, volume 3152 of *LNCS*, pages 306–316. Springer, 2004.
18. Antoine Joux and Thomas Peyrin. Hash functions and the (amplified) boomerang attack. In *CRYPTO'07*, volume 4622 of *LNCS*, pages 244–263. Springer, 2007.

19. John Kelsey and Tadayoshi Kohno. Herding hash functions and the Nostradamus attack. In *EUROCRYPT'06*, volume 4004 of *LNCS*, pages 183–200. Springer, 2006.
20. John Kelsey and Bruce Schneier. Second preimages on n-bit hash functions for much less than 2^n work. In *EUROCRYPT'05*, volume 3494 of *LNCS*, pages 474–490. Springer, 2005.
21. Jongsung Kim, Seokhie Hong, and Bart Preneel. Related-key rectangle attacks on reduced aes-192 and aes-256. In *FSE'07*, volume 4593 of *LNCS*, pages 225–241. Springer, 2007.
22. Lars R. Knudsen, Christian Rechberger, and Søren S. Thomsen. The Grindahl hash functions. In *FSE'07*, volume 4593 of *LNCS*, pages 39–57. Springer, 2007.
23. Stefan Lucks. A failure-friendly design principle for hash functions. In *ASIACRYPT'05*, volume 3788 of *LNCS*, pages 474–494. Springer, 2005.
24. Stephane Manuel and Thomas Peyrin. Collisions on SHA-0 in one hour. In *FSE'08*. Springer, to appear, 2008.
25. Ralph C. Merkle. One way hash functions and DES. In *CRYPTO'89*, volume 435 of *LNCS*, pages 428–446. Springer, 1989.
26. Thomas Peyrin. Cryptanalysis of Grindahl. In *ASIACRYPT'07*, volume 4833 of *LNCS*, pages 551–567. Springer, 2007.
27. Bart Preneel, René Govaerts, and Joos Vandewalle. Hash functions based on block ciphers: A synthetic approach. In *CRYPTO'93*, volume 558 of *LNCS*, pages 368–378. Springer, 1993.
28. Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In *EUROCRYPT'05*, volume 3494 of *LNCS*, pages 19–35. Springer, 2005.
29. Muhammad Reza Z'aba, Haavard Raddum, Matt Henricksen, and Ed Dawson. Bit-pattern based integral attack. In *FSE'08*. Springer, to appear, 2008.

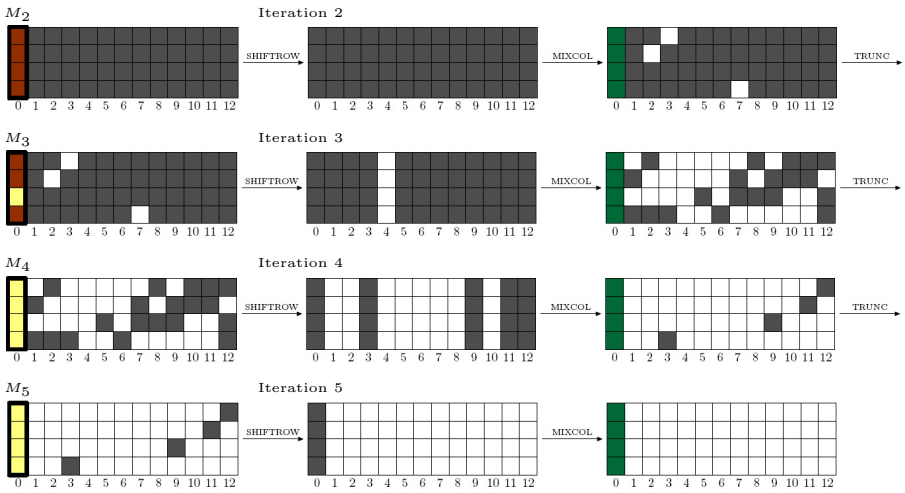

Fig. 3. Differential trail for GRINDAHL-256 (Table 1 (b)).

A Equivalence classes of differences

In order to reduce the complexity of the collision search one may consider not only linear differences but also more generic relations with the *equivalence property*. More formally, the following implications should hold:

$$\begin{cases} S \Delta S; \\ S_1 \Delta S_2 \;\Rightarrow\; S_2 \Delta S_1; \\ S_1 \Delta S_2,\, S_2 \Delta S_3 \;\Rightarrow\; S_1 \Delta S_3. \end{cases}$$

If Δ stands for a property of the sum of two states then we obtain a linear space of state differences.

B Structures of different size

In Section 4 we claimed that if the number of structures is constant than the smallest number of pairs is provided by the partition to structures of the same size. Indeed let m structures contain km states. Assume that the structures are of different size: a structure S' contains $k' > k$ states, and a structure S'' contains $k'' < k$ states. The number of pairs provided by the two structures is $k'(k'-1)/2 + k''(k''-1)/2$. Now move one state from the first structure to the second one. Then the partition results in

$$\frac{(k'-1)(k'-2)}{2} + \frac{(k''+1)k''}{2} = \frac{k'(k'-1)}{2} - \frac{2(k'-1)}{2} + \frac{k''(k''-1)}{2} + \frac{2k''}{2} = \\ = \frac{k'(k'-1)}{2} + \frac{k''(k''-1)}{2} + (k''-k'+1)$$

pairs. Since $k'' < k < k'$ we obtain that the resulting number of pairs is smaller than that of the initial partition.

We continue this procedure till all the structures are equal in size, and the number of pairs is monotonically decreasing. Thus we conclude that the partition to structures of equal size gives the smallest number of pairs.

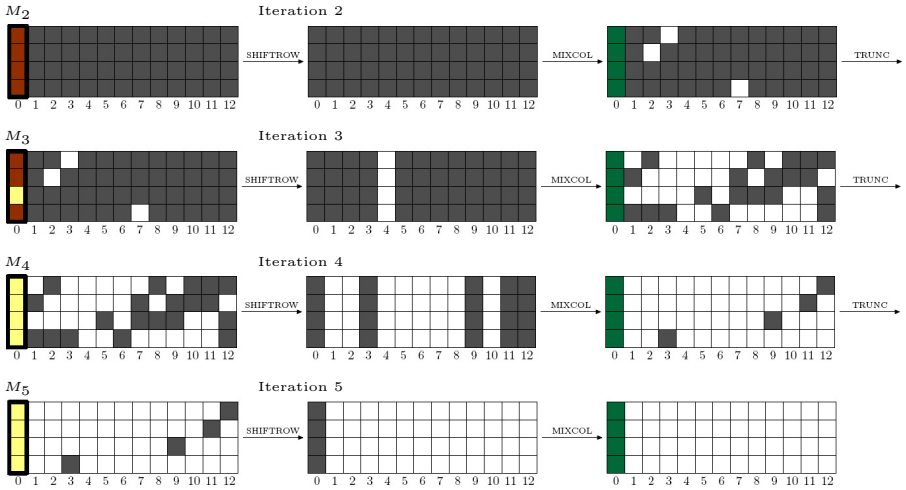

Fig. 4. Differential trail for Grindahl-512 (Table 2 (b)).

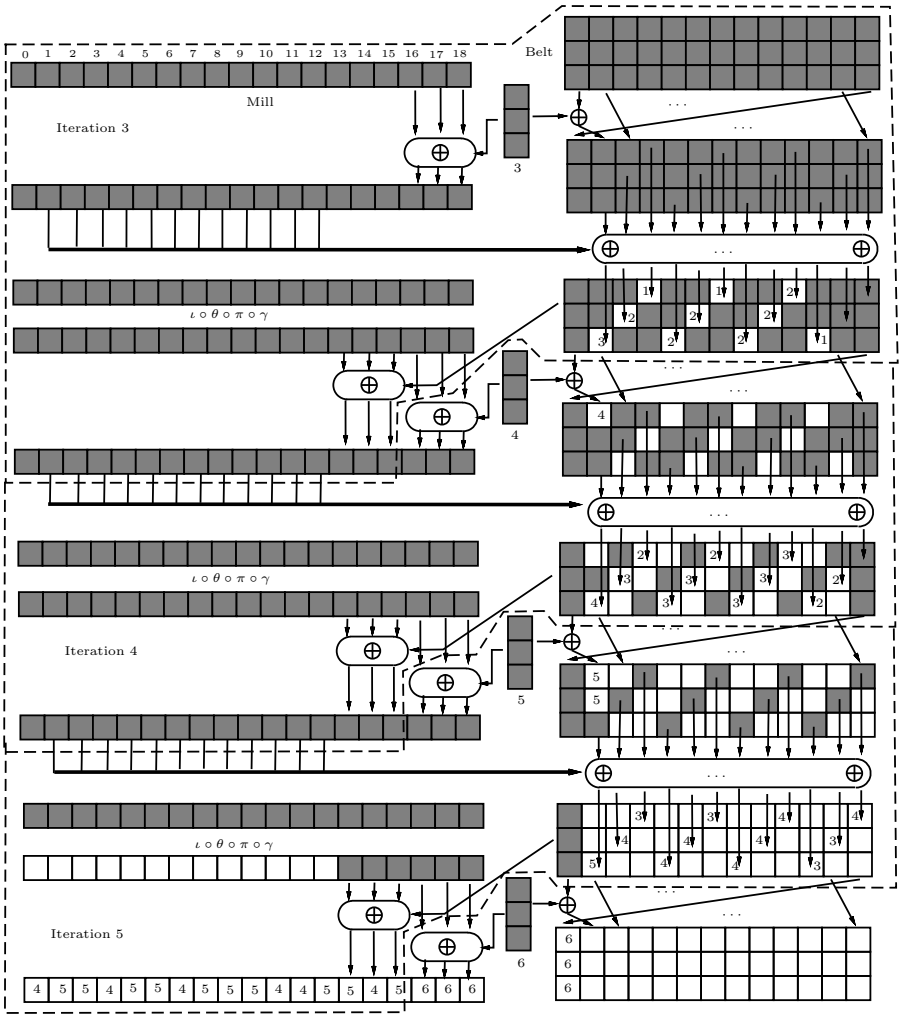

Fig. 5. Differential trail for RadioGatún.

It	Col	Cost	Message bytes											
			1				2				3			
2	2	1						B						
	3	1	B	A										
	7	1				D								
3	1	2									A			
	2	2										B		
	3	3					B	A						
	5	3					C		A					
	6	3						C	B					
	7	2								D				
	8	2							C					
	9	2	C	D	AC	BD								
	10	2												D
	11	2					D			A				
	12	1						D		B				
4	3	3									B	A		
	9	3					C	D	AC	BD				
	11	3									D			A
	12	3										B		

Table 5. Dependencies of the message block in the differential trail for Grindahl.

It	Index	Message words					
		1		2		3	
		Mill	Belt	Mill	Belt	Mill	Belt
3 Belt	(2,1)					ABC	C
	(1,2)			A	B		
	(0,3)	ABC	A				
	(2,4)			ABC			
	(1,5)			ABC			
	(0,6)	ABC					
	(2,7)			C			
	(1,8)			C			
	(0,9)			AB			
	(2,10)	ABC					
4 Belt	(1,2)					A	B
	(0,3)			ABC	A		
	(2,4)					ABC	
	(1,5)					ABC	
	(0,6)			ABC			
	(2,7)					C	
	(1,8)					C	
	(0,9)					AB	
	(2,10)			ABC			
	(1,11)			ABC			
5 Belt	(0,3)					ABC	A
	(0,6)					ABC	
	(2,10)					ABC	
	(1,11)					ABC	

It	Index	Message words			
		4		5	
		Mill	Belt	Mill	Belt
4 Belt	(0,1)		A		
	(2,1)		C		
5 Belt	(1,2)	A	B		
	(2,4)	ABC			
	(3,5)	ABC			
	(2,7)	C			
	(1,8)	C			
	(0,9)	AB			
	(0,12)	ABC			
5 Mill	0	ABC			
	3	ABC			
	6	ABC			
	10	ABC			
	11	ABC			
	14	ABC			

Table 6. Dependencies of the state words on the message words in the differential trail for RadioGatún.