Boomerang Distinguishers on MD4-Family: First Practical Results on Full 5-Pass HAVAL

Yu Sasaki

NTT Information Sharing Platform Laboratories, NTT Corporation
3-9-11 Midori-cho, Musashino-shi, Tokyo 180-8585 Japan
sasaki.yu@lab.ntt.co.jp

Abstract. In this paper, we study a boomerang attack approach on MD4-based hash functions, and present a practical 4-sum distinguisher against the compression function of the full 5-pass HAVAL. Our approach is based on the previous work by Kim *et al.*, which proposed the boomerang distinguisher on the encryption mode of MD4, MD5, and HAVAL in the related-key setting. Firstly, we prove that the differential path for 5-pass HAVAL used in the previous boomerang distinguisher contains a critical flaw and thus the attack cannot work. We then search for new differential paths. Finally, by using the new paths, we mount the distinguisher on the compression function of the full 5-pass HAVAL which generates a 4-sum quartet with a complexity of approximately 2^{11} compression function computations. As far as we know, this is the first result on the full compression function of 5-pass HAVAL that can be computed in practice. We also point out that the 4-sum distinguisher can also be constructed for other MD4-based hash functions such as MD5, 3-pass HAVAL, and 4-pass HAVAL. Our attacks are implemented on a PC and we present a generated 4-sum quartet for each attack target.

Keywords: boomerang attack, 4-sum distinguisher, hash, HAVAL.

1 Introduction

Hash functions are taking important roles in various aspects of the cryptography. After the breakthrough by Wang *et al.* [26,27] and through the SHA-3 competition [20], cryptanalysis against hash functions have been improved significantly.

The boomerang attack, which was proposed by Wagner [22], is a tool for the cryptanalysis against block-ciphers. At FSE2011, Biryukov *et al.* applied the boomerang attack for hash functions, and showed that a zero-sum distinguisher could be constructed on them [3], where zero-sum is a set of messages whose XOR is 0 and the XOR of their corresponding outputs is also 0. Lamberger and Mendel independently applied the boomerang attack on SHA-2 and obtained a significant improvement on the 4-sum distinguisher against its reduced-step compression function [10], where a k-sum is a set of k paired initial-values and messages such that the XOR of their outputs is 0. It seems that the boomerang attack is potentially very powerful against hash functions, and thus more investigation is required to understand their impact deeply. Note that at CRYPTO2007,

A. Miri and S. Vaudenay (Eds.): SAC 2011, LNCS 7118, pp. 1–18, 2012.

Joux and Peyrin proposed an (amplified) boomerang attack for SHA-1 [7]. They used the idea of the boomerang attack for the message modification technique in the collision attack, which the purpose is different from our research.

The boomerang attack on hash functions does not always discuss the security as the hash function. As done in [10], it often discusses the security of the compression function or the internal block-cipher. Although they do not impact to the security of the hash function immediately, such analyses are useful from several viewpoints; 1) The progress of the cryptanalysis, in other words, the security margin can be measured, 2) The attack could be used as a tool for different purposes in the future, e.g., a pseudo-collision attack on MD5 [4]. 3) The attack on a building-block may invalidate the security proof for the hash function. Specifically, hash functions using the PGV modes tend to have the reduction security by assuming the ideal behavior of the internal block-cipher.

MD4, which was proposed by Rivest in 1990 [13], is a hash function that is used as a base of various hash functions. MD4 has an interesting property in its message expansion. The sketch of its computation is as follows;

- Divide an input message block M into several message words m_0, $m_1, \ldots, m_{N_S-1}$.
- Iteratively apply a round function N_R times, where the round function consists of N_S steps.
- For N_S steps in each round, each of m_0 to m_{N_S-1} is used exactly once.
- The order of message words, in other words, the permutation of the message-word index may change for different rounds.

We call this type of the message expansion *message-words permutation*. MD4, MD5 [14], and HAVAL [32] are examples using the message-words permutation.

MD4, MD5, and HAVAL are now known to be vulnerable against various attacks. For example, Van Rompay *et al.* found collisions of 3-pass HAVAL in 2003 [21], and Wang *et al.* found collisions of MD4, MD5, and 3-pass HAVAL in 2004 [25,27]. The complexity of collision attacks were optimized to 2 for MD4 [18], 2^{10} for MD5 [29], 2^7 for 3-pass HAVAL [19,24], 2^{36} for 4-pass HAVAL [28,31], and 2^{123} for 5-pass HAVAL [31], where the unit of the complexity is one computation of the compression function. Note that, only the theoretical result is known for 5-pass HAVAL, and thus real collisions have not been found yet.

Theoretical preimage attacks are also presented. For example, [1,6,11] for MD4, [17] for MD5, [2,16] for 3-pass HAVAL, and [16] for 4-pass HAVAL. For 5-pass HAVAL, only the attack on 158-steps out of 160-steps is known [15].

Several researchers evaluated the security of the building block for these hash functions. Examples which analyzed full steps are [4,5] for MD5 and [8,9,30] for HAVAL. Among them, the work by Kim *et al.* [8,9], which applied the boomerang attack to distinguish their encryption modes from a random permutation in the related-key setting, is very powerful. They successfully distinguished these encryption modes with 2^6 queries for MD4, $2^{11.6}$ queries for MD5, and $2^{9.6}$ queries for 4-pass HAVAL. These attacks were implemented and an example of the boomerang quartet was presented for MD5. In addition, Kim *et al.* claimed that 5-pass HAVAL could also be distinguished with 2^{61} queries and the attack

Table 1. Comparison of the attack complexity

Attack	Target	MD4 (Time	Ref.)	MD5 (Time	Ref.)	HAVAL-3 (Time	Ref.)	HAVAL-4 (Time	Ref.)	HAVAL-5 (Time	Ref.)
Collision	Hash Function	2	[18]	2^{10}	[29]	2^{7}	[19,24]	2^{36}	[28,31]	2^{123}	[31]
Boomerang	Block-Cipher	2^{6}	[9]	$2^{11.6}$	[9]		-	$2^{9.6}$	[9]	2^{61}	[9]
Boomerang	Compress. Func.		-	2^{10}	Ours	2^{4}	Ours	2^{11}	Ours	2^{11}	Ours

was partially verified by implementing it for reduced-round variants. Note that although Kim *et al.* pointed out the vulnerability of the MD4-based structure against the boomerang attack, the analysis on 5-pass HAVAL is still infeasible.

Our Contributions

In this paper, we study the boomerang attack approach on MD4-based hash functions. We use the differential path for the boomerang attack to construct the 4-sum distinguisher on the compression function, while Kim *et al.* [9] used the boomerang path to distinguish its encryption mode from a random permutation. For both of our approach and the one in [9], the core of the attack is the existence of the differential path suitable for the boomerang attack. However, because the attack scenario is different, the procedure to optimize the attack is quite different. We first collect various techniques for the boomerang attack on hash functions from several papers (mainly [3,9,10]), and summarize the attack framework.

We then revisit the differential path for the boomerang attack against 5-pass HAVAL in [9]. On the contrary to the authors' claim, we prove that the differential path in [9] contains a critical flaw and thus the attack cannot work.

We then search for new differential paths for the boomerang attack and construct the attack procedure optimized for attacking the compression function. Finally, by using the new paths, we mount the distinguisher on the full compression function of 5-pass HAVAL which generates a 4-sum quartet with a complexity of 2^{11} compression function computations. The attack complexity is summarized in Table 1. As far as we know, this is the first result on the full 5-pass HAVAL that can be computed in practice. The attack is implemented on a PC and we present a generated 4-sum quartet.

Note that as long as the good boomerang differential path is available, 4-sum distinguishers can be constructed on the compression function. Then, with the differential paths in [9], we attack MD5, 3-pass HAVAL, and 4-pass HAVAL with a complexity of 2^{10}, 2^{4} and 2^{11} compression function computations, respectively. We present generated 4-sums in Appendix B.

Paper Outline

We describe the specification of HAVAL and clarify the terminology in Sect. 2. We summarize previous work in Sect. 3. We give a summary of techniques for the boomerang attack on hash functions in Sect. 4. We demonstrate a dedicate attack on 5-pass HAVAL in Sect. 5. Finally, we conclude this paper in Sect. 6.

Table 2. Word-wise rotation $\phi_{x,y}$ of HAVAL

	x_6 x_5 x_4 x_3 x_2 x_1 x_0		x_6 x_5 x_4 x_3 x_2 x_1 x_0		x_6 x_5 x_4 x_3 x_2 x_1 x_0
	$\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$		$\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$		$\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$
$\phi_{3,1}$	x_1 x_0 x_3 x_5 x_6 x_2 x_4	$\phi_{4,1}$	x_2 x_6 x_1 x_4 x_5 x_3 x_0	$\phi_{5,1}$	x_3 x_4 x_1 x_0 x_5 x_2 x_6
$\phi_{3,2}$	x_4 x_2 x_1 x_0 x_5 x_3 x_6	$\phi_{4,2}$	x_3 x_5 x_2 x_0 x_1 x_6 x_4	$\phi_{5,2}$	x_6 x_2 x_1 x_0 x_3 x_4 x_5
$\phi_{3,3}$	x_6 x_1 x_2 x_3 x_4 x_5 x_0	$\phi_{4,3}$	x_1 x_4 x_3 x_6 x_0 x_2 x_5	$\phi_{5,3}$	x_2 x_6 x_0 x_4 x_3 x_1 x_5
-	-	$\phi_{4,4}$	x_6 x_4 x_0 x_5 x_2 x_1 x_3	$\phi_{5,4}$	x_1 x_5 x_3 x_2 x_0 x_4 x_6
-	-	-	-	$\phi_{5,5}$	x_2 x_5 x_0 x_6 x_4 x_3 x_1

Table 3. Message-words permutation. The first column shows the round numbers.

	index for each round																														
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	5	14	26	18	11	28	7	16	0	23	20	22	1	10	4	8	30	3	21	9	17	24	29	6	19	12	15	13	2	25	31	27
3	19	9	4	20	28	17	8	22	29	14	25	12	24	30	16	26	31	15	7	3	1	0	18	27	13	6	21	10	23	11	5	2
4	24	4	0	14	2	7	28	23	26	6	30	20	18	25	19	3	22	11	31	21	8	27	12	9	1	29	5	15	17	10	16	13
5	27	3	21	26	17	11	20	29	19	0	12	7	13	8	31	10	5	9	14	30	18	6	28	24	2	23	16	22	4	1	25	15

2 Preliminaries

2.1 Specification of HAVAL

HAVAL [32] uses a narrow-pipe Merkle-Damgård structure. An input message M is padded to be a multiple of the block-size (1024 bits), and then divided into message blocks $(M_0, M_1, \ldots, M_{L-1})$. Then, chaining variable H_i starting from the pre-specified initial value H_0 is iteratively updated by the compression function CF; $H_{i+1} \leftarrow \mathrm{CF}(H_i, M_i)$, for $i = 0, 1, \ldots, L-1$. Finally, H_L is the hash value of M. HAVAL can produce a hash value of smaller sizes by using the output tailoring function. Because our attack target is the compression function, we omit the description for the padding and the output tailoring function.

The size of chaining variables is 256 bits. Inside the compression function, M_i is divided into thirty-two 32-bit message words $(m_0, m_1, \ldots, m_{31})$. Three algorithms are prepared for HAVAL; 3-pass, 4-pass, and 5-pass. The number of rounds for 3-pass, 4-pass, and 5-pass are 3 rounds (96 steps), 4 rounds (128 steps), and 5 rounds (160 steps), respectively.

Let us denote a 256-bit state before step j by p_j and denote p_j by eight 32-bit variables $Q_{j-7} \| Q_{j-6} \| Q_{j-5} \| Q_{j-4} \| Q_{j-3} \| Q_{j-2} \| Q_{j-1} \| Q_j$. The step function R_j computes Q_{j+1} as follows:

$$Q_{j+1} \leftarrow (Q_{j-7} \ggg 11) + (\Phi_j(\phi_{x,y}(Q_{j-6}, Q_{j-5}, \ldots, Q_j)) \ggg 7) + m_{\pi(j)} + k_j,$$

where $\phi_{x,y}$ is a word-wise rotation for x-pass HAVAL in round y defined in Table 2, and $\pi(j)$ is shown in Table 3.

2.2 Technical Terminologies

In this paper, we discuss *differences* of several computations. Let us consider the two computations $H_{i+1} \leftarrow \mathrm{CF}(H_i, M_i)$ and $H'_{i+1} \leftarrow \mathrm{CF}(H'_i, M'_i)$. The message

difference, input chaining-variable difference, and output difference are defined as $M_i \oplus M'_i$, $H_i \oplus H'_i$, and $H_{i+1} \oplus H'_{i+1}$, respectively. Similarly, the difference of two computations is defined as XOR of corresponding states.

The transition of the difference of internal state is described by several terms such as *differential path*, *differential trail*, and *differential characteristic*. As far as we know, the term *differential characteristic* was firstly used in the context of the symmetric-key cryptography. However, in the context of the hash function analysis, the term *differential path* seems to be used more frequently e.g., [26,27]. To follow this convention, in this paper, we use the term *differential path*.

When the input and output differences are fixed, we often consider all possible differential paths connecting them. A set of all possible differential paths is called *a differential* or *multiple-paths*. In this paper, we use the term *differential*.

3 Related Work

3.1 Boomerang Attack

The boomerang attack was proposed by Wagner [22] as a tool for attacking block-ciphers. The attack is a chosen-plaintext and adaptively chosen-ciphertext attack. It can be regarded as a type of the second-order differential attack. In this attack, the attacker divides the target cipher E into two parts E_1 and E_2 such that $E(\cdot) = E_2 \circ E_1(\cdot)$. Let us denote the differential for E_1 by $\Delta \rightarrow \Delta^*$ and for E_2 by $\nabla^* \rightarrow \nabla$. The differences $\Delta, \Delta^*, \nabla^*$, and ∇ are chosen by the attacker at offline. The attack procedure is as follows;

1. The attacker first prepares a plaintext P^1 and compute $P^2 \leftarrow P^1 \oplus \Delta$.
2. P^1 and P^2 are passed to the encryption oracle and the attacker obtains the corresponding ciphertexts C^1 and C^2.
3. The attacker prepares the paired ciphertexts $C^3 \leftarrow C^1 \oplus \nabla$ and $C^4 \leftarrow C^2 \oplus \nabla$, and passes them to the decryption oracle.
4. Finally, the attacker checks whether or not P^3 and P^4 has the difference Δ.

Assume that the probability for the differentials for E_1 and E_2 are p and q, respectively. Then, $\Pr[P^3 \oplus P^4 = \Delta]$ is expressed as p^2q^2. In the end, we can conclude that if E can be divided into two parts with a high-probability differential, the boomerang attack is very efficient.

For a long time, it was assumed that the differentials for E_1 and E_2 can be chosen independently. In 2009, Murphy pointed out that this was not sufficient, and discovered several examples of this case for DES and AES [12].

3.2 Boomerang Distinguishers for Hash Functions

In 2011, Biryukov *et al.* pointed out that the zero-sum distinguisher can be constructed by applying the boomerang attack on hash functions [3]. In the boomerang attack, $P^1 \oplus P^2 = \Delta$ and $P^3 \oplus P^4 = \Delta$. Therefore, $P^1 \oplus P^2 \oplus P^3 \oplus P^4 = \Delta \oplus \Delta = 0$. Similarly, $C^1 \oplus C^2 \oplus C^3 \oplus C^4 = \nabla \oplus \nabla = 0$. Hence, by starting

from a pair of plaintexts P^1 and P^2 such that $P^1 \oplus P^2 = \Delta$, the attacker finds a zero-sum quartet with a complexity of $(p^2q^2)^{-1}$. [3] considered the attack starting from the border state between E_1 and E_2, and optimized the attack by applying the message modification technique [26,27]. [3] also computed the complexity to find a zero-sum in a random function for n-bit output. They explained that by starting from two paired plaintexts (resp. ciphertexts) with pre-specified differences, the complexity to find a zero-sum quartet of ciphertexts (resp. plaintexts) is $2^{\frac{n}{2}}$. Note that [3] considered the differential path rather than the differential for their attack. In fact, to apply the message modification technique, considering a differential path is much easier than a differential. Also note that [3] considered the observation by Murphy [12]. They had to give up combining the best differential paths for E_1 and E_2 due to their dependency.

In 2011, Lamberger and Mendel independently applied the boomerang 4-sum for the SHA-2 compression function [10]. They claimed that the complexity for finding a 4-sum quartet in a random function without any limitation on the input is $2^{\frac{n}{3}}$ by using the generalized birthday attack [23].

3.3 Boomerang on Encryption Modes of MD4, MD5, and HAVAL

Kim *et al.* applied the boomerang attack approach on the encryption modes of MD4, MD5, and HAVAL in the related-key model [9]. They proposed boomerang distinguishers with 2^6 queries for MD4, $2^{11.6}$ queries for MD5, and $2^{9.6}$ queries for 4-pass HAVAL. These attacks were verified by the machine experiment. Furthermore, they proposed differential paths for 5-pass HAVAL, and claimed that the boomerang distinguisher with 2^{61} queries was possible. They also claimed that this distinguisher was partially verified with an experiment on a reduced-round variant which was truncated for the first and the last several rounds.

Our attack framework is close to the one discussed in Sect. 3.2, but use the differential paths in [9] as a tool. In fact, we use the same paths as [9] for MD5 and 4-pass HAVAL. However, we need new differential paths for 5-pass HAVAL due to the flaw which we will point out in Sect. 5.

4 Summary of Boomerang Attack on Hash Function

Because various techniques for the boomerang attack on hash functions are distributed in several papers, we summarize the attack framework. Therefore, most of the contents in this section were originally observed by [3,9,10].

The attack can be divided into five phases; 1) message differences (ΔM), 2) differential paths and sufficient conditions (DP), 3) contradiction between two paths (CP), 4) message modification (MM), and 5) amplified probability (AP). We explain each phase with several observations specific to message-words permutation hash functions.

4.1 Message Differences (ΔM)

A generic strategy for an N_R-round hash function is illustrated in Fig. 1. For $N_R = 4$, the first two and last two rounds are regarded as E_1 and E_2, respectively.

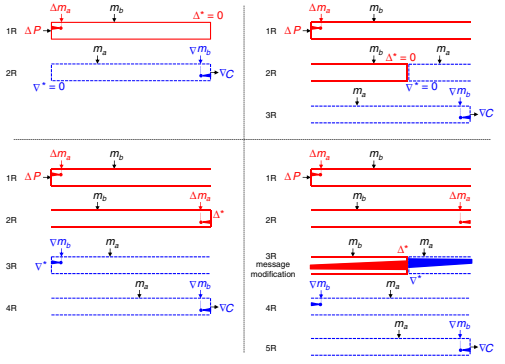

Fig. 1. Strategies for differential path

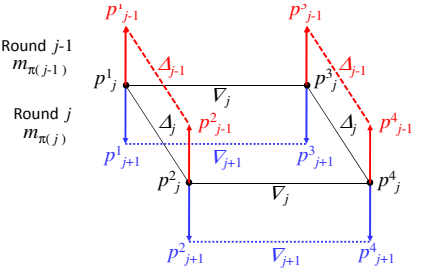

Fig. 2. Message search procedure

For E_1, we search for the message word which appear in an early step in the first round and in a late step in the second round. Then, the message difference is propagated until the beginning and end of E_1. The same strategy is applied for E_2. Because the differential paths for both of E_1 and E_2 are short, they are satisfied with high probability even without the message modification technique.

For $N_R = 5$, we extend the differential paths for the 4-round attack by a half more round. As shown in Fig. 1, the paths become long and hard to satisfy by the naive search. Wang *et al.* showed that the differential path for one round can be satisfied for free by the message modification technique [26,27]. Hence, with these techniques, 5 rounds can be attacked. In this paper, we denote the differential path between the end of round 2 and the beginning of round 4 by *inside path*, and the differential paths in round 1 and round 5 by *outside paths*.

4.2 Differential Paths and Sufficient Conditions (*DP*)

Based on the strategy in Fig. 1, we construct differential paths and conditions for chaining variables. These procedures are basically the same as the ones for previous collision attacks. We only list the differences of the path search procedure between the collision and boomerang attacks.

- If the feed-forward operation is performed by a modular addition ($H = P \boxplus C$), the attacker should introduce the additive difference among a quartet. in such a case, the difference for the quartet is defined as $(H^4 \boxminus H^3) \boxminus (H^2 \boxminus H^1) = H^1 \boxminus H^2 \boxminus H^3 \boxplus H^4$.
- The number of conditions must be minimized because setting x more conditions will increase the complexity by a factor of 2^{2x} rather than 2^x.
- In the middle round, we apply the message modification, and thus even complicated paths can be satisfied. However, by taking into account the Phase *CP*, the paths should be simplified around the border of two paths.

– If active-bit positions are concentrated around the MSB in both of E_1 and E_2 for the optimization, the risk of the contradiction of two paths will increase.

4.3 Contradiction between Two Paths (*CP*)

As Murphy pointed out [12], differential paths for E_1 and E_2 are not independent, and thus we need to check that any contradiction does not occur. As far as we know, no systematic method is known to check the contradiction. However, it can be said that the attacker at least needs to check the following conditions.

Condition 1. E_1 and E_2 do not require to fix the same bit to different values.
Condition 2. E_1 (resp. E_2) does not require to fix the value of an active bit for E_2 (resp. E_1).

The first case is obviously in contradiction. In the second case, even if the condition is satisfied between one pair, say P^1 and P^2, the condition is never satisfied for the other pair P^3 and P^4 due to the difference between P^1 and P^3.

As discussed in Sect. 4.2, if many bits are activated or active-bit positions are concentrated around MSB, the contradiction comes to occur more easily. Regarding HAVAL, due to the large word-size (32 bits) and the different rotation constants in forward and backward, the contradiction is less likely to occur. If the word size is smaller or if the similar rotation constants are used such as BLAKE, the contradiction seems to occur with a high probability.

If the contradiction occurs, rotating the path for either E_1 or E_2 by several bits may avoid the contradiction (though the efficiency becomes worse).

4.4 Message Modification (*MM*)

Let us denote a quartet of texts at step j forming the boomerang structure by $(p_j^1, p_j^2, p_j^3, p_j^4)$. The difference for E_1 is denoted by Δ, which is considered between p_1 and p_2 and between p_3 and p_4. The difference for E_2 is denoted by ∇, which is considered between p_1 and p_3 and between p_2 and p_4. We call conditions on the path for E_1 Δ-conditions, and for E_2 ∇-conditions.

The message search procedure is described in Fig. 2. The attack starts from the state at the border between E_1 and E_2. Let us denote this step by b. First, we set a chaining-variables quartet $(p_b^1, p_b^2, p_b^3, p_b^4)$ so that both of Δ- and ∇-conditions are satisfied. We then perform the backward computation for E_1 and forward computation for E_2 as shown in Alg. 1 and Alg. 2, respectively.

These procedures are computed until the inside path is ensured to be satisfied with probability of 1. This often occurs before all message words are fixed by the above procedure. Therefore, towards the outside paths, we do as follows.

– Assume that several message-words are not determined even after the inside path are ensured. Then, we never modify the message-words and chaining-variables related to the inside path, and compute the outside paths by randomly choosing the message-words not used for the inside path.

This enables us to iterate the outside computation with keeping the inside path satisfied. Hence, the complexity for satisfying the inside path can be ignored.

Algorithm 1. Message search procedure for step j in the backward direction

Input: Inside differential path and a chaining-variables quartet $(p^1_{j+1}, p^2_{j+1}, p^3_{j+1}, p^4_{j+1})$
Output: A message-words quartet $(m^1_{\pi(j)}, m^2_{\pi(j)}, m^3_{\pi(j)}, m^4_{\pi(j)})$ and a chaining-variables quartet $(p^1_j, p^2_j, p^3_j, p^4_j)$
1: Choose the value of p^1_j to satisfy conditions (Δ-conditions) for p^1_j. Then, compute $m^1_{\pi(j)}$ by solving equation R_j.
2: Compute $m^2_{\pi(j)}$, $m^3_{\pi(j)}$, and $m^4_{\pi(j)}$ with the specified differences ΔM and ∇M.
3: Compute p^3_j with $m^3_{\pi(j)}$ and check if all conditions (Δ-conditions) for p^3_j are satisfied. If so, compute p^2_j and p^4_j. If not, repeat the procedure with different p^1_j.

Algorithm 2. Message search procedure for step j in the forward direction

Input: Inside differential path and a chaining-variables quartet $(p^1_j, p^2_j, p^3_j, p^4_j)$
Output: A message-words quartet $(m^1_{\pi(j)}, m^2_{\pi(j)}, m^3_{\pi(j)}, m^4_{\pi(j)})$ and a chaining-variables quartet $(p^1_{j+1}, p^2_{j+1}, p^3_{j+1}, p^4_{j+1})$
1: Choose the value of p^1_{j+1} to satisfy conditions (∇-conditions) for p^1_{j+1}. Then, compute $m^1_{\pi(j)}$ by solving R_j.
2: Compute $m^2_{\pi(j)}$, $m^3_{\pi(j)}$, and $m^4_{\pi(j)}$ with the specified differences ΔM and ∇M.
3: Compute p^2_{j+1} and check if all conditions (∇-conditions) for p^2_{j+1} are satisfied. If so, compute p^3_{j+1} and p^4_{j+1}. If not, repeat the procedure with different p^1_{j+1}.

4.5 Amplified Probability (AP)

Amplified probability is the probability that each outside path results in the 4-sum. We consider the differential to estimate this probability. This is often estimated by an experiment. Alg. 3 shows how to compute the amplified probability AP^{Back} for the first round (from step $j-1$ to step 0). The amplified probability for the final round AP^{For} is similarly computed. Note that as long as the operation (usually either XOR or the modular addition) used to compute the 4-sum in this experiment and used in the feed-forward is identical, the success probability after the feed-forward is preserved as $AP^{Back} \times AP^{For}$.

5 4-Sum Distinguisher on 5-Pass HAVAL

We start from pointing out the flaw of the previous differential path (Sect. 5.1), and construct new differential paths (Sect. 5.2). We then explain the attack based on the discussion in Sect. 4 and finally show the experimental results.

Algorithm 3. Evaluation of the amplified probability

Input: Outside differential path
Output: Amplified probability of the outside differential path
1: Randomly choose a chaining-variables quartet $(p^1_j, p^2_j, p^3_j, p^4_j)$ and message-words used in steps $j-1$ to 0 with appropriate message differences ΔM and ∇M.
2: Compute this quartet until step 0 and check whether the 4-sum is constructed.
3: Repeat the above for an enough amount of times, and calculate the probability.

Table 4. Differential path and conditions for 5-pass HAVAL [9]. e_z represents that only z-th bit has a difference.

j	Output diff. at step j $(\Delta Q_{j-7}, \ldots, \Delta Q_j)$	Equation for Φ_j $\Phi_j(\phi_{5,3}(Q_{j-6}, \ldots, Q_j))$	Conditions on 20th bit for $\Phi_j = 0$
	$(0,0,0,0,0,0,0,e_{20})$		
70	$(0,0,0,0,0,0,e_{20},0)$	$\Phi_{70}(\ Q_{68},\ Q_{64}, \Delta Q_{70},\ Q_{66},\ Q_{67},\ Q_{69},\ Q_{65})$	$Q_{69} = 0$
71	$(0,0,0,0,0,e_{20},0,0)$	$\Phi_{71}(\ Q_{69},\ Q_{65},\ Q_{71},\ Q_{67},\ Q_{68}, \Delta Q_{70},\ Q_{66})$	$Q_{67}Q_{68} \oplus Q_{71} = 0$
72	$(0,0,0,0,e_{20},0,0,0)$	$\Phi_{72}(\Delta Q_{70},\ Q_{66},\ Q_{72},\ Q_{68},\ Q_{69},\ Q_{71},\ Q_{67})$	$Q_{68} = 0$
73	$(0,0,0,e_{20},0,0,0,0)$	$\Phi_{73}(\ Q_{71},\ Q_{67},\ Q_{73},\ Q_{69}, \Delta Q_{70},\ Q_{72},\ Q_{68})$	$Q_{72}Q_{69} \oplus Q_{67} = 0$
74	$(0,0,e_{20},0,0,0,0,0)$	$\Phi_{74}(\ Q_{72},\ Q_{68},\ Q_{74}, \Delta Q_{70},\ Q_{71},\ Q_{73},\ Q_{69})$	$Q_{73} \oplus Q_{71} \oplus Q_{72}Q_{69} = 0$
75	$(0,e_{20},0,0,0,0,0,0)$	$\Phi_{75}(\ Q_{73},\ Q_{69},\ Q_{75},\ Q_{71},\ Q_{72},\ Q_{74}, \Delta Q_{70})$	$Q_{71} = 1$
76	$(e_{20},0,0,0,0,0,0,0)$	$\Phi_{76}(\ Q_{74}, \Delta Q_{70},\ Q_{76},\ Q_{72},\ Q_{73},\ Q_{75},\ Q_{71})$	$Q_{73} = 0$
77	$(0,0,0,0,0,0,0,e_9)$	$\Phi_{77}(\ Q_{75},\ Q_{71},\ Q_{77},\ Q_{73},\ Q_{74},\ Q_{76},\ Q_{72})$	-

5.1 Proving Flaw of Previous Differential Path

We point out that the differential path for the third round in [9] cannot work. Note that the verifying experiment by [9] is for the reduced-round variants which are truncated for the first and the last several rounds [9, Sect. 4.2]. Hence, our claim does not contradict to the partial verification by [9].

The differential path during 8 steps (steps 70-77) in the third round is shown in Table 4. The authors claimed that the path could be satisfied with a probability of 2^{-7}. The necessary and sufficient condition for satisfying this path with a probability of 2^{-7} is that the difference in Q_{70} will not propagate through Φ_j. Φ_j for these steps is a bit-wise Boolean function expressed as

$$\Phi_j(x_6, x_5, x_4, x_3, x_2, x_1, x_0) = x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_5 \oplus x_3x_6 \oplus x_0x_3 \oplus x_0.$$

Conditions to achieve the path were not explained in [9]. We first derive the necessary and sufficient conditions to achieve the path, which are shown in Table 4.

Proof. For steps 70, 75, and 76, we have conditions $Q_{69} = 0$, $Q_{71} = 1$, and $Q_{73} = 0$. Then, the left-hand side of the condition for step 74 becomes $0 \oplus 1 \oplus (Q_{72} \cdot 0) = 1$, which contradicts to the condition that this value must be 0. □

We verified the above proof with a small experiment;

1. Randomly choose the values of Q_{63} to Q_{70} and $m_{\pi(70)}$ to $m_{\pi(77)}$.
2. Set $Q'_z \leftarrow Q_z$ for $z = 63, 64, \ldots, 70$. Then compute $Q'_{70} \leftarrow Q_{70} \oplus$ `0x00100000`.
3. Compute until step 77 and check whether or not the path is satisfied.

With 2^{30} trials, the differential path in Table 4 was not satisfied. This contradicts to the claim in [9], which the path is satisfied with a probability 2^{-7}.

5.2 Constructing New Differential Paths

We reconstruct the attack based on the strategy explained Sect. 4.1. For Phase ΔM, we confirmed that the message differences in [9] (Δm_2 = `0x80000000`,

Table 5. Boomerang path construction for 5-pass HAVAL

	index for each round																													
1	0	1	②	3	④	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	←	Δ															constant															
2	5	14	26	18	11	28	7	16	0	23	20	22	1	10	④	8	30	3	21	9	17	24	29	6	19	12	15	13	②	25	31	27
													constant																Δ	→		
3	19	9	④	20	28	17	8	22	29	14	25	12	24	30	16	26	31	15	7	3	1	0	18	27	13	6	21	10	23	11	5	②
							message modification														message modification										
4	24	④	0	14	②	7	28	23	26	6	30	20	18	25	19	3	22	11	31	21	8	27	12	9	1	29	5	15	17	10	16	13
	←	∇														constant																
5	27	3	21	26	17	11	20	29	19	0	12	7	13	8	31	10	5	9	14	30	18	6	28	24	②	23	16	22	④	1	25	15
													constant																∇	→		

$\Delta m_x = 0$ for $x \neq 2$, $\nabla m_4 =$ `0x80000000`, $\nabla m_x = 0$ for $x \neq 4$) match the strategy in Fig. 1, and thus we use the same message differences. This is described in Table 5.

For Phase *DP*, we need to specify how the differences will propagate to chaining variables. We describe our path search algorithm in Appendix A. The searched paths and conditions for chaining variables are given in Table 6.

For Phase *CP*, the overlap of the conditions and active-bit positions in Table 6 must be checked. According to Table 6, the conditions 1 and 2 described in Sect. 4.3 are satisfied. Note that as long as the step function is similar to MD4, MD5, or HAVAL, the active bit positions and conditions for Δ and ∇ tend to be different due to the asymmetric rotation constants in forward and backward directions. In fact, for all differential paths in [9] and ours, the best differential paths which were independently computed could be combined.

5.3 Attack Procedure

In Phase *MM*, we optimize the attack complexity based on the strategy in Sect. 4.4. The detailed procedure is given in Alg. 4.

As shown in Table 5 the inside path starts from step 60 and ends at step 97. Several words $(w_2, w_{25}, w_{31}, w_{27}, w_{24}, w_4)$ are used twice and we need a special attention. However, as shown in Table 6, conditions are set only on Q_{58} to Q_{93}, and thus, the second-time use of these words outside of Q_{58} to Q_{93} always succeed for any value. Hence, these values are chosen for satisfying conditions for Q_{58} to Q_{93}. After we satisfy all conditions for Q_{58} to Q_{93}, 4 message words w_{19}, w_{11}, w_5, and w_2 are still unfixed. Therefore, we can iterate the outside path search without changing the inside path up to 2^{128} times, which is enough to satisfy the outside paths.

The complexity of the message modification for satisfying the inside path (up to Step 3 in Alg. 4) is negligible. Hence, the attack complexity is only the iterative computations for satisfying the outside paths (Steps 4–11 in Alg. 4). This complexity is evaluated by considering the amplified probability in Phase *AP*, which will be explained in the following section.

Algorithm 4. Attack procedure with the message modification

Input: Entire differential paths and conditions
Output: A quartet of (H_{i-1}, M_{i-1}) satisfying the 4-sum property

1: Randomly choose the values of p^1_{80}, p^2_{80}, p^3_{80} and p^4_{80} so that the differences and conditions (both of Δ and ∇) in Table 6 can be satisfied. Note that, choosing p^x_{80} means choosing eight 32-bit variables Q^x_{80}, Q^x_{79}, Q^x_{78}, Q^x_{77}, Q^x_{76}, Q^x_{75}, Q^x_{74}, and Q^x_{73}.
2: Apply the backward computation in Alg. 1 to obtain p^1_{65}, p^2_{65}, p^3_{65} and p^4_{65}. This fixes chaining variables up to Q^x_{58} and message words from $m_{\pi(79)}$ to $m_{\pi(65)}$.
3: Apply the forward computation in Alg. 2 to obtain p^1_{93}, p^2_{93}, p^3_{93} and p^4_{93}. This fixes chaining variables up to Q^x_{93} and message words from $m_{\pi(80)}$ to $m_{\pi(92)}$.
`//End of the message modification for the inside path`
4: **while** a 4-sum quartet of the compression function output is not found **do**
5: Randomly choose the values of message-words quartet for $m_{\pi(93)} = m_{11}$, $m_{\pi(94)} = m_5$, and $m_{\pi(95)} = m_2$ with the message difference on m_2, and compute a chaining-variables quartet until p^1_{98}, p^2_{98}, p^3_{98} and p^4_{98}.
6: Randomly choose the values of message-words quartet for $m_{\pi(64)} = m_{19}$, and compute a chaining-variables quartet until p^1_{60}, p^2_{60}, p^3_{60} and p^4_{60}.
7: Compute a chaining-variables quartet until p^1_0, p^2_0, p^3_0 and p^4_0 in backward and p^1_{160}, p^2_{160}, p^3_{160} and p^4_{160} in forward.
8: **if** $(p^1_0 \boxplus p^1_{160}) \boxminus (p^2_0 \boxplus p^2_{160}) \boxminus (p^3_0 \boxplus p^3_{160}) \boxplus (p^4_0 \boxplus p^4_{160}) = 0$ **then**
9: **return** $(p^1_0, p^2_0, p^3_0, p^4_0)$ and (M^1, M^2, M^3, M^4)
10: **end if**
11: **end while**

Algorithm 5. Differential path search algorithm for E_1 from step 60 to step 79

Input: Message difference ΔM, where $\Delta m_2 =$ `0x80000000` and $\Delta m_x = 0$ for $x \neq 2$
Output: Differences of each chaining variable between step 60 and step 79

1: Initialize $tempHD \leftarrow 0$
2: **for** $x = 53$ **to** 60 **do**
3: $Q_x \leftarrow$ a randomly chosen value
4: $Q'_x \leftarrow Q_x$
5: **end for**
6: **for** $x = 60$ **to** 79 **do**
7: $m_{\pi(x)} \leftarrow$ a randomly chosen value
8: $m'_{\pi(x)} \leftarrow m_{\pi(x)} \oplus \Delta M$
9: Compute Q_{x+1} and Q'_{x+1}
10: $tempHD \leftarrow tempHD + HW(Q_{x+1} \oplus Q'_{x+1})$
11: **if** $tempHD > 10$ **then**
12: `goto` step 1
13: **end if**
14: **end for**
15: **print** $Q_y \oplus Q'_y$ for $y = 61, 62, \ldots, 80$

Table 6. New differential paths and conditions for 5-Pass HAVAL. $[z] = 0, [z] = 1$ are conditions on the value of z-th bit of the chaining variable. For the first and last several steps, we do not fix a particular difference for the amplified probability. The difference is considered in XOR. In some cases, we need conditions on the sign of the difference. $[z] = 0+, [z] = 1-$ mean the value is first fixed to 0 (resp. 1) and change to 1 (resp. 0) after the difference is inserted.

Path for E_1 with Δm_2 = 0x80000000				Path for E_2 with ∇m_4 = 0x80000000				
j	ΔQ_j	Conditions on Q_j	Δm	j	∇Q_j	Conditions on Q_j	∇m	$m_{\pi(j)}$
-7	AP	AP		-7				m_1
-6	AP	AP		-6				m_0
-5	AP	AP	0x80000000	-5				m_2
-4				-4				m_3
...	...	...	...	...	...	...	...	...
52				52				m_{13}
53			0x80000000	53				m_2
54				54				m_{25}
55				55				m_{31}
56				56				m_{27}
57				57				m_{19}
58		[31]=0		58				m_9
59		[31]=0		59				m_4
60		[31]=0		60				m_{20}
61	0x80000000			61				m_{28}
62		[31]=0		62				m_{17}
63		[31]=0		63				m_8
64		[31,24]=0		64				m_{22}
65		[24]=0		65				m_{29}
66		[24,20]=0		66				m_{14}
67	0x01000000	[20]=0		67				m_{25}
68		[24,20]=0		68				m_{12}
69	0x00100000	[24]=0		69				m_{24}
70		[24,20,17]=0		70				m_{30}
71		[20,17]=0		71				m_{16}
72		[24,20,17]=0		72				m_{26}
73	0x00020000	[17]=1−		73				
74		[20,17]=0		74	0x00000001	[0]=1−		
75		[17,9]=0		75		[18]=0		
76		[17,9]=0		76		[18]=0		start
77	0x00000200	[9]=0+		77		[18,0]=0		step
78		[17,10,9]=0		78	0x00040000	[21]=0,[18]=0+		
79	0x00000400	[10]=1−		79		[21]=0,[18]=1		
80				80		[21,18]=0		
81				81		[21,18,14]=0		m_{31}
82				82	0x00200000	[14]=0		m_{15}
83				83		[21]=1,[14]=0		m_7
84				84	0x00004000	[21]=0		m_3
85				85		[21]=0,[14]=1		m_1
86				86		[14]=0		m_0
87				87		[14,10]=0		m_{18}
88				88		[10]=0		m_{27}
89				89		[10]=0		m_{13}
90				90	0x00000400			m_6
91				91		[10]=1		m_{21}
92				92		[10]=1		m_{10}
93				93		[10]=1		m_{23}
94				94				m_{11}
95				95				m_5
96				96				m_2
97				97				m_{24}
98				98			0x80000000	m_4
99				99				m_0
...	...	...	...	...	...	...	...	...
156				156				m_{22}
157				157	0x80000000	AP	0x80000000	m_4
158				158	AP	AP		m_1
159				159	AP	AP		m_{25}
160				160	AP	AP		m_{15}

Table 7. Experimental results for the amplified probability

Direction	Number of trials	Number of obtained 4-sums	Amplified probability
Back	1,000,000	53,065	$2^{-4.24}$
For	1,000,000	37,623	$2^{-4.73}$
Total	1,000,000	1,975	$2^{-8.98}$

Table 8. An example of the boomerang quartet for the full 5-pass HAVAL

H_i^1	0x6ad6913b 0x52831497 0x42e2afea 0x042171e8 0x05c66540 0xf6308a5d 0x69b242bb 0xfeadf2df
M_i^1	0x55f408ea 0xade29473 0x5cd48f01 0x862fac29 0xb59b9103 0xdfe1dff3 0x44aaff68 0xa5716cc8
	0xd9b3c72a 0x9d9907bb 0x263e9a6f 0x0d81dbdd 0x1a1d1f69 0x35a88db0 0xb50f50b3 0xcb85d403
	0xe2898bd5 0x3dc4e64c 0x48a696ae 0x1568e06b 0x286a00c5 0x236529bd 0x8bb673fd 0x481411ed
	0xb2117cb1 0xe6911e8d 0x5816e997 0x1a8fc1d3 0xc5dda128 0x43e5f428 0xcf1e861f 0xf5258b98
H_{i+1}^1	0x50b484bf 0x9d28c720 0xc2a5ab4d 0x5aec2d4b 0x63659cae 0x0023f316 0xa02276be 0xeab5fb84
H_i^2	0x6ad6913b 0x52831497 0x42e2afea 0x042171e8 0x05c66540 0xf6308e5d 0x69b242bb 0xfeae32df
M_i^2	0x55f408ea 0xade29473 0xdcd48f01 0x862fac29 0xb59b9103 0xdfe1dff3 0x44aaff68 0xa5716cc8
	0xd9b3c72a 0x9d9907bb 0x263e9a6f 0x0d81dbdd 0x1a1d1f69 0x35a88db0 0xb50f50b3 0xcb85d403
	0xe2898bd5 0x3dc4e64c 0x48a696ae 0x1568e06b 0x286a00c5 0x236529bd 0x8bb673fd 0x481411ed
	0xb2117cb1 0xe6911e8d 0x5816e997 0x1a8fc1d3 0xc5dda128 0x43e5f428 0xcf1e861f 0xf5258b98
H_{i+1}^2	0xfa15769c 0x6ed1b19a 0x405b263b 0x57cd6359 0xd8688750 0xcdc3c9d3 0xa3dc7fd8 0x2e59f283
H_i^3	0xb70b5251 0x851d041a 0x7a5f5fad 0x98626bb1 0x9d739cbc 0x67bc3181 0xe48e4cac 0xeeb57f26
M_i^3	0x55f408ea 0xade29473 0x5cd48f01 0x862fac29 0x359b9103 0xdfe1dff3 0x44aaff68 0xa5716cc8
	0xd9b3c72a 0x9d9907bb 0x263e9a6f 0x0d81dbdd 0x1a1d1f69 0x35a88db0 0xb50f50b3 0xcb85d403
	0xe2898bd5 0x3dc4e64c 0x48a696ae 0x1568e06b 0x286a00c5 0x236529bd 0x8bb673fd 0x481411ed
	0xb2117cb1 0xe6911e8d 0x5816e997 0x1a8fc1d3 0xc5dda128 0x43e5f428 0xcf1e861f 0xf5258b98
H_{i+1}^3	0x9ce945d5 0xcfc2b6a3 0xfa225b10 0xef2d2714 0x7b12d42a 0x71af9a3a 0x1afe80af 0xdbbd87cb
H_i^4	0xb70b5251 0x851d041a 0x7a5f5fad 0x98626bb1 0x9d739cbc 0x67bc3581 0xe48e4cac 0xeeb5bf26
M_i^4	0x55f408ea 0xade29473 0xdcd48f01 0x862fac29 0x359b9103 0xdfe1dff3 0x44aaff68 0xa5716cc8
	0xd9b3c72a 0x9d9907bb 0x263e9a6f 0x0d81dbdd 0x1a1d1f69 0x35a88db0 0xb50f50b3 0xcb85d403
	0xe2898bd5 0x3dc4e64c 0x48a696ae 0x1568e06b 0x286a00c5 0x236529bd 0x8bb673fd 0x481411ed
	0xb2117cb1 0xe6911e8d 0x5816e997 0x1a8fc1d3 0xc5dda128 0x43e5f428 0xcf1e861f 0xf5258b98
H_{i+1}^4	0x464a37b2 0xa16ba11d 0x77d7d5fe 0xec0e5d22 0xf015becc 0x3f4f70f7 0x1eb889c9 0x1f617eca
4-sum	0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

Table 9. An example of the boomerang quartet for MD5

H_i^1	0x7ad51bee 0x68a07529 0x5369e5f1 0x62f52251
M_i^1	0x58df0f5e 0x678b3525 0x03105c08 0xa068f82a 0x21ead339 0xe6e2ea9c 0x5cf986e1 0x9890fd27
	0xcf8a438f 0x2cecb915 0x44935dfe 0xf06f103f 0x72d5b376 0x9688dfed 0x7b2ae2f6 0xe9256628
H_{i+1}^1	0x1de7b79a 0x6e573e2a 0x0ef900e3 0xc72985ef
H_i^2	0xfad51bee 0x68a07529 0xd369e5f1 0xe2f52251
M_i^2	0x58df0f5e 0x678b3525 0x83105c08 0xa068f82a 0x21ead339 0xe6e2ea9c 0x5cf986e1 0x9890fd27
	0xcf8a438f 0x2cecb915 0x44935dfe 0xf06f103f 0x72d5b376 0x9688dfed 0x7b2ae2f6 0xe9256628
H_{i+1}^2	0x03d5ae50 0x722a5685 0x361b13a1 0x75a3a89d
H_i^3	0x97e364fe 0xb191e24c 0xdec0361f 0x6a8d3d9f
M_i^3	0x58df0f5e 0x678b3525 0x03105c08 0xa068f82a 0x21ead339 0xe6e2ea9c 0x5cf986e1 0x9890fd27
	0xcf8a438f 0x2cecb915 0x44935dfe 0x706f103f 0x72d5b376 0x9688dfed 0x7b2ae2f6 0xe9256628
H_{i+1}^3	0x3af600aa 0xb748a94d 0x9a4f4f11 0xcec19f3d
H_i^4	0x17e364fe 0xb191e24c 0x5ec0361f 0xea8d3d9f
M_i^4	0x58df0f5e 0x678b3525 0x83105c08 0xa068f82a 0x21ead339 0xe6e2ea9c 0x5cf986e1 0x9890fd27
	0xcf8a438f 0x2cecb915 0x44935dfe 0x706f103f 0x72d5b376 0x9688dfed 0x7b2ae2f6 0xe9256628
H_{i+1}^4	0x20e3f760 0xbb1bc1a8 0xc17161cf 0x7d3bc1eb
4-sum	0x00000000 0x00000000 0x00000000 0x00000000

Table 10. An example of the boomerang quartet for 3-pass HAVAL

H_i^1	0x8af103dd 0x89952e4e 0xba8ba930 0xb1681125 0x8bf68d12 0x11f454da 0x31babeaf 0x1c684f37
M_i^1	0x14c97b03 0x03021d0b 0x6e0a398b 0x12acd59d 0xa0e58017 0x56a25710 0x31381427 0x193906fa
	0xa97fe484 0x9228f3e7 0x3d307061 0x7ea148a1 0xcf1cf1f5 0x2b250fb8 0xd874f573 0xb71f7585
	0xb277563c 0xdb382652 0x1068c5fc 0x12cd8ceb 0x290580bf 0xc95cca2a 0x931d8c52 0xc835f9e8
	0x3ef066f9 0x098a53d0 0xf25db814 0xdb003165 0x31779903 0x4ebc57a0 0x9060622a 0x24c0bf29
H_{i+1}^1	0x9b18b769 0x01959420 0x480cea32 0x7c38cf17 0x70323bda 0xd46e06e9 0x09d05ae3 0xd315f8f6
H_i^2	0x8af103dd 0x89952e4e 0xba8ba930 0xb1681125 0x8bf68d12 0x11f454da 0x31babeaf 0x1c684b37
M_i^2	0x94c97b03 0x03021d0b 0x6e0a398b 0x12acd59d 0xa0e58017 0x56a25710 0x31381427 0x193906fa
	0xa97fe484 0x9228f3e7 0x3d307061 0x7ea148a1 0xcf1cf1f5 0x2b250fb8 0xd874f573 0xb71f7585
	0xb277563c 0xdb382652 0x1068c5fc 0x12cd8ceb 0x290580bf 0xc95cca2a 0x931d8c52 0xc835f9e8
	0x3ef066f9 0x098a53d0 0xf25db814 0xdb003165 0x31779903 0x4ebc57a0 0x9060622a 0x24c0bf29
H_{i+1}^2	0x2ab0c721 0xda378441 0x99789481 0xaf2db9cb 0x900971d1 0xdfa8ec61 0x122c330e 0xa77c26af
H_i^3	0x6ca48418 0x6711d760 0x52670414 0x4dfe762f 0xf9bae1d5 0x7a9a2074 0x4518e6bf 0x6acec54d
M_i^3	0x14c97b03 0x03021d0b 0x6e0a398b 0x12acd59d 0xa0e58017 0xd6a25710 0x31381427 0x193906fa
	0xa97fe484 0x9228f3e7 0x3d307061 0x7ea148a1 0xcf1cf1f5 0x2b250fb8 0xd874f573 0xb71f7585
	0xb277563c 0xdb382652 0x1068c5fc 0x12cd8ceb 0x290580bf 0xc95cca2a 0x931d8c52 0xc835f9e8
	0x3ef066f9 0x098a53d0 0xf25db814 0xdb003165 0x31779903 0x4ebc57a0 0x9060622a 0x24c0bf29
H_{i+1}^3	0x7ccc37a4 0xdf123d32 0xdfe84516 0x18cf3421 0xddf6909d 0x3d13d283 0x9d2e82f3 0x217c6f0c
H_i^4	0x6ca48418 0x6711d760 0x52670414 0x4dfe762f 0xf9bae1d5 0x7a9a2074 0x4518e6bf 0x6acec14d
M_i^4	0x94c97b03 0x03021d0b 0x6e0a398b 0x12acd59d 0xa0e58017 0xd6a25710 0x31381427 0x193906fa
	0xa97fe484 0x9228f3e7 0x3d307061 0x7ea148a1 0xcf1cf1f5 0x2b250fb8 0xd874f573 0xb71f7585
	0xb277563c 0xdb382652 0x1068c5fc 0x12cd8ceb 0x290580bf 0xc95cca2a 0x931d8c52 0xc835f9e8
	0x3ef066f9 0x098a53d0 0xf25db814 0xdb003165 0x31779903 0x4ebc57a0 0x9060622a 0x24c0bf29
H_{i+1}^4	0x0c64475c 0xb7b42d53 0x3153ef65 0x4bc41ed5 0xfdcdc694 0x484eb7fb 0xa58a5b1e 0xf5e29cc5
4-sum	0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

Table 11. An example of the boomerang quartet for 4-pass HAVAL

H_i^1	0x564187b3 0x5af775fb 0x10136ca0 0x8a9ffa0c 0x3edeecd0 0xd2e74f6c 0x15576f1c 0x70de0eb7
M_i^1	0x8bc2d93c 0xc9e9f4eb 0xd4e85905 0x39828bfb 0x6aefef32 0xc3284793 0x0ed3477d 0x7e8a91f9
	0x84963793 0x263a6675 0xafa8892c 0x340904ff 0xaccb5103 0xf3bac932 0xbe1f0ae4 0x93c377c1
	0x48142ead 0x5b911f1b 0xe5693f5f 0xd1c28e92 0x11b24646 0xac7dd73d 0x745b0c46 0x5ca1756c
	0xdfb80fbd 0x88cee1fd 0x7bd6c417 0x43ab29df 0xfdb5d87a 0x3569ce43 0xc7dc1347 0x462ef5da
H_{i+1}^1	0x910e3d63 0x83c406ec 0x464230f7 0x3bfc4d84 0x14fddff2 0x5092bf5f 0x07cd2ad3 0x31c0e36a
H_i^2	0x564187b3 0x5af775fb 0x10136ca0 0x8a9ffa0c 0x3edeecd0 0xd2e74b6c 0x15572f1c 0x70e24eb7
M_i^2	0x8bc2d93c 0xc9e9f4eb 0x54e85905 0x39828bfb 0x6aefef32 0xc3284793 0x0ed3477d 0x7e8a91f9
	0x84963793 0x263a6675 0xafa8892c 0x340904ff 0xaccb5103 0xf3bac932 0xbe1f0ae4 0x93c377c1
	0x48142ead 0x5b911f1b 0xe5693f5f 0xd1c28e92 0x11b24646 0xac7dd73d 0x745b0c46 0x5ca1756c
	0xdfb80fbd 0x88cee1fd 0x7bd6c417 0x43ab29df 0xfdb5d87a 0x3569ce43 0xc7dc1347 0x462ef5da
H_{i+1}^2	0xe6b6ce20 0x3260de7a 0x681aa45d 0x277995cd 0x9d4959bc 0xaec251c2 0x41446efa 0x75cf2b80
H_i^3	0xaf8abf6d 0xd2aafd4c 0xc7f01506 0xfd258be0 0x299edd95 0xa561cfbc 0x61175f52 0xec8049a0
M_i^3	0x8bc2d93c 0xc9e9f4eb 0xd4e85905 0x39828bfb 0x6aefef32 0xc3284793 0x0ed3477d 0x7e8a91f9
	0x84963793 0x263a6675 0xafa8892c 0x340904ff 0xaccb5103 0xf3bac932 0xbe1f0ae4 0x93c377c1
	0x48142ead 0xdb911f1b 0xe5693f5f 0xd1c28e92 0x11b24646 0xac7dd73d 0x745b0c46 0x5ca1756c
	0xdfb80fbd 0x88cee1fd 0x7bd6c417 0x43ab29df 0xfdb5d87a 0x3569ce43 0xc7dc1347 0x462ef5da
H_{i+1}^3	0xea57751d 0xfb778e3d 0xfe1ed95d 0xae81df58 0x7fbdd0b7 0x230d3faf 0x538d1b09 0xad631e53
H_i^4	0xaf8abf6d 0xd2aafd4c 0xc7f01506 0xfd258be0 0x299edd95 0xa561cbbc 0x61171f52 0xec8489a0
M_i^4	0x8bc2d93c 0xc9e9f4eb 0x54e85905 0x39828bfb 0x6aefef32 0xc3284793 0x0ed3477d 0x7e8a91f9
	0x84963793 0x263a6675 0xafa8892c 0x340904ff 0xaccb5103 0xf3bac932 0xbe1f0ae4 0x93c377c1
	0x48142ead 0xdb911f1b 0xe5693f5f 0xd1c28e92 0x11b24646 0xac7dd73d 0x745b0c46 0x5ca1756c
	0xdfb80fbd 0x88cee1fd 0x7bd6c417 0x43ab29df 0xfdb5d87a 0x3569ce43 0xc7dc1347 0x462ef5da
H_{i+1}^4	0x400005da 0xaa1465cb 0x1ff74cc3 0x99ff27a1 0x08094a81 0x813cd212 0x8d045f30 0xf1716669
4-sum	0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

5.4 Experimental Results

By following the algorithm in Alg. 3, we evaluated the amplified probability for the first and last several rounds. The results are shown in Table 7. According to our experiments, $AP^{Back} = 2^{-4.24}$, $AP^{For} = 2^{-4.73}$, and the entire success probability is $2^{-8.98}$, which matches $AP^{Back} \times AP^{For} = 2^{-8.97}$. The attack complexity is for $2^{8.98}$ iterations of Steps 4–11 in Alg. 4. Because we compute quartets, the complexity is approximately $2^{11} (\approx 4 \times 2^{8.98})$ compression function computations. Finally, we implemented our 4-sum distinguisher on 5-pass HAVAL. An example of the generated 4-sum quartet is presented in Table 8.

6 Concluding Remarks

We studied the boomerang attack approach on hash functions. We proved that the previous differential path on 5-pass HAVAL contained a flaw. We then constructed the new path and proposed the 4-sum distinguisher on the compression function with a complexity of approximately 2^{11} computations. We implemented the attack and showed an example of the 4-sum quartet. As far as we know, this is the first feasible result on the full compression function of 5-pass HAVAL.

References

1. Aoki, K., Sasaki, Y.: Preimage Attacks on One-Block MD4, 63-Step MD5 and More. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 103–119. Springer, Heidelberg (2009)
2. Aumasson, J.-P., Meier, W., Mendel, F.: Preimage Attacks on 3-Pass HAVAL and Step-Reduced MD5. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 120–135. Springer, Heidelberg (2009)
3. Biryukov, A., Nikolić, I., Roy, A.: Boomerang Attacks on BLAKE-32. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 218–237. Springer, Heidelberg (2011)
4. den Boer, B., Bosselaers, A.: Collisions for the Compression Function of MD-5. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 293–304. Springer, Heidelberg (1994)
5. Dobbertin, H.: The Status of MD5 after a Recent Attack. CryptoBytes The technical newsletter of RSA Laboratories, a division of RSA Data Security, Inc. 2(2) (Summer 1996)
6. Guo, J., Ling, S., Rechberger, C., Wang, H.: Advanced Meet-in-the-Middle Preimage Attacks: First Results on Full Tiger, and Improved Results on MD4 and SHA-2. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 56–75. Springer, Heidelberg (2010)
7. Joux, A., Peyrin, T.: Hash Functions and the (Amplified) Boomerang Attack. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 244–263. Springer, Heidelberg (2007)
8. Kim, J.-S., Biryukov, A., Preneel, B., Lee, S.-J.: On the Security of Encryption Modes of MD4, MD5 and HAVAL. In: Qing, S., Mao, W., López, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 147–158. Springer, Heidelberg (2005)

9. Kim, J., Biryukov, A., Preneel, B., Lee, S.: On the Security of Encryption Modes of MD4, MD5 and HAVAL. Cryptology ePrint Archive, Report 2005/327 (2005); In: Qing, S., Mao, W., López, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 147–158. Springer, Heidelberg (2005)
10. Lamberger, M., Mendel, F.: Higher-Order Differential Attack on Reduced SHA-256. Cryptology ePrint Archive, Report 2011/037 (2011), http://eprint.iacr.org/2011/037
11. Leurent, G.: MD4 is Not One-Way. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 412–428. Springer, Heidelberg (2008)
12. Murphy, S.: The Return of the Cryptographic Boomerang. IEEE Transactions on Information Theory 57(4), 2517–2521 (2011)
13. Rivest, R.L.: The MD4 Message Digest Algorithm. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 303–311. Springer, Heidelberg (1991), also appeared in RFC 1320 http://www.ietf.org/rfc/rfc1320.txt
14. Rivest, R.L.: Request for Comments 1321: The MD5 Message Digest Algorithm. The Internet Engineering Task Force (1992)
15. Sakai, Y., Sasaki, Y., Wang, L., Ohta, K., Sakiyama, K.: Preimage Attacks on 5- Pass HAVAL Reduced to 158-Steps and One-Block 3-Pass HAVAL. Industrial Track of ACNS 2011 (2011)
16. Sasaki, Y., Aoki, K.: Preimage Attacks on 3, 4, and 5-Pass HAVAL. In: Pieprzyk, J.P. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 253–271. Springer, Heidelberg (2008)
17. Sasaki, Y., Aoki, K.: Finding Preimages in Full MD5 Faster than Exhaustive Search. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 134–152. Springer, Heidelberg (2009)
18. Sasaki, Y., Wang, L., Ohta, K., Kunihiro, N.: New Message Difference for MD4. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 329–348. Springer, Heidelberg (2007)
19. Suzuki, K., Kurosawa, K.: How to Find Many Collisions of 3-Pass HAVAL. In: Miyaji, A., Kikuchi, H., Rannenberg, K. (eds.) IWSEC 2007. LNCS, vol. 4752, pp. 428–443. Springer, Heidelberg (2007)
20. U.S. Department of Commerce, National Institute of Standards and Technology: Federal Register Vol. 72, No. 212/Friday, November 2, 2007/Notices (2007)
21. Van Rompay, B., Biryukov, A., Preneel, B., Vandewalle, J.: Cryptanalysis of 3-Pass HAVAL. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 228–245. Springer, Heidelberg (2003)
22. Wagner, D.: The Boomerang Attack. In: Knudsen, L.R. (ed.) FSE 1999. LNCS, vol. 1636, pp. 156–170. Springer, Heidelberg (1999)
23. Wagner, D.: A Generalized Birthday Problem. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 288–303. Springer, Heidelberg (2002)
24. Wang, X., Feng, D., Yu, X.: An Attack on Hash Function HAVAL-128. Science in China (Information Sciences) 48(5), 545–556 (2005)
25. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the Hash Functions MD4 and RIPEMD. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 1–18. Springer, Heidelberg (2005)
26. Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
27. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)

28. Wang, Z., Zhang, H., Qin, Z., Meng, Q.: Cryptanalysis of 4-Pass HAVAL. Crptology ePrint Archive, Report 2006/161 (2006)
29. Xie, T., Liu, F., Feng, D.: Could the 1-MSB Input Difference be the Fastest Collision Attack for MD5? Cryptology ePrint Archive, Report 2008/391 (2008)
30. Yoshida, H., Biryukov, A., De Cannière, C., Lano, J., Preneel, B.: Non-Randomness of the Full 4 and 5-Pass HAVAL. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 324–336. Springer, Heidelberg (2005)
31. Yu, H., Wang, X., Yun, A., Park, S.: Cryptanalysis of the Full HAVAL with 4 and 5 Passes. In: Robshaw, M.J.B. (ed.) FSE 2006. LNCS, vol. 4047, pp. 89–110. Springer, Heidelberg (2006)
32. Zheng, Y., Pieprzyk, J., Seberry, J.: HAVAL — One-Way Hashing Algorithm with Variable Length of Output. In: Zheng, Y., Seberry, J. (eds.) AUSCRYPT 1992. LNCS, vol. 718, pp. 83–104. Springer, Heidelberg (1993)

A Differential Path Search Algorithm for 5-Pass HAVAL

Our path search algorithm is semi-automated and minimizes the Hamming distance of the entire inside path. We independently searched for the path for E_1 (steps 60 – 79) with $\Delta m_2 =$ `0x80000000` and path for E_2 (steps 98 – 73) with $\nabla m_4 =$ `0x80000000`. Conditions and contradiction of two paths were later checked by hand. We only explain the algorithm for E_1 in Alg. 5. $HW(\cdot)$ returns the Hamming weight of the input variable.

After an enough number of iterations of Alg. 5, we obtained the path in Table 6 whose $tempHD$ is 6.

B Examples of Boomerang Quartet

The differential paths in [9] can be used to construct a 4-sum on the compression function. We show the generated 4-sums for MD5, 3-pass HAVAL, and 4-pass HAVAL. The amplified probability to satisfy the entire path is approximately 2^{-8} for MD5, 2^{-2} for 3-pass HAVAL, and 2^{-9} for 4-pass HAVAL.