Second Preimage Attack on 3-Pass HAVAL and Partial Key-Recovery Attacks on HMAC/NMAC-3-Pass HAVAL

Eunjin Lee[1], Donghoon Chang[1], Jongsung Kim[1], Jaechul Sung[2], Seokhie Hong[1]

[1] Center for Information Security Technologies(CIST),
Korea University, Seoul, Korea
{walgadak,pointchang,joshep,hsh }@cist.korea.ac.kr
[2] University of Seoul,Seoul, Korea
jcsung@uos.ac.kr

Abstract. In 1992, Zheng, Pieprzyk and Seberry proposed a one-way hashing algorithm called HAVAL, which compresses a message of arbitrary length into a digest of 128, 160, 192, 224 or 256 bits. It operates in so called passes where each pass contains 32 steps. The number of passes can be chosen equal to 3, 4 or 5. In this paper, we devise a new differential path of 3-pass HAVAL with probability 2^{-114}, which allows us to design a second preimage attack on 3-pass HAVAL and partial key recovery attacks on HMAC/NMAC-3-pass HAVAL. Our partial key-recovery attack works with 2^{122} oracle queries, $5 \cdot 2^{32}$ memory bytes and 2^{96} 3-pass HAVAL computations.

Keywords : HAVAL, NMAC, HMAC, Second preimage attack, Key recovery attack.

1 Introduction

In 2004 and 2005, Biham *et al.* and Wang *et al.* published several important cryptanalytic articles [1, 2, 12–15] that demonstrate efficient collision search algorithms for the MD4-family of hash functions. Their proposed neutral-bit and message modification techniques make it possible to significantly improve previous known collision attacks on MD4, MD5, HAVAL, RIPEMD, SHA-0 and SHA-1 [3, 9, 10, 17], including the second preimage attack on MD4 which finds a second preimage for a random message with probability 2^{-56} [18].

There have also been several articles that present attacks on NMAC and HMAC based on the MD4 family. In 2006, Kim *et al.* first proposed distinguishing and forgery attacks on NMAC and HMAC based on the full or reduced HAVAL, MD4, MD5, SHA-0 and SHA-1 [7] and Contini and Yin presented forgery and partial key recovery attacks on HMAC/NMAC-MD4, -SHA-0, -reduced 34-round SHA-1 and NMAC-MD5 [4]. More recently, full key-recovery attacks on HMAC/NMAC-MD4, reduced 61-round SHA-1 and NMAC-MD5 were proposed in FC 2007 [8] and in CRYPTO 2007 [6].

The motivation of this paper is that 1) there are strong collision producing differentials of HAVAL for collision attacks [10, 11], but no differential of HAVAL has been proposed for second preimage attacks, and 2) there are distinguishing/forgery attacks on HMAC/NMAC-HAVAL [7], but no key-recovery attack has been proposed. This paper investigates if 3-pass HAVAL and HMAC/NMAC-3-pass HAVAL are vulnerable to the second preimage and partial key recovery attacks, respectively. (After our submission, we learned that Hongbo Yu worked independently for her doctoral dissertation [16] on partial key recovery attacks on HAVAL-based HMAC and second preimage attack on HAVAL).

The cryptographic hash function HAVAL was proposed by Y. Zheng et al. in 1992 [19]. It takes an input value of arbitrary length and digests it into variant lengths of 128, 160, 192, 224 or 256 bits. In this paper, we present a new second preimage differential path of 3-pass HAVAL with probability 2^{-114} and devise a second preimage attack on 3-pass HAVAL, and a partial key recovery attack on HMAC/NMAC-3-pass HAVAL with 2^{122} oracle queries, $5 \cdot 2^{32}$ memory bytes and 2^{96} 3-pass HAVAL computations.

This paper is organized as follows. In Section 2, we describe HAVAL, HMAC, NMAC, and notations. Next, we present a second preimage attack on 3-pass HAVAL in Section 3 and apply it to recover a partial key of HMAC/NMAC-3-pass HAVAL in Section 4. Finally, we conclude in Section 5.

2 Preliminaries

In this section, we give a brief description of the HAVAL hash function, the HMAC/NMAC algorithms and notations used in the paper.

2.1 Description of HAVAL

HAVAL produces hashes in different lengths of 128, 160, 192, 224 and 256 bits. It allows that users can choose the number of passes 3, 4 or 5, where each pass contains 32 steps. It computes the hashes in the following procedure:

- Padding: an inserted message is padded into a multiple of 1024 bits.
- Compression function H: let $M^0, M^1, \cdots, M^S$ be 1024-bit message blocks and each M^i consists of 32 32-bit words, that is, $M^i = M_0^i || M_1^i || \cdots || M_{31}^i$, where M_j^i is a 32-bit word.
  - $h_0 = H(IV, M^0)$, where IV is the initial value.
  - $h_1 = H(h_0, M^1), \cdots, h_s = H(h_{s-1}, M^S)$
- Output of HAVAL: H_n

The HAVAL compression function H processes 3, 4 or 5 passes. Let F_1, F_2, F_3, F_4 and F_5 be the five passes and (D_{in}, M) be the input value of H, where D_{in} is a 256-bit initial block and M is a 1024-bit message block. Then the output of the compression function D_{out} can be computed in the following way.

$$E_0 = D_{in},\ E_1 = F_1(E_0, M),\ E_2 = F_2(E_1, M),\ E_3 = F_3(E_2, M);$$
$$E_4 = F_4(E_3, M)\ \ (pass = 4, 5),\ E_5 = F_5(E_4, M)\ \ (pass = 5);$$

$$D_{out} = \begin{cases} E_3 \boxplus E_0, & pass = 3 \\ E_4 \boxplus E_0, & pass = 4 \\ E_5 \boxplus E_0, & pass = 5 \end{cases}$$

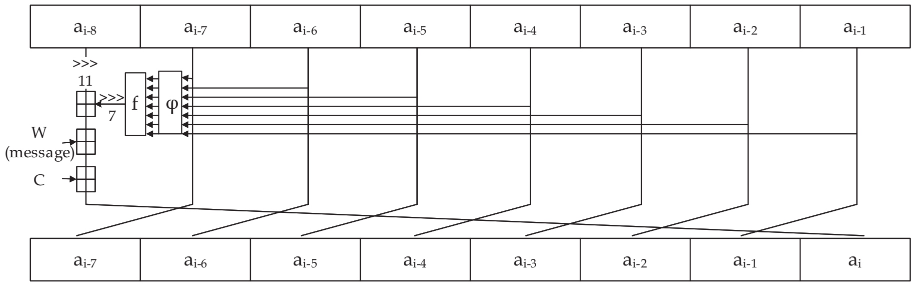

Fig. 1. i-th step of HAVAL hash function

Fig. 1 shows the i-th step of HAVAL, where a_i represents the updated 32-bit value of the i-th step. Let a 1024-bit message block M be denoted $M = M_0||M_1||\cdots||M_{30}||M_{31}$, where M_i $(i = 0, 1, \cdots, 31)$ is a 32-bit word, then the orders of the message words in each pass are as in Table 1.

Each pass employs a different Boolean function f_i $(i = 1, 2, 3, 4, 5)$ and a different permutation function. The following f_i is used in pass i:

$$\begin{aligned}
f_1(x_6, x_5, x_4, x_3, x_2, x_1, x_0) &= x_1x_4 \oplus x_2x_5 \oplus x_3x_6 \oplus x_0x_1 \oplus x_0 \\
f_2(x_6, x_5, x_4, x_3, x_2, x_1, x_0) &= x_1x_2x_3 \oplus x_2x_4x_5 \oplus x_1x_2 \oplus x_1x_4 \oplus \\
&\quad x_2x_6 \oplus x_3x_5 \oplus x_4x_5 \oplus x_0x_2 \oplus x_0 \\
f_3(x_6, x_5, x_4, x_3, x_2, x_1, x_0) &= x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_5 \oplus x_3x_6 \oplus x_0x_3 \oplus x_0 \\
f_4(x_6, x_5, x_4, x_3, x_2, x_1, x_0) &= x_1x_2x_3 \oplus x_2x_4x_5 \oplus x_3x_4x_6 \oplus x_1x_4 \oplus x_2x_6 \oplus \\
&\quad x_3x_4 \oplus x_3x_5 \oplus x_3x_6 \oplus x_4x_5 \oplus x_4x_6 \oplus x_0x_4 \oplus x_0 \\
f_5(x_6, x_5, x_4, x_3, x_2, x_1, x_0) &= x_1x_4 \oplus x_2x_5 \oplus x_3x_6 \oplus x_0x_1x_2x_3 \oplus x_0x_5 \oplus x_0
\end{aligned}$$

Let $\varphi_{i,j}$ be the permutation function of the j-th pass of the i-pass HAVAL. Table 2 shows the $\varphi_{i,j}$ used in each pass. In each step, the updated value a_i is

Table 1. Orders of message words

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

computed as

$$a_i = (a_{i-8} \ggg 11) \boxplus (f(\varphi(a_{i-7}, a_{i-6}, \cdots, a_{i-1})) \ggg 7) \boxplus M_i \boxplus C,$$

where $X \ggg i$ is the right cyclic rotation of X by i bits, and C is a constant.

Table 2. $\varphi_{i,j}$ used in each pass

permutations	x_6 x_5 x_4 x_3 x_2 x_1 x_0
$\varphi_{3,1}$	x_1 x_0 x_3 x_5 x_6 x_2 x_4
$\varphi_{3,2}$	x_4 x_2 x_1 x_0 x_5 x_3 x_6
$\varphi_{3,3}$	x_6 x_1 x_2 x_3 x_4 x_5 x_0
$\varphi_{4,1}$	x_2 x_6 x_1 x_4 x_5 x_3 x_0
$\varphi_{4,2}$	x_3 x_5 x_2 x_0 x_1 x_6 x_4
$\varphi_{4,3}$	x_1 x_4 x_3 x_6 x_0 x_2 x_5
$\varphi_{4,4}$	x_6 x_4 x_0 x_5 x_2 x_1 x_3
$\varphi_{5,1}$	x_3 x_4 x_1 x_0 x_5 x_2 x_6
$\varphi_{5,2}$	x_6 x_2 x_1 x_0 x_3 x_4 x_5
$\varphi_{5,3}$	x_2 x_6 x_0 x_4 x_3 x_1 x_5
$\varphi_{5,4}$	x_1 x_5 x_3 x_2 x_0 x_4 x_6
$\varphi_{5,5}$	x_2 x_5 x_0 x_6 x_4 x_3 x_1

2.2 Description of HMAC/NMAC

Fig. 2 shows NMAC and HMAC based on a compression function f which maps $\{0,1\}^n \times \{0,1\}^b$ to $\{0,1\}^n$. The K_1 and K_2 are all n-bit keys and the $\overline{K} = K||0^{b-n}$, where K is an n-bit key. The opad is formed by repeating the byte

'0x36' as many times as needed to get a b-bit block, and the ipad is defined similarly using the byte '0x5c'.

Let $F : \{IV\} \times (\{0,1\}^b)^* \rightarrow \{0,1\}^n$ be the iterated hash function defined as $F(IV, M^1||M^2||\cdots||M^S) = f(\cdots f(f(IV, M^1), M^2)\cdots, M^S)$, where M^i is a b bit message. Let g be a padding method, $g(x) = x||10^t||\mathsf{bin}_{64}(x)$, where t is the smallest non-negative integer such that $g(x)$ is a multiple of b and $\mathsf{bin}_i(x)$ is the i-bit binary representation of x. Then, NMAC and HMAC are defined as follows:

$$\mathrm{NMAC}_{K_1,K_2}(M) = H(K_2, g(H(K_1, g(M))))$$
$$\mathrm{HMAC}_K(M) = H(IV, g(\overline{K} \oplus \mathsf{opad}||H(IV, g(\overline{K} \oplus \mathsf{ipad}||M)))).$$

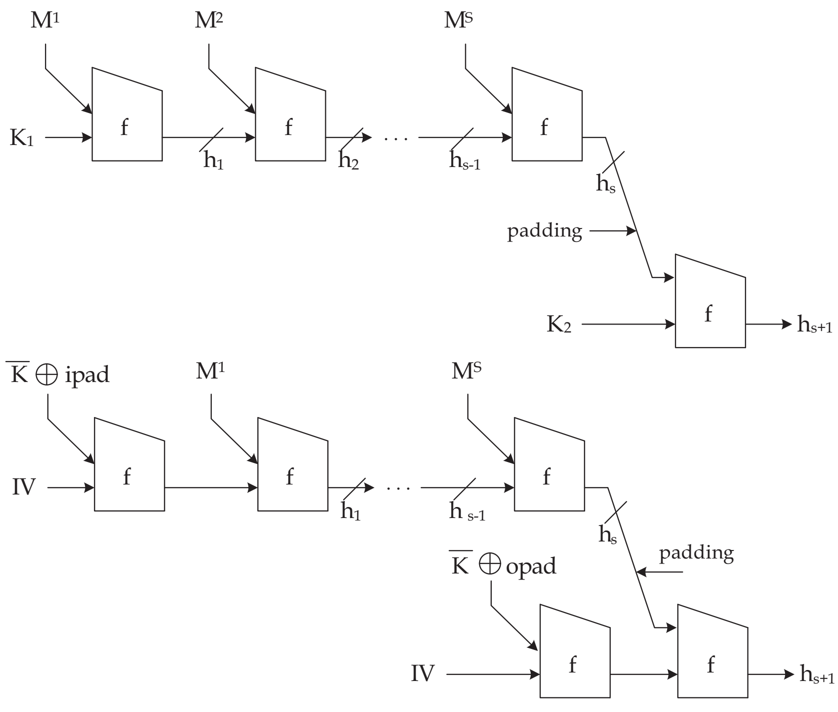

Fig. 2. NMAC and HMAC

2.3 Notations

Let M and M' be 1024-bit messages such that $M = M_0||M_1||\cdots||M_{31}$ and $M' = M'_0||M'_1||\cdots||M'_{31}$, where M_i $(i = 0, 1, 2, \cdots, 31)$ and M'_j $(j = 0, 1, 2, \cdots, 31)$ are

32-bit words. We denote by a_i (resp., a'_i) the updated value of the i-th step using the message M (resp., M'). Let t_i (resp., t'_i) be the output value of the Boolean function of the i-th step using the message M (resp., M'). The j-th bits of a_i and t_i are denoted $a_{i,j}$ and $t_{i,j}$. Additionally, we use several following notations in our attacks, where $0 \leq j \leq 31$.

- $a_i[j]$: $a_{i,j} = 0, a'_{i,j} = 1$,
- $a_i[-j]$: $a_{i,j} = 1, a'_{i,j} = 0$,
- $t_i[j]$: $t_{i,j} = 0, t'_{i,j} = 1$,
- $t_i[-j]$: $t_{i,j} = 1, t'_{i,j} = 0$.

3 Second Preimage Attack on 3-Pass HAVAL

In this section, we show how to construct a second preimage differential path of 3-pass HAVAL. Using this differential path, we find a second preimage of 3-pass HAVAL with probability 2^{-114}, i.e., for a given message M, we find another message M' with probability 2^{-114} satisfying $H(M) = H(M')$, where H is 3-pass HAVAL. Our differential path of 3-pass HAVAL is stronger than the previous ones [7, 9, 11, 12] against the second preimage attack.

3.1 Second Preimage Differential Path of 3-Pass HAVAL

Let two 1024-bit message blocks $M = M_0 || M_1||M_2|| \cdots ||M_{31}$ and $M' = M'_0||M'_1|| M'_2|| \cdots ||M'_{31}$ satisfy $M_i = M'_i$ for $i = 0, 1, \cdots , 21, 23, 24, \cdots , 31$ and $M_{22} \oplus M'_{22} = 2^{31}$. Then we can use these two messages to construct a second preimage differential path of 3-pass HAVAL with probability 2^{-114}. Table 3 shows our second preimage differential path of 3-pass HAVAL, which has been constructed as follows.

First of all, from the message pair we get the input difference to the 23-rd step $(\Delta a_{15}, \Delta a_{16}, \Delta a_{17}, \Delta a_{18}, \Delta a_{19}, \Delta a_{20}, \Delta a_{21}, \Delta a_{22}) = (0, 0, 0, 0, 0, 0, 0, a_{22}[31])$ if a condition $a_{22,31} = 0$ holds. Recall that $(a_{i-8}, a_{i-7}, \cdots , a_{i-2}, a_{i-1})$ is the input state to the i-th step. We assume that the output differences of the Boolean functions from the 23-rd step to the 36-th step are all zeroes. Then we can obtain the input difference to the 37-th step is $(0, a_{30}[20], 0, 0, 0, 0, 0, 0)$. It is easy to see that the required assumption works if several conditions hold in our differential, which we call sufficient conditions. For example, consider a difference Δt_{24}. The input difference to the 24-th step is $(\Delta a_{16}, \Delta a_{17}, \Delta a_{18}, \Delta a_{19}, \Delta a_{20}, \Delta a_{21}, \Delta a_{22}, \Delta a_{23}) = (0, 0, 0, 0, 0, 0, a_{22}[31], 0)$. The permutation is $\varphi(x_6, x_5, x_4, x_3, x_2, x_1, x_0) = (x_1, x_0, x_3, x_5, x_6, x_2, x_4)$ and the Boolean function is $f(x_6, x_5, x_4, x_3, x_2, x_1, x_0) = x_1x_4 \oplus x_2x_5 \oplus x_3x_6 \oplus x_0x_1 \oplus x_0$ in the 24-th step. Thus, $f(\varphi(x_6, x_5, x_4, x_3, x_2, x_1, x_0)) = x_2x_3 \oplus x_6x_0 \oplus x_5x_1 \oplus x_4x_2 \oplus x_4$ and the most significant bit of the output of the Boolean function in the 24-th step is $a_{20,31}a_{21,31} \oplus a_{17,31}a_{23,31} \oplus a_{18,31}a_{22,31} \oplus a_{19,31}a_{21,31} \oplus a_{19,31}$. If $a_{18,31} = 0$, then the difference of $a_{22,31}$ does not have effect on the output difference of the Boolean function and thus $\Delta t_{24} = 0$. Thus, $a_{18,31} = 0$ is one of the sufficient conditions. We show

Table 3. Second preimage differential path of 3-pass HAVAL

step	ΔM_i	Δt_i	Δa_{i-8}	Δa_{i-7}	Δa_{i-6}	Δa_{i-5}	Δa_{i-4}	Δa_{i-3}	Δa_{i-2}	Δa_{i-1}
0	0	0	0	0	0	0	0	0	0	0
...	...	...		...					...	
21	0	0	0	0	0	0	0	0	0	0
22	±31	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	31
24	0	0	0	0	0	0	0	0	31	0
25	0	0	0	0	0	0	0	31	0	0
26	0	0	0	0	0	0	31	0	0	0
27	0	0	0	0	0	31	0	0	0	0
28	0	0	0	0	31	0	0	0	0	0
29	0	0	0	31	0	0	0	0	0	0
30	0	0	31	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	20
32	0	0	0	0	0	0	0	0	20	0
33	0	0	0	0	0	0	0	20	0	0
34	0	0	0	0	0	0	20	0	0	0
35	0	0	0	0	0	20	0	0	0	0
36	0	0	0	0	20	0	0	0	0	0
37	0	20	0	20	0	0	0	0	0	0
38	0	0	20	0	0	0	0	0	0	13
39	0	0	0	0	0	0	0	0	13	9
40	0	0	0	0	0	0	0	13	9	0
41	0	0	0	0	0	0	13	9	0	0
42	0	13	0	0	0	13	9	0	0	0
43	±31	±6	0	0	13	9	0	0	0	6
44	0	0	0	13	9	0	0	0	6	0
45	0	-9	13	9	0	0	0	6	0	0
46	0	0	9	0	0	0	6	0	0	0
47	0	0	0	0	0	6	0	0	0	30
48	0	0	0	0	6	0	0	0	30	0
49	0	0	0	6	0	0	0	30	0	0
50	0	0	6	0	0	0	30	0	0	0
51	0	0	0	0	0	30	0	0	0	-27,28
52	0	0	0	0	30	0	0	0	-27,28	0
53	0	30	0	30	0	0	0	-27,28	0	0
54	0	0	30	0	0	0	-27,28	0	0	23
55	0	28	0	0	0	-27,28	0	0	23	19
56	0	21	0	0	-27,28	0	0	23	19	21
57	0	0	0	-27,28	0	0	23	19	21	-14,15
58	0	-23	-27,28	0	0	23	19	21	-14,15	0
59	0	0	0	0	23	19	21	-14,15	0	0
60	0	0	0	23	19	21	-14,15	0	0	0
61	0	-19	23	19	21	-14,15	0	0	0	0
62	0	-15	19	21	-14,15	0	0	0	0	0
63	0	0	21	-14,15	0	0	0	0	0	0
64	0	-10	-14,15	0	0	0	0	0	0	10
65	0	0	0	0	0	0	0	0	10	0
66	0	0	0	0	0	0	0	10	0	0
67	0	0	0	0	0	0	10	0	0	0
68	0	0	0	0	0	10	0	0	0	0
69	0	0	0	0	10	0	0	0	0	0
70	0	0	0	10	0	0	0	0	0	0
71	±31	0	10	0	0	0	0	0	0	0
72	0	0	0	0	0	0	0	0	0	0
...	...	...		...					...	
95	0	0	0	0	0	0	0	0	0	0

in Table 5 of appendix all the sufficient conditions which satisfy our differential path .

In order to compute the probability that a message M satisfies the sufficient conditions listed in Table 5, we need to check the dependency of the conditions. To make the problem easier we first solve and simplify the conditions. In this process we may reduce the number of the sufficient conditions. Consider the conditions on the 20-th bit from the 31-st step to the 37-th step in Table 5.

1. 31-st step : $a_{30,20} = 0,\ a_{24,20} = 0$
2. 32-nd step : $a_{29,20}a_{26,20} \oplus a_{28,20} \oplus a_{29,20} = 0$
3. 33-rd step : $a_{31,20}a_{27,20} \oplus a_{32,20} \oplus a_{31,20} = 0$
4. 34-th step : $a_{33,20}a_{28,20} \oplus a_{28,20} \oplus a_{32,20} = 0$
5. 35-th step : $a_{29,20} = 0$
6. 36-th step : $a_{35,20}a_{32,20} \oplus a_{34,20}a_{33,20} \oplus a_{32,20} \oplus a_{31,20} \oplus a_{35,20} = 0$
7. 37-th step : $a_{31,20} = 0,\ a_{33,20}a_{35,20} \oplus a_{36,20}a_{34,20} \oplus a_{35,20}a_{34,20} = 0$

In the 32-nd step, we can simplify the condition to $a_{28,20} = 0$ by inserting the value $a_{29,20} = 0$ which is the condition in the 35-th step. Using this condition $a_{28,20} = 0$, we can obtain $a_{32,20} = 0$ in the 34-th step. This simplified condition $a_{32,20} = 0$ and the 37-th step condition $a_{31,20} = 0$ make the 33-rd step condition always hold. Moreover, the 36-th step condition is simplified to $a_{34,20}a_{33,20} \oplus a_{35,20} = 0$ due to the conditions $a_{31,20} = 0$ and $a_{32,20} = 0$. Following is the simplified conditions for steps 31-37 (note that the number of the sufficient conditions is reduced from 9 to 8 by solving the conditions):

1. 31-st step : $a_{30,20} = 0,\ a_{24,20} = 0$
2. 32-nd step : $a_{28,20} = 0$
3. 33-rd step : no condition
4. 34-th step : $a_{32,20} = 0$
5. 35-th step : $a_{29,20} = 0$
6. 36-th step : $a_{34,20}a_{33,20} \oplus a_{35,20} = 0$
7. 37-th step : $a_{31,20} = 0,\ a_{33,20}a_{35,20} \oplus a_{36,20}a_{34,20} \oplus a_{35,20}a_{34,20} = 0$

Table 6 in appendix collects all the simplified conditions for those of Table 5. We notice that the number of the sufficient conditions listed in Table 6 is 112, which seems to make the probability that a message satisfy all these conditions is 2^{-112}. However, it is not 2^{-112}, but approximately 2^{-114}. This is due to the fact that there are still dependencies in some conditions. For example, consider the conditions on the 13-th bit from the 38-th step to the 41-st step in Table 6.

1. 38-th step : $a_{38,13} = 1,\ a_{34,13}a_{32,13} \oplus a_{35,13} = 0$
2. 39-th step : $a_{33,13} \neq a_{35,13}$
3. 40-th step : $a_{34,13} \neq a_{39,13}$
4. 41-st step : $a_{40,13}a_{35,13} \oplus a_{35,13} \oplus a_{39,13} = 1$

These 5 conditions do not hold with probability 2^{-5}, but with probability $2^{-3} \cdot \frac{3}{16}$. The reason is as follows. The probability that the condition $a_{38,13} = 1$ is satisfied is 2^{-1}. Table 4 lists all the possible values of $a_{32,13}, a_{34,13}$ and $a_{35,13}$ which satisfy $a_{34,13}a_{32,13} \oplus a_{35,13} = 0$. The probability that this condition holds is $\frac{1}{2}(= \frac{4}{8})$ according to Table 4. In the 39-th step, the probability that $a_{33,13} \neq a_{35,13}$ is satisfied is 2^{-1} since $a_{33,13}$ is used only in the 39-th step. In the 40-th and 41-st steps, if $a_{35,13} = 0$, then $a_{39,13}$ and $a_{34,13}$ should be 0 and 1, respectively, and $a_{40,13}$ is either 0 or 1. The probability that $a_{35,13} = 0$ and $a_{34,13} = 1$ hold is $\frac{1}{4}$ (one out of four cases, see Table 4). Thus the probability that $a_{34,13} = 1, a_{35,13} = 0$, and $a_{39,13} = 0$ are satisfied is $\frac{1}{8}(= \frac{1}{4} \cdot \frac{1}{2})$ (recall that $a_{40,13}$ does not have effect on the condition $a_{40,13}a_{35,13} \oplus a_{35,13} \oplus a_{39,13} = 1$). If $a_{35,13} = 1$ and $a_{39,13} = 1$, then $a_{40,13} = 1$ and $a_{34,13} = 0$ due to the conditions $a_{40,13}a_{35,13} \oplus a_{35,13} \oplus a_{39,13} = 1$ and $a_{34,13} \neq a_{39,13}$. However, this is a contradiction to the condition of the 38-th step (see Table 4), and thus if $a_{35,13} = 1$, then $a_{39,13} = 0$, $a_{40,13} = 1$ and $a_{34,13} = 1$. The probability that $a_{35,13} = 1$ and $a_{34,13} = 1$ hold is $\frac{1}{4}$ by Table 4 and each probability of $a_{39,13} = 0$ and $a_{40,13} = 1$ is $\frac{1}{2}$, so the probability that $(a_{34,13}, a_{35,13}, a_{39,13}, a_{40,13}) = (1, 1, 0, 1)$ is $\frac{1}{16}$. Therefore, we can compute the probability that the conditions in the 40-th and 41-st step hold is $\frac{3}{16}(= \frac{1}{8} + \frac{1}{16})$, leading to a total probability $2^{-3} \cdot \frac{3}{16}$ for the above 5 conditions. In this way, we analyze the probability that the sufficient conditions in Table 6 are satisfied is 2^{-114}.

Table 4. Possible values for the conditions on the 38-th, 40-th and 41-st step

step		$a_{32,13}$	$a_{34,13}$	$a_{35,13}$	probability
38		1	1	1	1/8
		0	1	0	1/8
		1	0	0	1/8
		0	0	0	1/8

step	$a_{34,13}$	$a_{35,13}$	$a_{39,13}$	$a_{40,13}$	probability
40, 41	1	0	0	0	$1/4 \times 1/2 \times 1/2$
	1	0	0	1	$1/4 \times 1/2 \times 1/2$
	1	1	0	1	$1/4 \times 1/2 \times 1/2$

3.2 Attack on 3-Pass HAVAL

The second preimage resistance on a hash function plays an important role to block the attacker to produce a second preimage when a meaningful and sensitive message (e.g. a finance-related message) is used. In literature, it is defined as follows:

Second preimage resistance on a hash function H. for any given message M, it is computationally infeasible to find another message M' satisfying $H(M) = H(M')$

It follows that the second preimage attack on a hash function exists if for a given message M there is an algorithm that finds another message M' such that $H(M) = H(M')$ with probability larger than 2^{-n}, where n is the bit-length of hash values. The second preimage attack on 3-pass HAVAL works due to our differential path;

- For a given message M, the probability that M holds the sufficient conditions listed in Table 6 is 2^{-114}.
- If the message M holds the sufficient conditions, then the message M' which only differs from M at the most significant bit of the 22-nd message word has a same hash value.

4 Partial Key-Recovery Attacks on HMAC/NMAC-3-Pass HAVAL

In this section, we present partial key recovery attacks on HMAC/NMAC-3-pass HAVAL, which works based on our differential path described in Section 3. More precisely, we show how to find the partial key K_1 of NMAC-3-pass HAVAL and $f(\bar{K} \oplus \mathsf{ipad})$ of HMAC-3-pass HAVAL (note that knowing $f(\bar{K} \oplus \mathsf{ipad})$ and $f(\bar{K} \oplus \mathsf{opad})$ allows to compute the MAC value for any message). Since HMAC = NMAC if $f(\bar{K} \oplus \mathsf{ipad}) = K_1$ and $f(\bar{K} \oplus \mathsf{opad}) = K_2$, we focus on the NMAC-3-pass HAVAL attack which finds K_1 with message/MAC pairs. Recall that K_1 is placed at the position of the initial state in NMAC. This implies that recovering the initial value of 3-pass HAVAL is equivalent to getting the partial key K_1 of NMAC-3-pass HAVAL.

The main idea behind of our attack is that the attacker can recover the initial state of NMAC-3-pass HAVAL (in our attack it is K_1) if he knows a 256-bit input value at any step of 3-pass HAVAL. This idea has firstly been introduced in [4]. In this section, we first find a_{16}, a_{18}, a_{21} and a_{23} which are used as a part of an input value to the 24-th step. Remaining four-word input values a_{17}, a_{19}, a_{20} and a_{22} to the 24-th step is then found by 2^{128} exhaustive searches. Let $a_{i,j}$ be the j-th bit of a_i and $\gamma_i = (a_{i-8} \ggg 11) \boxplus (t_i \ggg 7) \boxplus C$, where C is a constant used in step i (note $\gamma_i \boxplus M_i = a_i$).

The value a_{16} is then revealed by the following Algorithm.

Algorithm 1. In order to recover the value a_{16}, we use a condition $a_{16,31} = 0$ depicted in Table 6. The procedure goes as follows:

1. The attacker has access to the oracle $\mathcal{O}$ (=NMAC-3-pass HAVAL) and makes 2^{121} queries for 2^{120} message pairs $M = M_0, M_1, \cdots, M_{30}, M_{31}$ and $M' = M'_0, M'_1, \cdots, M'_{30}, M'_{31}$ that have the message difference given in Table 5. Among the 2^{120} message pairs, $M_0, M_1, \cdots, M_{15}$ and $M'_0, M'_1, \cdots, M'_{15}$ are all identically fixed, M_{16} and M'_{16} vary in all 2^{32} possible values, and 2^{88} message pairs in the remaining words $M_{17}, M_{18}, \cdots, M_{31}$ and $M'_{17}, M'_{18}, \cdots, M'_{31}$

are randomly chosen. In this case, what the attacker knows is that γ_{16} is identically fixed for all the 2^{120} message pairs even though he does not know the actual value γ_{16}.

2. For each candidate value γ_{16} in $0, 1, \cdots, 2^{32} - 1$;
   (a) Choose the message pairs (M, M') that make collisions for the corresponding MAC pairs.
   (b) Count the number of the message pairs chosen in Step 2(a) that satisfy $msb(\gamma_{16} \boxplus M_{16}) = 1$.
3. Output $\gamma_{16} \boxplus M_{16}$ as a_{16}, where γ_{16} has the least count number in Step 2 (b).

As mentioned before, this algorithm works due to our differential with probability 2^{-114}. Notice that our differential encompasses a sufficient condition $a_{16,31} = 0$, and each message pair among the 2^{120} message pairs satisfies the condition $a_{16,31} = 0$, our differential holds with probability 2^{-113} with respect to this message pair. If the message pair (M, M') makes the most significant bits of a_{16} and a'_{16} be 1, then the probability that the message pair (M, M') makes a collision is $2^{-121} (= 2^{-113} \cdot 2^{-8})$, for it forces additionally 8 more sufficient conditions in our collision producing differential. The reason is as follows. If $a_{16,31} = 1$, then a difference Δt_{23} is not zero, but $\pm 2^{31}$. However, this difference value can be canceled by the output difference of the Boolean function in the 31-st step. In this procedure, each of steps 24-31 requires one more additional condition, leading to total 8 additional conditions. Thus, the probability that the message pair (M, M') has a same MAC value is not a random probability but 2^{-121}, where the most significant bits of a_{16} and a'_{16} are 1. It follows that if the right γ_{16} is guessed, we expect $2^{-2} (= 2^{119} \cdot 2^{-121})$ collision pairs. On the other hand, if γ_{16} is wrongly guessed, the expectation of collision pairs is $2^5 (= 2^{118} \cdot 2^{-113} + 2^{118} \cdot 2^{-121})$, (note that in the group of the message pairs such that $msb(\gamma_{16} \boxplus M_{16}) = 1$ there are on average half message pairs satisfying the actual $a_{16,31} = 0$). Since the probability that a wrong γ_{16} does not cause any collision pair is $(1 - 2^{-113})^{2^{118}} \cdot (1 - 2^{-121})^{2^{118}} < (1 - 2^{-113})^{2^{118}} (\approx e^{-32}) < 2^{-32}$, we expect that there is no wrong γ_{16} which leads to no collision in Step 2. Hence, we can determine the right γ_{16}. To summarize, Algorithm 1 requires 2^{121} oracle queries (in Step 1) and 2^{32} memory bytes (the memory complexity of this attack is dominated by the counters for γ_{16}).

Next, we show how to recover the value a_{18}, for which we use the condition $a_{18,31} = 0$ required in our differential. Since there is no condition on a_{17} (see Table 6), the attacker chooses any message word M_{17}. The main idea is similar to Algorithm 1.

First of all, the attacker selects 2^{119} message pairs (M, M') that have the message difference given in Table 6, where $M_0, M_1, \cdots, M_{17}$ and $M'_0, M'_1, \cdots, M'_{17}$ are all identically fixed ($M_0, M_1, \cdots, M_{16}$ and $M'_0, M'_1, \cdots, M'_{16}$ should be the same as those selected in Algorithm 1, which leads to $a_{16,31} = 0$), M_{18} and M'_{18} vary in all 2^{32} possible values, and 2^{87} message pairs in the remaining words are randomly chosen. Once the attacker gets the corresponding MAC pairs, he performs Steps 2 and 3 of Algorithm 1 to recover a_{18} by setting γ_{18}, M_{18}, a_{18}

instead of γ_{16}, M_{16} and a_{16}. The reason why recovering a_{18} requires half of the message pairs, compared to when recovering a_{16}, is that this attack algorithm exploits message pairs satisfying $a_{16,31} = 0$ from the beginning. It increases by twice the probability that our differential holds. The remaining analysis is the same as that of Algorithm 1. To summarize, recovering a_{18} requires 2^{120} oracle queries.

Next, let us see how to recover a_{21}. In order to recover a_{21} we need to use the condition $a_{20,31} = a_{21,31}$, which is of a different form from the previous two conditions $a_{16,31} = a_{18,31} = 0$. However, the core in our attack is that $a_{20,31}$ is always a same value if $M_0, M_1, \cdots, M_{20}$ and $M'_0, M'_1, \cdots, M'_{20}$ are all identically fixed in all required message pairs, i.e, in 2^{118} message pairs (note that all these message pairs satisfy $a_{16,31} = a_{18,31} = 0$, which the attacker can select from the above algorithms). Similarly, among the 2^{118} pairs, M_{21} and M'_{21} vary in all 2^{32} possible values and 2^{86} pairs of remaining words are randomly chosen.

Algorithm 2. The attack algorithm to recover a_{21} goes as follows:

1. The attacker chooses the 2^{118} message pairs as above and asks the oracle $\mathcal{O}$ for the corresponding 2^{118} MAC pairs.
2. Choose the message pairs (M, M') that make collisions for the corresponding MAC pairs.
3. For each candidate value γ_{21} in $0, 1, \cdots, 2^{32} - 1$;
   (a) Divide two groups of which one contains message pairs that satisfy $msb(\gamma_{21} \boxplus M_{21}) = 0$ and the other one contains message pairs that satisfy $msb(\gamma_{21} \boxplus M_{21}) = 1$.
   (b) Count the number of message pairs in each group that make collisions for the corresponding MAC pairs
4. Output $\gamma_{21} \boxplus M_{21}$ as a_{21}, where γ_{21} is the value that has a group containing the least count, and M_{21} is the one of the values satisfying $a_{20,31} = a_{21,31}$.

If the values a_{20} and a_{21} satisfy the sufficient condition $a_{20,31} = a_{21,31}$, then the probability that the message pair (M, M') makes a collision is 2^{-111} (note that the three conditions $a_{16,31} = a_{18,31} = 0, a_{21,31} = a_{20,31}$ are excluded in the list of the sufficient conditions). On the other hand, if $a_{21,31} \neq a_{20,31}$, then the probability that the message pair (M, M') makes a collision is 2^{-119} (similarly, 8 more conditions are additionally needed). In case the right γ_{21} is guessed, one of the two groups is expected to have $2^{-111} \cdot 2^{117} = 2^6$ collision pairs and the other one is expected to have $2^{-119} \cdot 2^{117} = 2^{-2}$ collision pair. On the other hand, if a wrong γ is guessed, then the both groups are expected to have $2^{-111} \cdot 2^{116} = 2^5$ collision pairs each. It implies that the probability that a wrong γ_{16} does not cause any collision pair is about $e^{-32} < 2^{-32}$, and thus there is no wrong γ_{21} to pass Step 3. To summarize, Algorithm 2 needs 2^{119} oracle queries and 2^{32} memory bytes. Recovering a_{23} is quite similar to recovering a_{16} and a_{18}, which requires 2^{118} oracle queries.

Exhaustive search for the remaining four words. Using the above algorithms, we can compute the 128-bit a_{16}, a_{18}, a_{21} and a_{23} values. The remaining 128-bit a_{17}, a_{19}, a_{20} and a_{22} values are found by the following algorithm. We consider a message pair (M, M') selected from the above algorithms which makes a collision.

1. Guess a 128-bit a_{17}, a_{19}, a_{20}, a_{22} value;
   (a) Check with the computed $a_{16}, a_{18}, a_{21}, a_{23}$ and the guessed $a_{17}, a_{19}, a_{20}, a_{22}$ that the message pair (M, M') makes a collision. If so, we determine the guessed value as the right value. Otherwise, repeat Step 1.
   (b) For the given message pair (M, M') and $a_{16}, a_{17}, \cdots, a_{22}$, recover the initial value.

If a wrong value is guessed, the probability that it causes a collision is 2^{-256}. Since the number of wrong $a_{17}, a_{19}, a_{20}, a_{22}$ tested in the attack is 2^{128} at most, we can recover the right initial value. The time complexity of the exhaustive search step is 2^{128} 3-pass HAVAL computations.

As a result, our partial key recovery attack has $2^{121}+2^{120}+2^{119}+2^{118} = 2^{121.9}$ oracle queries and 2^{128} 3-pass HAVAL computations.

Reducing the number of the 3-pass HAVAL computations. As described above, our partial key-recovery attack is completed by two phases; the first phase is to recover some portions of the 256-bit input value at step i, and the second is the exhaustive search phase for its remaining input bits. If we apply our attack to the input value to step 29 instead of step 24, then we can recover $a_{21}, a_{23}, a_{24}, a_{26}$ and a_{28} from the first phase with 2^{122} oracle queries and we recover the remaining a_{22}, a_{25} and a_{27} with 2^{96} 3-pass HAVAL computations from the second phase.

5 Conclusion

In this paper, we have presented a new second preimage differential path of 3-pass HAVAL with probability 2^{-114} and exploited it to devise a second preimage attack on 3-pass HAVAL, and partial key-recovery attacks on HMAC/NMAC-3-pass HAVAL with 2^{122} oracle queries, $5 \cdot 2^{32}$ memory bytes and 2^{96} 3-pass HAVAL computations. We expect that our attacks would be useful for the further analysis of HAVAL and HMAC/NMAC-HAVAL (e.g., full key-recovery attacks on HMAC/NMAC-HAVAL).

References

1. E. Biham and R. Chen, "Near-Collisions of SHA-0", *Advances in Cryptology – CRYPTO 2004*, LNCS 3152, Springer-Verlag, pp. 290–305.
2. E. Biham, R. Chen, A. Joux, P. Carribault, C. Lemuet and W. Jalby, "Collisions of SHA-0 and Reduced SHA-1", *Advances in Cryptology – EUROCRYPT 2005*, LNCS 3494, Springer-Verlag, pp. 22–35.

3. B. D. Boer and A. Bosselaers, "Collisions for the Compression Function of MD-5", *Advances in Cryptology – EUROCRYPT 1993*, LNCS 765, Springer-Verlag, pp. 293–304.
4. S. Contini and Y. L. Yin, "Forgery and Partial Key-Recovery Attacks on HMAC and NMAC Using Hash Collisions", *Advances in Cryptology – ASIACRYPT 2006*, LNCS 4284, Springer-Verlag, pp. 37–53.
5. H. Dobbertin, "Cryptanalsis of MD4", *FSE 1996*, LNCS 1039, Springer-Verlag, pp. 53–69.
6. P. A. Fouque, G. Leurent and P. Q. Nguyen, "Full Key-Recovery Attacks on HMAC/NMAC-MD4 and NMAC-MD5", *Advances in Cryptology – CRYPTO 2007*, LNCS 4622, Springer-Verlag, pp. 13–30.
7. J. Kim, A. Biryukov, B. Preneel and S. Hong, "On the Security of HMAC and NMAC Based on HAVAL, MD4, MD5, SHA-0 and SHA-1", *Security and Cryptography for Networks – SCN 2006*, LNCS 4116, Springer-Verlag, pp. 242–256.
8. C. Rechberger and V. Rijmen, "On Authentication With HMAC and Non-Rondom Properties". *Financial Cryptography and Data Security – FC 2007*, to appear.
9. B. Van Rompay, A. Biryukov, B. Preneel and J. Vandewalle, "Cryptanalysis of 3-pass HAVAL," *Advances in Cryptology – ASIACRYPT 2003*, LNCS 2894, Springer-Verlag, pp. 228–245, 2003.
10. X. Wang, D. Feng, X. Lai and H. Yu, "Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD", *Cryptology ePrint Archive*, Report 2004/199.
11. X. Wang, D. Feng and H. Yu, "The Collision Attack on Hash Function HAVAL-128", *Science in China, Series E*, Vol. 35(4), pp. 405–416, April, 2005.
12. X. Wang, X. Lai, D. Feng, H. Chen and X. Yu. "Cryptanalysis of the Hash Functions MD4 and RIPEMD", *Advances in Cryptology – EUROCRYPT 2005*, LNCS 3494, Springer-Verlag, pp. 1–18.
13. X. Wang, X. Y. Yin and H. Yu, "Finding Collision in the Full SHA-1", *Advances in Cryptology – CRYPTO 2005*, LNCS 3621, Springer-Verlag, pp. 17–36.
14. X. Wang, H. Yu and X. Y. Yin, "Efficient Collision Search Attacks on SHA-0", *Advances in Cryptology – CRYPTO 2005*, LNCS 3621, Springer-Verlag, pp. 1–16.
15. X. Wang and H. Yu, "How to Break MD5 and Other Hash Functions", *Advances in Cryptology – EUROCRYPT 2005*, LNCS 3494, Springer-Verlag, pp. 19–35
16. H. Yu, "Cryptanalysis of Hash Functions and HMAC/NMAC", *Doctoral dissertation*, SHANDONG
17. H. Yu, X. Wang, A. Yun and S. Park, "Cryptanalysis of the Full HAVAL with 4 and 5 Passes", *FSE 2006*, LNCS 4047, Springer-Verlag, pp. 89–110.
18. H. Yu, G. Wang, G. Zhang and X. Wang, "The Second-Preimage Attack on MD4", *CANS 2005*, LNCS 3810, Springer-Verlag, pp. 1-12.
19. Y. Zheng, J. Pieprzyk and J. Seberry, "HAVAL - a one-way hashing algorithm with variable length of output", *Advances in Cryptology – AUSCRYPT 1992*, LNCS 718, Springer-Verlag , pp. 83–104.

A Sufficient Conditions of the Second Preimage Differential Path of 3-Pass HAVAL

Table 5 shows the sufficient conditions of the second preimage differential path of 3-pass HAVAL, which are derived from the property of the Boolean function f_i of appendix B. We solve and simplify the conditions of Table 5 and list the solutions in Table 6.

Table 5. Sufficient conditions of the second preimage differential path of 3-pass HAVAL

S	Sufficient conditions
23	$a_{16,31} = 0, a_{22,31} = 0$
24	$a_{18,31} = 0$
25	$a_{20,31} = a_{21,31}$
26	$a_{23,31} = 0$
27	$a_{24,31} = 1$
28	$a_{26,31} = 0$
29	$a_{28,31} = 0$
31	$a_{30,20} = 0,\ a_{24,20} = 0$
32	$a_{29,20}a_{26,20} \oplus a_{28,20} \oplus a_{29,20} = 0$
33	$a_{31,20}a_{27,20} \oplus a_{32,20} \oplus a_{31,20} = 0$
34	$a_{33,20}a_{28,20} \oplus a_{28,20} \oplus a_{32,20} = 0$
35	$a_{29,20} = 0$
36	$a_{35,20}a_{32,20} \oplus a_{34,20}a_{33,20} \oplus a_{32,20} \oplus a_{31,20} \oplus a_{35,20} = 0$
37	$a_{31,20} = 0,\ a_{33,20}a_{35,20} \oplus a_{36,20}a_{34,20} \oplus a_{35,20}a_{34,20} = 0$
38	$a_{37,13} = 0,\ a_{34,13}a_{32,13} \oplus a_{35,13} = 0$
39	$a_{38,9} = 0,\ a_{35,9}a_{33,9} \oplus a_{36,9} = 0,\ a_{36,13}a_{33,13} \oplus a_{35,13} \oplus a_{36,13} = 0$
40	$a_{37,9}a_{34,9} \oplus a_{36,9} \oplus a_{37,9} = 0,\ a_{38,13}a_{34,13} \oplus a_{39,13} \oplus a_{38,13} = 0$
41	$a_{39,9}a_{35,9} \oplus a_{40,9} \oplus a_{39,9} = 0,\ a_{40,13}a_{35,13} \oplus a_{35,13} \oplus a_{39,13} = 0$
42	$a_{41,9}a_{36,9} \oplus a_{36,9} \oplus a_{40,9} = 0,\ a_{36,13} = 1$
43	$a_{42,6} = 0,\ a_{37,9} = 0,\ a_{39,6}a_{37,6} \oplus a_{40,6} = 0,$ $a_{42,13}a_{39,13} \oplus a_{41,13}a_{40,13} \oplus a_{39,13} \oplus a_{38,13} \oplus a_{36,13} = 0$
44	$a_{41,6}a_{38,6} \oplus a_{40,6} \oplus a_{41,6} = 0,\ a_{38,13} = 1,\ a_{43,9}a_{40,9} \oplus a_{42,9}a_{41,9} \oplus a_{40,9} \oplus a_{39,9} \oplus a_{37,9} = 0$
45	$a_{43,6}a_{39,6} \oplus a_{44,6} \oplus a_{43,6} = 0,\ a_{39,9} = 0,$ $a_{44,9}a_{41,9}a_{39,9} \oplus a_{39,9}a_{43,9}a_{42,9} \oplus a_{41,9}a_{43,9} \oplus a_{39,9}a_{40,9} \oplus a_{44,9}a_{41,9} \oplus a_{43,9}a_{42,9} = 1$
46	$a_{45,6}a_{40,6} \oplus a_{40,6} \oplus a_{44,6} = 0$
47	$a_{46,30} = 0,\ a_{43,30}a_{41,30} \oplus a_{44,30} = 0,\ a_{41,6} = 0$
48	$a_{45,30}a_{42,30} \oplus a_{44,30} \oplus a_{45,30} = 0,\ a_{47,6}a_{44,6} \oplus a_{46,6}a_{45,6} \oplus a_{44,6} \oplus a_{43,6} \oplus a_{41,6} = 0$
49	$a_{47,30}a_{43,30} \oplus a_{48,30} \oplus a_{47,30} = 0,\ a_{43,6} = 1$
50	$a_{49,30}a_{44,30} \oplus a_{44,30} \oplus a_{48,30} = 0$
51	$a_{50,27} = 1,\ a_{50,28} = 0,\ a_{47,27}a_{45,27} \oplus a_{48,27} = 0,\ a_{47,28}a_{45,28} \oplus a_{48,28} = 0,\ a_{45,30} = 0$
52	$a_{49,27}a_{46,27} \oplus a_{48,27} \oplus a_{49,27} = 0,\ a_{49,28}a_{46,28} \oplus a_{48,28} \oplus a_{49,28} = 0,$ $a_{51,30}a_{48,30} \oplus a_{50,30}a_{49,30} \oplus a_{48,30} \oplus a_{47,30} \oplus a_{45,30} = 0$
53	$a_{51,27}a_{47,27} \oplus a_{52,27} \oplus a_{51,27} = 0,\ a_{51,28}a_{47,28} \oplus a_{52,28} \oplus a_{51,28} = 0,\ a_{47,30} = 0$
54	$a_{53,23} = 0,\ a_{50,23}a_{48,23} \oplus a_{51,23} = 0,\ a_{53,27}a_{48,27} \oplus a_{48,27} \oplus a_{52,27} = 0,$ $a_{53,28}a_{48,28} \oplus a_{48,28} \oplus a_{52,28} = 0$
55	$a_{54,19} = 0,\ a_{51,19}a_{49,19} \oplus a_{52,19} = 0,\ a_{49,23} \oplus a_{51,23} = 0,\ a_{49,27} = 0,\ a_{49,28} = 1$
56	$a_{55,21} = 0,\ a_{52,21} = 1,\ a_{50,21} = 1,\ \oplus a_{53,21} = 1,\ a_{53,19}a_{50,19} \oplus a_{52,19} \oplus a_{53,19} = 0,$ $a_{50,23} \oplus a_{55,23} = 1,\ a_{55,27}a_{52,27} \oplus a_{54,27}a_{53,27} \oplus a_{52,27} \oplus a_{51,27} \oplus a_{49,27} = 0,$ $a_{55,28}a_{52,28} \oplus a_{54,28}a_{53,28} \oplus a_{52,28} \oplus a_{51,28} \oplus a_{49,28} = 0,\ a_{52,21}a_{51,21} \oplus a_{51,21} = 0$
57	$a_{56,14} = 1,\ a_{56,15} = 0,\ a_{54,21}a_{51,21} \oplus a_{53,21} \oplus a_{54,21} = 0,\ a_{56,23}a_{51,23} \oplus a_{51,23} \oplus a_{55,23} = 0,$ $a_{55,19}a_{51,19} \oplus a_{56,19} \oplus a_{55,19} = 0,\ a_{51,27} = 1,\ a_{51,28} = 1$
58	$a_{55,14}a_{52,14} \oplus a_{55,14} \oplus a_{54,14} = 0,\ a_{55,15}a_{52,15} \oplus a_{55,15} \oplus a_{54,15} = 0,$ $a_{56,21}a_{52,21} \oplus a_{57,21} \oplus a_{56,21} = 0,\ a_{57,19}a_{52,19} \oplus a_{52,19} \oplus a_{56,19} = 0,$ $a_{57,23} \oplus a_{56,23} \oplus a_{57,23}a_{55,23} = 0,\ a_{52,23} = 1$
59	$a_{57,14}a_{53,14} \oplus a_{58,14} \oplus a_{57,14} = 0, a_{57,15}a_{53,15} \oplus a_{58,15} \oplus a_{57,15} = 0,\ a_{53,19} = 0,$ $a_{58,21}a_{53,21} \oplus a_{53,21} \oplus a_{57,21} = 0,\ a_{58,23}a_{55,23} \oplus a_{57,23}a_{56,23} \oplus a_{51,23} = 1$
60	$a_{59,14}a_{54,14} \oplus a_{54,14} = 0,\ a_{59,15}a_{54,15} \oplus a_{54,15} \oplus a_{58,15} = 0,$ $a_{59,19}a_{56,19} \oplus a_{58,19}a_{57,19} \oplus a_{56,19} \oplus a_{55,19} \oplus a_{53,19} = 0,\ a_{54,21} = 0,\ a_{54,23} = 1$
61	$a_{55,14} = 0,\ a_{55,15} = 0,\ a_{55,19} = 0,\ a_{60,21}a_{57,21} \oplus a_{59,21}a_{58,21} \oplus a_{57,21} \oplus a_{56,21} \oplus a_{53,21} = 0$ $a_{59,19}a_{57,19} \oplus a_{60,19}a_{58,19} \oplus a_{59,19}a_{58,19} = 1$
62	$a_{61,14}a_{58,14} \oplus a_{60,14}a_{59,14} \oplus a_{58,14} \oplus a_{57,14} \oplus a_{61,14} = 0,\ a_{60,15}a_{59,15} \oplus a_{58,15} = 1$ $a_{61,15}a_{58,15} \oplus a_{60,15}a_{59,15} \oplus a_{58,15} \oplus a_{57,15} \oplus a_{61,15} = 1,\ a_{56,21} = 1$
63	$a_{57,14} = 1,\ a_{57,15} = 1$
64	$a_{60,10} = 0,\ a_{63,10} = 0,\ a_{61,10}a_{58,10} \oplus a_{62,10}a_{59,10} = 1$
67	$a_{62,10}a_{61,10} \oplus a_{60,10} \oplus a_{66,10} = 0$
68	$a_{64,10}a_{62,10} \oplus a_{66,10} = 0$
69	$a_{65,10}a_{64,10} \oplus a_{66,10} = 0$
70	$a_{66,10} = 0$

Table 6. Simplified sufficient conditions of the second preimage differential path of 3-pass HAVAL

S	Sufficient conditions
	$a_{16,31}=0,\ a_{18,31}=0,\ a_{20,31}=a_{21,31}, a_{22,31}=0$
23	$a_{23,31}=0$
24	$a_{24,31}=1,\ a_{24,20}=0$
26	$a_{26,31}=0$
28	$a_{28,31}=0,\ a_{28,20}=0$
29	$a_{29,20}=0$
30	$a_{30,20}=0$
31	$a_{31,20}=0$
32	$a_{32,20}=0$
36	$a_{36,13}=1,\ a_{36,9}=0,\ a_{34,20}a_{33,20}\oplus a_{35,20}=0$
37	$a_{37,9}=0,\ a_{37,6}=0,\ a_{37,13}=0,\ a_{33,20}a_{35,20}\oplus a_{36,20}a_{34,20}\oplus a_{35,20}a_{34,20}=0$
38	$a_{38,13}=1,\ a_{38,9}=0,\ a_{34,13}a_{32,13}\oplus a_{35,13}=0,$
39	$a_{39,6}=1,\ a_{39,9}=0,\ a_{35,9}a_{33,9}=0,\ a_{33,13}\neq a_{35,13}$
40	$a_{40,9}=0,\ a_{40,6}=0,\ a_{34,13}\neq a_{39,13}$
41	$a_{41,6}=0,\ a_{40,13}a_{35,13}\oplus a_{35,13}\oplus a_{39,13}=0$
42	$a_{42,6}=0$
43	$a_{43,6}=1,\ a_{42,13}a_{39,13}\oplus a_{41,13}a_{40,13}\oplus a_{39,13}=0$
44	$a_{44,30}=0,\ a_{44,6}=0,\ a_{42,9}a_{41,9}=0$
45	$a_{45,6}=1,\ a_{45,30}=0,\ a_{41,9}a_{43,9}\oplus a_{44,9}a_{41,9}\oplus a_{43,9}a_{42,9}=1$
46	$a_{46,6}=1,\ a_{46,30}=0$
47	$a_{47,30}=0,\ a_{43,30}a_{41,30}=0$
48	$a_{48,30}=0$
49	$a_{49,27}=0,\ a_{49,28}=1$
50	$a_{50,21}=1,\ a_{50,27}=1,\ a_{50,28}=0$
51	$a_{51,27}=1,\ a_{51,28}=1,\ a_{51,15}=0,\ a_{51,14}=0,$ $a_{47,27}a_{45,27}\oplus a_{48,27}=0,\ a_{47,28}a_{45,28}\oplus a_{48,28}=0$
52	$a_{52,19}=0,\ a_{52,21}=1, a_{50,30}a_{49,30}=0,\ a_{52,23}=1,\ a_{48,27}=0,\ a_{46,28}=a_{48,28}$
53	$a_{53,19}=0,\ a_{53,14}=0,\ a_{53,15}=1,\ a_{53,21}=0,\ a_{53,23}=1,\ a_{47,28}\oplus a_{52,28}=1$
54	$a_{54,14}=0,\ a_{54,15}=0,\ a_{54,19}=0,\ a_{50,23}a_{48,23}\oplus a_{51,23}=0,\ a_{52,27}=0,$ $a_{54,21}=0,\ a_{54,23}=1,\ a_{53,28}a_{48,28}\oplus a_{48,28}\oplus a_{52,28}=0$
55	$a_{55,14}=0,\ a_{55,15}=0,\ a_{55,19}=0,\ a_{55,21}=0,\ a_{51,19}a_{49,19}=0,\ a_{49,23}=a_{51,23}$
56	$a_{56,19}=0,\ a_{56,15}=0,\ a_{56,14}=1,\ a_{56,21}=1,\ a_{50,23}=a_{55,23}$ $a_{54,27}a_{53,27}\oplus a_{55,27}=1,\ a_{55,28}a_{52,28}\oplus a_{54,28}a_{53,28}\oplus a_{52,28}\oplus a_{51,28}\oplus a_{55,28}=0$
57	$a_{57,14}=1,\ a_{57,15}=1,\ a_{57,21}=0,\ a_{56,23}a_{51,23}\oplus a_{51,23}\oplus a_{55,23}=0$
58	$a_{58,21}=1,\ a_{58,14}=0, a_{58,15}=0,\ a_{57,23}\oplus a_{56,23}\oplus a_{57,23}a_{55,23}=0$
59	$a_{59,21}=1,\ a_{58,23}a_{55,23}\oplus a_{57,23}a_{56,23}\oplus a_{55,23}\oplus a_{51,23}=1,\ a_{59,15}=1$
60	$a_{58,19}a_{57,19}\oplus a_{55,19}=0,\ a_{60,10}=0,\ a_{60,15}=1$
61	$a_{59,19}a_{57,19}\oplus a_{60,19}a_{58,19}\oplus a_{59,19}a_{58,19}=1,\ a_{61,15}=1$
62	$a_{60,14}a_{59,14}\oplus a_{61,14}=1$
63	$a_{63,10}=0$
64	$a_{61,10}a_{58,10}\oplus a_{62,10}a_{59,10}=1$
66	$a_{66,10}=0$
67	$a_{62,10}a_{61,10}=0$
68	$a_{64,10}a_{62,10}=0$
69	$a_{65,10}a_{64,10}=0$

B Property of the Boolean Functions f_1, f_2 and f_3

Recall that the input value of the i-th step is denoted $a_{i-8}, a_{i-7}, \cdots, a_{i-1}$ and the output value of the Boolean functions of the i-th step is denoted t_i. Tables 7, 8 and 9 show the relations between the input difference and t_i of the i-th step. In the column of Assumption in the Tables, $\mathbf{a_s[j]}$ represents the difference $(\Delta a_{i-8}, a_{i-7}, \cdots, \Delta a_s, \cdots, \Delta a_{i-1}) = (0, 0, \cdots, a_s[j], 0, \cdots, 0)$ for $i-1 \leq s \leq i-7$ and $t_i[]$ means that the output difference of the Boolean function of the i-th step is zero (see Section 2.3 for the notations $a_s[j]$ and $t_i[j]$). Note that even though the sign is altered from $+j$ to $-j$ in both $\mathbf{a_s[j]}$ and $t_i[j]$, still the conditions are the same as in Tables 7, 8 and 9, however if the sign is altered only in one of $\mathbf{a_s[j]}$ and $t_i[j]$, the second conditions should be 1 (and the first ones are not altered).

Table 7. Property of the Boolean function f_1

Assumption		Conditions for satisfying the Assumption
	$t_i[]$	$a_{i-7} = 0$
$\mathbf{a_{i-1}[j]}$	$t_i[j]$	$a_{i-7} = 1, a_{i-3}a_{i-4} \oplus a_{i-2}a_{i-6} \oplus a_{i-5}a_{i-3} \oplus a_{i-5} = 0$
	$t_i[]$	$a_{i-6} = 0$
$\mathbf{a_{i-2}[j]}$	$t_i[j]$	$a_{i-6} = 1,\ a_{i-3}a_{i-4} \oplus a_{i-7}a_{i-1} \oplus a_{i-5}a_{i-3} \oplus a_{i-5} = 0$
	$t_i[]$	$a_{i-4} = a_{i-5}$
$\mathbf{a_{i-3}[j]}$	$t_i[j]$	$a_{i-4} \neq a_{i-5},\ a_{i-1}a_{i-7} \oplus a_{i-6}a_{i-2} \oplus a_{i-5} = 0$
	$t_i[]$	$a_{i-3} = 0$
$\mathbf{a_{i-4}[j]}$	$t_i[j]$	$a_{i-3} = 1,\ a_{i-1}a_{i-7} \oplus a_{i-6}a_{i-2} \oplus a_{i-5}a_{i-3} \oplus a_{i-5} = 0$
	$t_i[]$	$a_{i-3} = 1$
$\mathbf{a_{i-5}[j]}$	$t_i[j]$	$a_{i-3}a_{i-4} \oplus a_{i-3} = 0,\ a_{i-1}a_{i-7} \oplus a_{i-6}a_{i-2} = 0$
	$t_i[]$	$a_{i-2} = 0$
$\mathbf{a_{i-6}[j]}$	$t_i[j]$	$a_{i-2} = 1,\ a_{i-3}a_{i-4} \oplus a_{i-7}a_{i-1} \oplus a_{i-5}a_{i-3} \oplus a_{i-5} = 0$
	$t_i[]$	$a_{i-1} = 0$
$\mathbf{a_{i-7}[j]}$	$t_i[j]$	$a_{i-1} = 1,\ a_{i-3}a_{i-4} \oplus a_{i-6}a_{i-2} \oplus a_{i-5}a_{i-3} \oplus a_{i-5} = 0$

Table 8. Property of the Boolean function f_2

Assumption		Conditions for satisfying the Assumption
$\mathbf{a_{i-1}[j]}$	$t_i[]$	$a_{i-4}a_{i-6} \oplus a_{i-3} = 0$
	$t_i[j]$	$a_{i-4}a_{i-6} \oplus a_{i-3} = 1$, $a_{i-6}a_{i-2}a_{i-3} \oplus a_{i-4}a_{i-6} \oplus a_{i-4}a_{i-2} \oplus a_{i-6}a_{i-5} \oplus a_{i-2}a_{i-3} \oplus a_{i-7}a_{i-6} \oplus a_{i-7} = 0$
$\mathbf{a_{i-2}[j]}$	$t_i[]$	$a_{i-3}a_{i-6} \oplus a_{i-4} \oplus a_{i-3} = 0$
	$t_i[j]$	$a_{i-3}a_{i-6} \oplus a_{i-4} \oplus a_{i-3} = 1$, $a_{i-4}a_{i-6}a_{i-1} \oplus a_{i-4}a_{i-6} \oplus a_{i-6}a_{i-5} \oplus a_{i-1}a_{i-3} \oplus a_{i-7}a_{i-6} \oplus a_{i-7} = 0$
$\mathbf{a_{i-3}[j]}$	$t_i[]$	$a_{i-2}a_{i-6} \oplus a_{i-1} \oplus a_{i-2} = 0$
	$t_i[j]$	$a_{i-2}a_{i-6} \oplus a_{i-1} \oplus a_{i-2} = 1$, $a_{i-4}a_{i-6}a_{i-1} \oplus a_{i-4}a_{i-6} \oplus a_{i-6}a_{i-5} \oplus a_{i-7}a_{i-6} \oplus a_{i-7} = 0$
$\mathbf{a_{i-4}[j]}$	$t_i[]$	$a_{i-1}a_{i-6} \oplus a_{i-6} \oplus a_{i-2} = 0$
	$t_i[j]$	$a_{i-1}a_{i-6} \oplus a_{i-6} \oplus a_{i-2} = 1$ $a_{i-6}a_{i-2}a_{i-3} \oplus a_{i-6}a_{i-5} \oplus a_{i-1}a_{i-3} \oplus a_{i-2}a_{i-3} \oplus a_{i-7}a_{i-6} \oplus a_{i-7} = 0$
$\mathbf{a_{i-5}[j]}$	$t_i[]$	$a_{i-6} = 0$
	$t_i[j]$	$a_{i-6} = 1$, $a_{i-4}a_{i-6}a_{i-1} \oplus a_{i-6}a_{i-2}a_{i-3} \oplus a_{i-4}a_{i-6} \oplus a_{i-4}a_{i-2} \oplus a_{i-1}a_{i-3} \oplus a_{i-2}a_{i-3} \oplus a_{i-7}a_{i-6} \oplus a_{i-7} = 0$
$\mathbf{a_{i-6}[j]}$	$t_i[]$	$a_{i-1}a_{i-4} \oplus a_{i-2}a_{i-3} \oplus a_{i-4} \oplus a_{i-5} \oplus a_{i-7} = 0$
	$t_i[j]$	$a_{i-1}a_{i-4} \oplus a_{i-2}a_{i-3} \oplus a_{i-4} \oplus a_{i-5} \oplus a_{i-7} = 1$, $a_{i-4}a_{i-2} \oplus a_{i-1}a_{i-3} \oplus a_{i-2}a_{i-3} \oplus a_{i-7} = 0$
$\mathbf{a_{i-7}[j]}$	$t_i[]$	$a_{i-6} = 1$
	$t_i[j]$	$a_{i-6} = 0$, $a_{i-4}a_{i-5}a_i \oplus a_{i-6}a_{i-2}a_{i-3} \oplus a_{i-4}a_{i-6} \oplus a_{i-4}a_{i-2} \oplus a_{i-6}a_{i-5} \oplus a_{i-1}a_{i-3} \oplus a_{i-2}a_{i-3} = 0$

Table 9. The property of the Boolean function f_3

Assumption		Conditions for satisfying the Assumption
$\mathbf{a_{i-1}[j]}$	$t_i[]$	$a_{i-4}=1$
	$t_i[j]$	$a_{i-4}=0, a_{i-4}a_{i-5}a_{i-6}\oplus a_{i-6}a_{i-3}\oplus a_{i-5}a_{i-2}\oplus a_{i-4}a_{i-7}=0$
$\mathbf{a_{i-2}[j]}$	$t_i[]$	$a_{i-5}=0$
	$t_i[j]$	$a_{i-5}=1,\ a_{i-6}a_{i-3}\oplus a_{i-4}a_{i-7}\oplus a_{i-1}a_{i-4}\oplus a_{i-1}=0$
$\mathbf{a_{i-3}[j]}$	$t_i[]$	$a_{i-6}=0$
	$t_i[j]$	$a_{i-6}=1,\ a_{i-4}a_{i-5}a_{i-6}\oplus a_{i-5}a_{i-2}\oplus a_{i-4}a_{i-7}\oplus a_{i-1}a_{i-4}\oplus a_{i-1}=0$
$\mathbf{a_{i-4}[j]}$	$t_i[]$	$a_{i-5}a_{i-6}\oplus a_{i-7}\oplus a_{i-1}=0$
	$t_i[j]$	$a_{i-5}a_{i-6}\oplus a_{i-7}\oplus a_{i-1}=1,\ a_{i-6}a_{i-3}\oplus a_{i-5}a_{i-2}\oplus a_{i-1}=0$
$\mathbf{a_{i-5}[j]}$	$t_i[]$	$a_{i-4}a_{i-6}\oplus a_{i-2}=0$
	$t_i[j]$	$a_{i-4}a_{i-6}\oplus a_{i-2}=1,\ a_{i-6}a_{i-3}\oplus a_{i-4}a_{i-7}\oplus a_{i-1}a_{i-4}\oplus a_{i-1}=0$
$\mathbf{a_{i-6}[j]}$	$t_i[]$	$a_{i-4}a_{i-5}\oplus a_{i-3}=0$
	$t_i[j]$	$a_{i-4}a_{i-5}\oplus a_{i-3}=1,\ a_{i-5}a_{i-2}\oplus a_{i-4}a_{i-7}\oplus a_{i-1}a_{i-4}\oplus a_{i-1}=0$
$\mathbf{a_{i-7}[j]}$	$t_i[]$	$a_{i-4}=0$
	$t_i[j]$	$a_{i-4}=1,\ a_{i-4}a_{i-5}a_{i-6}\oplus a_{i-6}a_{i-3}\oplus a_{i-5}a_{i-2}\oplus a_{i-1}a_{i-4}\oplus a_{i-1}=0$