# Distinguishing Attacks on MAC/HMAC Based on A New Dedicated Compression Function Framework *

Zheng Yuan[1,2,3],** Haixia Liu[1], Xiaoqiu Ren[1]

[1] Beijing Electronic Science and Technology Institute, Beijing 100070,China
[2] School of Telecommunications Engineering,Xidian university,Xi'an,Shanxi 710126,China
[3] Beijing University of Posts and Telecommunications, Beijing 100876,China
yuanzheng@besti.edu.cn
zyuan@mail.tsinghua.edu.cn

**Abstract.** A new distinguishing attack on HMAC and NMAC based on a dedicated compression function framework $\boldsymbol{H}$, proposed in ChinaCrypt2008, is first presented in this paper, which distinguish the HMAC/NMAC-$\boldsymbol{H}$ from HMAC/NMAC with a random function. The attack needs $2^{17}$ chosen messages and $2^{23}$ queries, with a success rate of 0.873. Furthermore, according to distinguishing attack on SPMAC-$\boldsymbol{H}$, a key recovery attack on the SPMAC-$\boldsymbol{H}$ is present, which recover all 256-bit key with $2^{17}$ chosen messages, $2^{19}$ queries, and $(t+1)\times 8$ times decrypting algorithms.
**Keywords:** distinguishing attacks, the block-collisions property, a dedicated compression function framework, HMAC, NMAC.

## 1 Introduction

Message Authentication Code (MAC) is a kind of fixed-length information used to ensure data integrity and authenticity. A MAC algorithm takes a secret key and a message of arbitrary length as input, and the output is a short digest. HMAC and NMAC are hash-based message authentication codes proposed by Bellare et al.[1]. NMAC is the theoretical foundation of HMAC, and HMAC has been implemented in widely-used protocols including SSL/TLS, SSH and IPsec. The securities of NMAC and HMAC has been carefully analyzed in[1,2,3]. It is proved that NMAC is a pseudo-random function family (PRF) assuming that the compression function of the keyed hash function in it be a PRF[3]. The security of HMAC is based on the security of NMAC, assume that both the compression function of the keyed hash function and the key derivation function

* Supported by the Beijing Natural Science Foundation (No.4102055), the National Natural Science Foundation of China (NSFC Grant No.61070250), the Foundation of State Key Laboratory of Information Security (Institute of Software, Chinese Academy of Sciences)(Nos.01-01, 01-02-6), the Foundation of Key Laboratory of Information Security of BESTI(Nos.YZDJ0905,YGCJ1004).
** To whom correspondence should be addressed.

in HMAC is a PRF, HMAC is also proved a PRF[1,3]. However, if the underlying hash function is weak, the above proofs may not work[4]. How to estimate the security of a hash function? Recently, NIST proposed the security requirements of hash functions[8], the details are as follows:

1. It may be provided in a wide variety of cryptographic applications, including digital signatures, key derivation, hash-based message authentication codes, deterministic random bit generators, and additional applications that may be brought up by NIST or by the public during the evaluation process.
2. It can support HMAC, PRF, and Randomized Hashing.
3. The hash algorithm of message digest size $n$ to meet the following security requirements at a minimum.
   • Collision resistance of approximately $n/2$ bits,
   • Preimage resistance of approximately $n$ bits,
   • Second-preimage resistance of approximately $n$-$k$ bits for any message shorter than $2^k$ bits,
   • Resistance to length-extension attacks, and
   • Any $m$-bit hash function specified by taking a fixed subset of the candidate function's output bits is expected to meet the above requirements with m replacing n.
   Additionally, increasing the second preimage resistance property and resistance against other attacks, such as multicollision attacks.
4. Evaluations relating to attack resistance.

In ChinaCrypt2008, a new dedicated compression function framework (i.e. hash function $\boldsymbol{H}$) and two improvement schemes for MD construction were proposed[7]. The authors asserted that their hash function $\boldsymbol{H}$ was secure, because two schemes had the properties of pseudo collision resistant which could withstand some attacks used the weakness of MD constructions. But they hadn't thought of the HMAC/NMAC based on their hash function $\boldsymbol{H}$, which is NIST's requirement criterion 2 above.

In this paper, we cryptanalyse the SPMAC/HMAC/NMAC-$\boldsymbol{H}$, and first present novel distinguishing attacks on them, which result in forgery attacks directly. Furthermore, our distinguishing attack on SPMAC-$\boldsymbol{H}$ can be applied to recover its secret key.

There are three types of the attacks on HMAC/NMAC, namely, distinguishing attacks, existential forgery attacks and universal forgery attacks. Distinguishing attacks can be divided into distinguishing-R and distinguishing-H attacks[6]. Distinguishing-R attack distinguishes a HMAC/NMAC from a random function, and distinguishing-H attack differentiates an instantiated HMAC/NMAC constructed by an underlying hash function or block cipher from a HMAC/NMAC constructed by a random function. In 1995, Preneel et al.[9] introduced a general distinguishing-R attack on all iterated MACs by the birthday paradox, requiring about $2^{\frac{n}{2}}$ messages with a success rate of 0.63, where $n$ is the length of the hash output. This distinguishing-R attack can be converted into a general forgery attack immediately. In 2006, Kim et al.[6] presented two kinds of distinguishers-H attack on the HMAC structure, the differential distinguisher and the rectangle distinguisher, and used them to analyze the security of HMAC based on HAVAL, MD4, MD5, SHA-0 and SHA-1. In 2009, Wang et al.[10,11,12,5]gave new distinguishing attacks on MAC/HMAC/NMAC, which detect an inner near-collision with differential paths of the cryptographic primitive embedded in a MAC/HMAC/NMAC. Using these distinguishing attacks, forgery attacks, second-preimage attacks, and recovery partial keys attacks might be done. All of these attacks with birthday attack complexity. This paper focuses on the distinguishing-H attack. For simplicity, we call it as distinguishing attack.

Inspired by Wang et al.works, new techniques to identify the underlying cryptographic primitive of HMAC/NMAC were proposed in this paper, which separate the output of the HMAC/NMAC into $j$ block, and adopt segmenting techniques to detect $j$ block-collisions, named the block-collisions property in this paper, with specific differential paths or not. The data complexity and computation complexity are far less than inner collision, pseudo-collisions, or near-collisions.

Using our techniques, new distinguishing attacks on SPMAC/HMAC/NMAC-***H*** are first shown, which regard the block-collisions property as the distinguisher to part SPMAC/HMAC/NMAC-***H*** from SPMAC/HMAC/NMAC-RF with only $2^{17}$ data. Furthermore, the distinguishing attack on SPMAC-***H*** can be applied to recover its secret key with same data complexity.

This paper is organized as follows: Section 2 gives backgrounds including notations, a brief descriptions of the hash function ***H*** and SPMAC/HMAC/NMAC Based on ***H***. In Section 3, a distinguishing attack on SPMAC/HMAC/NMAC-***H*** is present, a key recovery attack on the SPMAC-H is suggested in Section 4, and Section 6 is our conclusions.

## 2 Backgrounds

### 2.1 Notations

| | | |
|---|---|---|
| $K/K_i$ | : | the (initial 512-bit key/$i^{th}$ round subkey) of the block cipher $E$ |
| $i \in [1, 32]$ | : | the round number of the block cipher $E$ |
| $j \in [0, 7]$ | : | the $j$ block of a concatenation of eight blocks |
| $M$ | : | $M = M_0 \vert\vert \dots \vert\vert M_{t-1}$ consisting of $t$-block 512-bit message blocks |
| $M^x$ | : | the $x^{th}$ message, which have $t$ 512-block messages |
| $M^x_{tj}$ | : | the $j^{th}$ message expansion block of the $t^{th}$ 512-bit block of $M^x$ |
| $H(M)$ | : | the hash value of a message $M$ |
| $H^x_{tj}$ | : | the $j^{th}$ 32-bit block of the $t^{th}$ 256-bit chaining variable of the |
| | : | message $M^x$'s hash value |
| HMAC-***H*** | : | HMAC based on the hash function ***H*** |
| $k$ | : | 256-bit key of MACs based on the hash function ***H*** |
| $T$ | : | the MAC value of a message $M$ under the key $k$ |
| $T_j$ | : | the $j^{th}(0 \le j \le 7)$ 32-bit block of MAC value $T$ |
| $0^z$ | : | a concatenation of $z$ '0' |
| $P^{x'}_j$ | : | the $j^{th}(0 \le j \le 7)$ 32-bit block of 512-bit nor-zero string $P^{x'}$ |

### 2.2 A Brief Description of the Hash Function $\boldsymbol{H}$

We firstly introduce a new block cipher $E$ used to construct a new dedicated compression function $h$. Then describe the hash function $\boldsymbol{H}$ based on the compression function $h$.

**The Block Cipher $\boldsymbol{E}$.** The block cipher $E$ is an 32-bit block iteration algorithm possessing 32 same rounds, whose initial 512-bit key $K$ provides previous 16-round subkeys, and each subkey of the following 16 rounds can be computed from the formula: $K_i = (K_{i-3} \oplus K_{i-5} \oplus K_{i-8} \oplus K_{i-11} \oplus K_{i-14} \oplus K_{i-16})^{<<<1}, (i > 15)$.

Give a 4-byte plaintext $B = B_{00} \parallel B_{01} \parallel B_{02} \parallel B_{03}$, the $i^{th}$ round encrypting iteration algorithm of the block cipher $E$ is below:

1. XOR: byte $B_{iz} (0 \le z \le 3)$ XOR subkey byte $K_{iz}$:
$$B_i \oplus K_i = (B_{i0} \oplus K_{i0}) \| (B_{i1} \oplus K_{i1}) \| (B_{i2} \oplus K_{i2}) \| (B_{i3} \oplus K_{i3})$$
2. SubBytes: like the SubBytes of AES, the output byte $C_{iz} = f((B_{iz} \oplus K_{iz})^{-1})$, where $f \in GF(2^8)$ is a reversible affine transformation.
3. RowColumns: a 4×4 MDS matrix multiplies the four bytes $(C_{i0}, C_{i1}, C_{i2}, C_{i3})^T$, and the output is $B_{i+1,0}$:
$$\begin{pmatrix} B_{i+1,0} \\ B_{i+1,1} \\ B_{i+1,2} \\ B_{i+1,3} \end{pmatrix} = MDS_{4\times4} \times \begin{pmatrix} C_{i0} \\ C_{i1} \\ C_{i2} \\ C_{i3} \end{pmatrix} = MDS_{4\times4} \times \begin{pmatrix} f((B_{i0} \oplus K_{i0})^{-1}) \\ f((B_{i1} \oplus K_{i1})^{-1}) \\ f((B_{i2} \oplus K_{i2})^{-1}) \\ f((B_{i3} \oplus K_{i3})^{-1}) \end{pmatrix}$$
$B_{i+1} = B_{i+1,0} \parallel B_{i+1,1} \parallel B_{i+1,2} \parallel B_{i+1,3}$ is the $i^{th}$ round output, as well as the $(i+1)^{th}$ round input. Finally, after 32 rounds iteration operations, output is ciphertext $D = E_K(B)$.

**Construct A New Dedicated Compression Function $\boldsymbol{h}$ Using $\boldsymbol{E}$.** The compression function $h$ is a concatenation of eight block ciphers(See Fig.1), whose input are eight 32-bit initial vector $IV_y = IV_{y0} \parallel \ldots \parallel IV_{i7}$ and a 512-bit message block $M_y$, and output is a 256-bit value $H_y = h(M_y, IV_y)$. Four 128-bit message block $M_y$ expands eight 512-bit message expansion block $M_{yj}$ by the equations below:

$$M_{y0} = M_y = U_{y0} \parallel U_{y1} \parallel U_{y2} \parallel U_{y3} \tag{1}$$
$$M_{y1} = U_{y3} \parallel U_{y0} \parallel U_{y1} \parallel U_{y2} \tag{2}$$
$$M_{y2} = U_{y2} \parallel U_{y3} \parallel U_{y0} \parallel U_{y1} \tag{3}$$
$$M_{y3} = U_{y1} \parallel U_{y2} \parallel U_{y3} \parallel U_{y0} \tag{4}$$
$$M_{y4} = U_{y3} \parallel U_{y2} \parallel U_{y1} \parallel U_{y0} \tag{5}$$
$$M_{y5} = U_{y0} \parallel U_{y3} \parallel U_{y2} \parallel U_{y1} \tag{6}$$
$$M_{y6} = U_{y1} \parallel U_{y0} \parallel U_{y3} \parallel U_{y2} \tag{7}$$
$$M_{y7} = U_{y2} \parallel U_{y1} \parallel U_{y0} \parallel U_{y3} \tag{8}$$

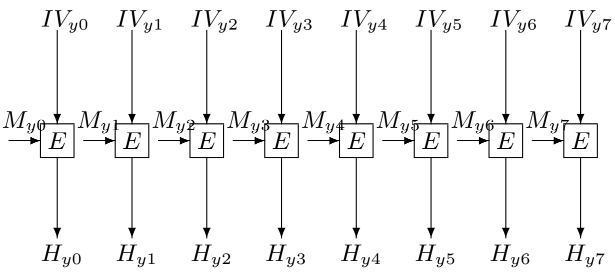


**Fig. 1.** The Construction of the New Dedicated Compression Function $h$

Obviously, 256-bit output $H_y$ is also a concatenation of eight 32-bit blocks, i.e. $H_y = H_{y0} \parallel \ldots || H_{y7} = E_{M_{y0}}(IV_{y0}) \parallel \ldots \parallel E_{M_{y7}}(IV_{y7})$.

**Construct the Hash Function *H* Using *h*.** Give a message $M = M_0 \parallel \ldots \parallel M_{t-1}$, then hash function $\boldsymbol{H}$ with the original MD construction is:

$$H_0 = IV_0; H_y = h(H_{y-1}, M_{y-1}), (0 \le y \le t); H(M) = g(H_t).$$

$H_{y-1}$ serves as the 256-bit chaining variable between stage $y-1$ and stage $y$, and $H_0$ is a pre-defined initializing value ($IV_0$). An optional output transformation $g$ is used in a final step to map the 256-bit chaining variable to an $m$-bit result $g(H_t)$. Without loss of generality, $g$ is the identity mapping $g(H_t) = H_t$, and output $H(M) = H_t$ is the hash value.

According to Fig.1 the hash value $H(M)$ is also a concatenations of eight 32-bit blocks, that is say, $H(M) = H_{t0}|| \ldots ||H_{t7}$.

### 2.3 Secret Prefix MAC, HMAC and NMAC Based on the Hash Function *H*

A MAC algorithm is a hash function with a secret key $k$ as one input. HMAC and NMAC are two popular MAC algorithms which are all derived from efficient hash functions.

Another three earlier MACs, based on hash, are constructed by the *Secret Prefix Method*, *Secret Suffix Method* and *Envelope Method. secret prefix* MAC is the keyed hash. We denote this kind of MAC based on the hash function $\boldsymbol{H}$ as SPMAC-H, which is defined as

$$SPMAC - H = H(M) = H(M, k)$$

SPMAC-H is an algorithm $A : \{0,1\}^{256} \times \{0,1\}^{512t} \longrightarrow \{0,1\}^{256}$, which is the basic design unit for HMAC/NMAC-H. Similarly, SPMAC-H can also be split into eight completely independent 32-bit blocks.

Let $(k_1, k_2)$ be an independent key pair, according to paper[1], NMAC algorithm is defined as: $NMAC_{(k_1,k_2)} = H_{k_1}(H_{k_2}(M))$. In fact, the outer function acts on the output of the iterated hash function, which is basically the compression function $H_{k_1}$ acting on $H_{k_2}(M)$. $H_{k_2}(M)$(maybe padded) is a full block size, so HMAC-H need pad one "1" following 255 "0" to $H_{k_2}(M)$, and is defined as:

$$NMAC - H = H_{k_1}(H_{k_2}(M)||10^{255})$$

If $k_1 = h(IV, \boldsymbol{k} \oplus opad)$ and $k_2 = h(IV, \boldsymbol{k} \oplus ipad)$, where $\boldsymbol{k}$ is obtained by padding number 0 at the end of $k$ to make the length $|\boldsymbol{k}| = b$. Both $ipad$ and $opad$ are $b$-bit constants, and get $ipad$ and $opad$ by repeating concatenating $0x5c$ and $0x36$, respectively. Thus $HMAC - H$ algorithm can be written as:

$$HMAC - H = NMAC_{k_1,k_2}(M) - H = H_{k_1}(H_{k_2}(M)||10^{255}).$$

For simplicity, we denote the $HMAC - \boldsymbol{H}$ by $H_{out}(H_{in}(M))$.

## 3 A Distinguishing Attack on HMAC/NMAC-*H*

We first present the block-collisions property of the hash function $\boldsymbol{H}$, which is the basis of our distinguishing attack.

### 3.1 The Block-collisions Property of the Hash Function *H*

**Property:** For $1 \leq x \leq 2^{16.5}$, randomly choose a structure $S_1 = \{M^x | M^x = (M_0^x \parallel \ldots \parallel M_{t-1}^x)\}$ composed of $2^{16.5}$ different messages, and compute their hash values $H^x(M^x) = H_t^x = E_{M_{t-1}^x}(H_{t-1}^x)$, which is eight 32-bit blocks combined. If $t$ is large enough to guarantee the *chaining variable* $H_{t-1}^x$ uniform, then each of 32-bit block $H_{(t)j}^x(0 \leq j \leq 7)$ has at least one collision pair $(H_{(t)j}^u, H_{(t)j}^v)$.
**proof:** Each of chaining variables $H_1^x, H_2^x, \ldots, H_{t-1}^x$ and hash values $H_t^x$ is a concatenations of eight 32-bit blocks, they can be obtained by eight coordinate block ciphers $E$ encryptions with the relevant eight message expansion blocks $M_{(y-1)0}^x, \ldots, M_{(y-1)7}^x$ as keys,respectively, i.e.

$$H_y^x = E_{M_{(y-1)0}^x}(H_{(y-1)0}^x) \parallel \ldots \parallel E_{M_{(y-1)7}^x}(H_{(y-1)7}^x)(0 < y < t),$$

and each message expansion block is determined by $M^x$ using Equ.(1)-(8), respectively. Thus, inputs of each block cipher $E$, $(H_{(y-1)j}^x, M_{(y-1)j}^x)(0 \leq j \leq 7)$, are independent each other.

It is recommended to choose $t = 10$ enable the chaining variable $H_{t-1}^x = H_{(t-1)0}^x \parallel \ldots \parallel H_{(t-1)7}^x$ uniform, which means that the eight input blocks of hash values $H_t^x$ are random. Block cipher $E$ is a permutation with 32-bit output, randomly $2^{16.5}$ inputs result in at least one block-collision pair $(H_{(t)j}^u, H_{(t)j}^v)$ by birthday attack[13].

### 3.2 An Adaptive Chosen Message Attack SPMAC-$\boldsymbol{H}$

To describe the distinguishing attack on HMAC/NMAC-$\boldsymbol{H}$, we start with a distinguishing attack on SPMAC-$\boldsymbol{H}$, which is an adaptive chosen message attack. Our method relies on the block-collisions property in Section 3.1. The core of our distinguishing attack is to detect eight output block-differences $(\Delta\overline{T_0}, \ldots, \Delta\overline{T_7}) = (\overline{T_0^a} \oplus \overline{T_0^b}, \ldots, \overline{T_7^o} \oplus \overline{T_7^p})(0 \leq j \leq 7)$, which are the block-differences of MAC values of $(M^u \parallel P^{x'}, M^v \parallel P^{x'})$. According to the block-collisions property, such message pair $(M^u, M^v)$ exists, and can be detected by padding $2^{16.5}$ 512-bit nor-zero string $P^{x'}$.

Construct a structure $S_1 = \{M^x \parallel M^x = (M_0^x \parallel \ldots \parallel M_{t-1}^x)\}$. Assume that the MAC algorithm is either a SPMAC-$\boldsymbol{H}$ or SPMAC with a random function (SPMAC-RF), then the distinguisher works in the following manner:

1. Collect $2^{17}$ randomly chosen messages, denoted $M^x \in S$, and append a 512-bit nor-zero string $P$ to each $M^x$. Query each of their MAC value $T^x$. If the MAC algorithm is SPMAC-$\boldsymbol{H}$, $T^x$ obviously is a concatenation of eight 32-bit blocks, written as $T^x = T_0^x \parallel \ldots \parallel T_7^x$, or else $T^x$ is not.
2. Find pairs $(M^u \parallel P, M^v \parallel P)$ such that at least one 32-bit output block colliding $T_j^u = T_j^v (0 \leq j \leq 7)$.
3. For each searched pairs $(M^u \parallel P, M^v \parallel P)$, padding another 512-bit nor-zero string $P' \neq P$ to $(M^u, M^v)$, ask for MACs values $(\overline{T^u}, \overline{T^v})$ of $(M^u \parallel P', M^v \parallel P')$, and save the messages pair $(M^u, M^v)$ whose the same $j^{th}$ block of MAC values does collided, that is, $\overline{T^u}_j \neq \overline{T^v}_j$.
4. Randomly choose $2^{16.5}$ $P^{x'} \neq P$, append $P^{x'}$ to one pair $(M^u, M^v)$ remained in step 3. Inquire MAC values $(\overline{T^u}, \overline{T^v})$ of $(M^u \parallel P^{x'}, M^v \parallel P^{x'})$, and check whether there exists the block-collisions property:
   (a) For $\forall j \in [0, 7]$, if at least one MAC pair block collides, satisfying that $\overline{T_j^u} = \overline{T_j^v}$. Output is the SPMAC-$\boldsymbol{H}$.
   (b) Otherwise, it is SPMAC-RF.

**Complexity.** This attack requires about $2^{17}$ chosen messages, at most $2^{17} + 16 + 2 \times 2^{16.5} \approx 2^{18.28} < 2^{19}$ queries.
**Success probability.** From the above process, we observe that, when at least a specific message pair $(M^u, M^v)$, satisfying the block-collisions property conditions, is found in step 3, our attack succeeds in the following cases. If the SP-MAC is based on $\boldsymbol{H}$, eight pairs $(M^u \parallel P^{x'}, M^v \parallel P^{x'})$ such that for $\forall j \in [0, 7]$ $(\overline{T_j^u} = \overline{T_j^v})$ is searched in step 4. Otherwise, if it is SPMAC-RF. The detailed computation of the probability is as follows.

For a random messages pair $(M^u, M^v)$, the output difference satisfies the block-collisions property conditions with probability $2^{-32}$.

According to the birthday paradox and Taylor series expansion, no matter what kind of oracle MAC is, among the $2^{17}$ messages, we can find a message pair $(M^u, M^v)$ satisfying the block-collisions property conditions with

$$p \approx 1 - (1 - \frac{1}{2^{32}})^{C_{2^{17}}^2} \approx 1 - e^{-2} \approx 0.865.$$

For SPMAC-$\boldsymbol{H}$, the block-collisions property, that is $\forall j \in [0,7]$ $(\overline{T_j^u} = \overline{T_j^v})$, happens in step 4 with higher probability $2^{-16.5}$ instead of the average probability $2^{-32}$. So, when the SPMAC is based on $\boldsymbol{H}$, we can find the block-collisions property among $2 \times 2^{16.5} = 2^{17.5}$ adaptive chosen messages in Step 4 with probability

$$p_1 = 1 - (1 - \frac{1}{2^{16.5}})^{2^{17.5}} \approx 1 - e^{-2} \approx 0.865.$$

Otherwise, a collision occurs for SPMAC-RF with a low probability

$$p_2 = 1 - (1 - \frac{1}{2^{32}})^{2^{17.5}} \approx 1 - e^{-2^{14.5}} \approx 0.$$

Hence, the success rate of this attack is

$$p \times [\frac{p_1}{2} + (1 - \frac{1-p_2}{2})] \approx 0.865 \times (0.865 \times \frac{1}{2} + \frac{1}{2}) \approx 0.807.$$

The success rate can be improved by repeating the attack several times.

**Remark:** Using our method in paper [14], we can easily obtain forgery attack on the SPMAC-$\boldsymbol{H}$ with the same complexity and success rate.

### 3.3 An Adaptive Chosen Message Attack HMAC-$\boldsymbol{H}$

The above attack cannot be applied to HMAC-$\boldsymbol{H}$ directly due to the fact that the block-collisions property of $H_{in}$ is concealed by the outer level hashing $H_{out}$. However, we can discard all other block-collisions by some concrete detective techniques, and save the block-collisions satisfying the block-collisions property.

Suppose that we get a 32-bit block-collision of HMAC-$\boldsymbol{H}$ which has the form $(M^u \parallel P, M^v \parallel P)$, denote the HMAC-$\boldsymbol{H}$ value of $(M^u \parallel P, M^v \parallel P)$ as $(T^u, T^v)$, and the $j^{th}(0 \le j \le 7)$ 32-bit block-collision as $(T_j^u, T_j^v)$. For simplicity, denote $\overline{H}_{in}(M^u)$ as $TI^u$, and $\overline{H}_{in}(M^v)$ as $TI^v$. Let $\Delta TI_j = TI_j^u \oplus TI_j^v$ be a 32-bit block difference. The main of our attack is to distinguish the block-collisions satisfying the block-collisions property from other block-collisions according to the relation of $\overline{H}_{in}(M^u)$ and $\overline{H}_{in}(M^v)$:

- Internal block-collision: If $\Delta TI_j = 0$, $(M^u \parallel P, M^v \parallel P)$ have an internal block-collision.
- External block-collision: If $\Delta TI_j \neq 0$, $(M^u \parallel P, M^v \parallel P)$ have an external block-collision. Furthermore, when $(TI^u, P)$ and $(TI^v, P)$ satisfy the block-collisions property condition, and $(M^u, M^v)$ have the block-collisions property. Otherwise, the block-collision is non block-collisions property.

The adversary performs as follows:

1. Generate $2^{17}$ $t$-block messages $M^x$ randomly, and append a fixed block message $P$ (taking padding into consideration) to each $M^x$. Query all the messages $M^x \parallel P$ to get their MAC values $T^x$.

2. Find all 32-bit block collided messages $(M^u \parallel P, M^v \parallel P)$ satisfying $T_j^u = T_j^v$. Note that on average, there are $2^5$ internal block-collisions, $2^6$ external block-collisions, and 4 of them satisfy the block-collisions property.
3. For all $(M^u \parallel P, M^v \parallel P)$ collected in step 2 , we append one 512-block message $P' \neq P$ to $M^u$ and $M^v$, query MACs $(\overline{T^u}, \overline{T^v})$ of $(M^u \parallel P', M^v \parallel P')$, and check if the same $j^{th}$ 32-bit block still collide $(\overline{T_j^u}, \overline{T_j^v})$. This way, the internal block-collisions can be detected. Later, we only need to distinguish external collisions satisfying the block-collisions property from the other external block-collisions.
4. For the remaining $(M^u \parallel P, M^v \parallel P)$, append $2^{16.5}$ random $P^{x'} \neq P$ to $M^u$ and $M^v$, respectively, ask for the MAC values $(\overline{T_j^u}, \overline{T_j^v})$ of $(M^u \parallel P^{x'}$, $M^v \parallel P^{x'})$, and check whether satisfies the block-collisions property:
   (a) For $\forall j \in [0, 7]$, at least one block-collision $T_j^{u'} = T_j^{v'}$ is found, we conclude that the original $(M^u, M^v)$ satisfies the block-collisions property, and output is HMAC-$\boldsymbol{H}$.
   (b) Otherwise, it is a HMAC-RF.

***Complexity evaluation*** There are at most $2^{34}$ pairs produced by $2^{17}$ messages, so the expected number of internal 32-bit block-collisions is $2^{34-32} \times 8 = 2^5$. Similarly, the expectation of external block-collisions is $2^5 + 2^5 = 2^6$ where $2^5$ block-collisions occur after $H_{in}$ and other $2^5$ block-collisions occur after $H_{out}$. For two messages $M^u$ and $M^v$, the output difference $H_{k_2}(M^u \parallel P) \oplus H_{k_2}(M^v \parallel P)$ satisfies the block-collisions property condition with probability $2^{-32}$. Consequently, there are $2^{34-32} = 4$ pairs satisfying the block-collisions property conditions. In step 1, the data complexity is $2^{17}$. We keep a table of $2^{17}$ entries in step 2, finding $2^5 + 2^6 \approx 2^{6.57}$ block-collisions needs about $2^{17}$ table lookups. Step 3 needs about $2 \times (2^5 + 2^6) \approx 2^{7.57}$ MAC computations. In step 4 , both the data and time complexity are about $2^6 \times 2^{16.5} = 2^{22.5} < 2^{23}$.

Therefore, the total time complexity is about $2^{23}$ MAC computations and $2^{17}$ table lookups, and data complexity is about $2^{17}$ chosen messages.
**Success rate**: As analyzed in section 3.1, we divide the success rate into two parts:

– If the MAC is HMAC-$\boldsymbol{H}$, the attack succeeds when the block-collisions property is found among $2^{6.53}$ block-collisions in step 4.
  The probability that there exists the block-collisions property conditions among $2^{6.57}$ block-collisions is:
$$1 - (1 - \frac{1}{2^{32} + 2^{32}})^{2^{34}} \approx 1 - e^{-2} \approx 0.865.$$
  The probability that the block-collisions property can be detected in step 4 is about
$$1 - (1 - \frac{1}{2^{16.5}})^{2^{17.5}} \approx 1 - e^{-2} \approx 0.865.$$
  Thus, if the MAC is HMAC-$\boldsymbol{H}$, the attack can the block-collisions property with probability $0.865 \times 0.865 = 0.748$.

– If the MAC is HMAC-RF, the attack succeeds when no block-collisions property is detected. The success probability is about

$$((1-\frac{1}{2^{32}})^{2^{17.5}})^{2^6}=((1-\frac{1}{2^{32}})^{2^{32}})^{2^{-8.5}}\approx 0.997.$$

Therefore, the success rate of the whole attack is about

$$\frac{1}{2}\times 0.748+\frac{1}{2}\times 0.997=0.873.$$

### 3.4 Chosen Message Attack on HMAC-*H*

Here, we relax the adaptive chosen message attack to a chosen message attack. The data complexity will increase up to $2^{33}$. The chosen message distinguishing attack on HMAC-***H*** is described as follows:

1. Select a set of $2^{16.5}$ $t$-block messages $M^x$ at random. Append one block message $P$ to all the $M^x$, and form a structure of $2^{16.5}$ messages $M^x \parallel P$. Choose $2^{16.5}$ different one block messages $P$ to produce $2^{16.5}$ structures. Make $2^{33}$ MAC queries for all of the structures.
2. For each structure, fulfill the birthday attack[13] to find all $(M^u \parallel P, M^v \parallel P)$ which have 32-bit block-collisions $T_j^u = T_j^v (0 \le j \le 7)$.
3. For each $(M^u \parallel P, M^v \parallel P)$ found in step 2, we determine the type of the 32-bit block-collision.
   (a) Check whether all the pairs $(M^u \parallel P^{x'}, M^v \parallel P^{x'})$ in other structures have 32-bit block-collisions in the same position. If they have, then $(M^u \parallel P, M^v \parallel P)$ has a 32-bit internal block-collision. We discard these pairs.
   (b) For $\forall j \in [0,7]$, Check whether all the pairs $(M^u \parallel P, M^v \parallel P)$ in other structures have at least one block-collision $(\overline{T_j^u}, \overline{T_j^v})$ of $(M^u \parallel P^{x'}, M^v \parallel P^{x'})$. If so, we conclude that $(M^u \parallel P, M^v \parallel P)$ satisfy the block-collisions property conditions, and these eight block-collisions the block-collisions property
4. -If we can find one pair $(M^u \parallel P, M^v \parallel P)$ satisfy the block-collisions property conditions, we conclude that the MAC is HMAC-***H***.
   -or else, it is HMAC-RF.

***Complexity and Success rate.*** It is clear that the attack needs about $2^{16.5}\times 2^{16.5}=2^{33}$ chosen messages. For each structure, the expectation is 8 internal 32-bit block-collisions and 16 external 32-bit block-collisions. So the total number of block-collisions in all $2^{16.5}$ structures is about $24\times 2^{16.5}\le 2^{22}$. For each block-collision, $2^{16.5}$ table lookups are needed. Therefore the time complexity is less than $2^{22}\times 2^{16.5}\approx 2^{39}$ table lookups, and the table size is $2^{33}$ entries. The computation of success rate is the same as in subsection 3.3.
**Application to NMAC-*H***: NMAC-***H*** is a generalized version of HMAC-***H*** as introduced in subsection 2.3. Since the above attack on HMAC-***H*** has no relation with the secret key, hence it can be applied to NMAC-***H*** directly.
**Remark**: Using our method in paper [14], we can easily obtain forgery attack on the HMAC/NMAC-***H*** with the same complexity and success rate.

## 4 A Key Recovery Attack on the SPMAC-$\boldsymbol{H}$

It should be noted that our distinguishing attack on HMAC/NMAC-$\boldsymbol{H}$ can not be extended to recover the inner key $k_2$, we can not derive $k_2$ by the backward decrypting computations because of lack of the value $H_{k_2}(M)$. But using the distinguishing attack in Section 3.2, we can also easily present a key recovery attack on the SPMAC-$\boldsymbol{H}$.

Once we confirm the SPMAC-$\boldsymbol{H}$ in Step 4, we can recovery its key by $(t+1)\times 8$ times decrypting algorithms. This process can be divided into two phases.

- **Phase 1:** Find all pairs $(M^u, M^v)$ satisfying the block-collisions property, i.e., $\forall j \in [0, 7]$, at least one block-collision $(\overline{T_j^u}, \overline{T_j^v})$ exist.
  Note that by the techniques described in Section 3.2, it's easy to find a pair $(M^u, M^v)$ which appends some $P^{x'}$ can acquire eight block-collision pairs $(\overline{T_0^a}, \overline{T_0^b})$, $(\overline{T_1^c}, \overline{T_1^d})$,…, $(\overline{T_7^o}, \overline{T_7^p})$ with $2^{17}$ chosen messages and $2^{19}$ MAC queries.
- **Phase 2:** Let $k = k_1||\ldots||k_7$ be the secret key of SPMAC-$\boldsymbol{H}$. For $\forall j \in [0, 7]$, decrypt one 32-bit block $\overline{T_j^x}$, found in Phase 1, using $P_j^{x'}$ as the key , to obtain the chaining variable $\overline{H_{(t-1)j}^x}$. In turn, decrypting $\overline{H_{(t)j}^x}, \ldots, \overline{H_{(1)j}^x}$ by corresponding keys $M_{(t-1)j}^u, \ldots, M_{(0)j}^u$, to retrieve each of the 32-bit secret key $k_j = E^{-1}_{M^u_{(0)j}}(\overline{H_{(1)j}^x})$. Then, verify the secret key $k$ by questioning the MAC value of corresponding another one. We recover the secret key $k$ as follows:
  1. Decrypt the block cipher $E^{-1}_{P_0^{a'}}(\overline{T_0^a})$, and obtain the corresponding chaining variable $\overline{H_{(t)0}^a}$. In turn, we can get other chaining variable values, $\overline{H_{(t-1)0}^a}, \ldots, \overline{H_{(1)0}^a}$ by $t$ times decrypting algorithms, and recover the first 32-bit block-key $k_0$ in the end, which is as follows:
  $$k_0 = E^{-1}_{M^u_{(0)0}}(H^a_{(1)0}) = E^{-1}_{M^u_{(0)0}}(E^{-1}_{M^u_{(1)0}}(\ldots(E^{-1}_{P_0^{a'}}(\overline{T_0^a}))\ldots)).$$
  2. Similarly, we can retrieve other seven 32-bit keys $k_1$,…,$k_7$ by decrypting the block cipher $E^{-1}_{P_1^{c'}}(\overline{T_1^c})$, $E^{-1}_{P_2^{e'}}(\overline{T_2^e})$, $E^{-1}_{P_3^{g'}}(\overline{T_3^g})$, $E^{-1}_{P_4^{i'}}(\overline{T_5^i})$, $E^{-1}_{P_5^{k'}}(\overline{T_5^k})$, $E^{-1}_{P_6^{m'}}(\overline{T_6^m})$, and $E^{-1}_{P_7^{o'}}(\overline{T_7^o})$,respectively.
  3. Ask each MAC value of $M^v \parallel P^{a'}$,$M^v \parallel P^{c'}$, $M^v \parallel P^{e'}$, $M^v \parallel P^{g'}$, $M^v \parallel P^{i'}$, $M^v \parallel P^{k'}$, $M^v \parallel P^{m'}$ and $M^v \parallel P^{o'}$, under the obtained $k$. Test whether eight block-collisions occur simultaneity:
     - If eight block-collisions $(\overline{T_0^a}, \overline{\overline{T_0^b}})$, $(\overline{T_1^c}, \overline{\overline{T_1^d}})$,…, $(\overline{T_7^o}, \overline{\overline{T_7^p}})$ still occur, which means that the obtained secret key $k$ is right.
     - Otherwise, go Phase 1 to choose another pair $(M^{u'}, M^{v'})$ satisfying the block-collisions property.

  In all, recovery secret key $k$ requires $2^{17}$ chosen messages, $2^{19}$ MAC queries, and $(t + 1) \times 8$ times decrypting algorithms.

## 5 Conclusions

In this paper, we utilize new techniques to construct distinguishing attacks on SPMAC/HMAC/NMAC-$\boldsymbol{H}$, which separate the output of the HMAC/NMAC into $j$ block, and adopt segmenting techniques to detect $j$ block-collisions, named the block-collisions property in this paper, with specific differential paths or not. The complexities of distinguishing attack on SPMAC-$\boldsymbol{H}$ are $2^{17}$ chosen messages and $2^{19}$ queries with 0.807 success probability. We give two Distinguishing attacks on HMAC/NMAC-$\boldsymbol{H}$, one needs $2^{17}$ chosen messages and $2^{23}$ MAC computations under adaptive chosen message attack with a success rate of 0.873, another requires $2^{17}$ chosen messages and $2^{39}$ MAC computations with the same success rate. Furthermore, the distinguishing attack on SPHMAC-$\boldsymbol{H}$ can be applied to recover its secret key $k$. Besides $2^{17}$ chosen messages and $2^{19}$ queries in Section 3.2, the key recovery attack additional demands $(t+1) \times 8$ times decrypting algorithms.

## References


1. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 1-15. Springer, Heidelberg (1996)
2. Bellare, M., Kilian, J., Rogaway, P.: The Security of the Cipher Block Chaining Message Authentication Code. In: Desmedt, Y. G. (ed.) CRYPTO'94. LNCS 839, pp. 341-358. Springer, Heidelberg (1994)
3. Bellare, M.: New Proofs for NMAC and HMAC: Security without collisionresistance. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 602-619. Springer, Heidelberg (2006)
4. Contini, S., Yin, Y.L.: Forgery and partial key-recovery attacks on HMAC and NMAC using hash collisions. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 37-53. Springer, Heidelberg (2006)
5. Keting Jia, Xiaoyun Wang, Zheng Yuan, Guangwu Xu. Distinguishing and Second-Preimage Attack on CBC-like MACs. Advances in Cryptology and Network Security-CANS 2009, LNCS 5888, Springer-Verlag, pp.349-361(2009).
6. Kim, J., Biryukov, A., Preneel, B., Hong, S.: On the Security of HMAC and NMAC Based on HAVAL, MD4, MD5, SHA-0, and SHA-1. In: De Prisco et al.(eds.) SCN 2006. LNCS, vol. 4116, pp. 242-256. Springer, Heidelberg (2006)
7. Yong Liu, Yu Chen, Zhong Chen, Lin Yang. A New Dedicated Compression Function Framework. In: Huanguo.Z et al.(eds.) ChinaCrypt'2008. pp. 214-219, 2008.
8. National Institute of Standards and Technology: Cryptographic Hash Algorithm Competition (November 2007), http://csrc.nist.gov/groups/ST/hash/sha-3/index.html
9. Preneel, B., van Oorschot, P.: MDx-MAC and Building Fast MACs from Hash Functions. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 1-14. Springer, Heidelberg (1995)
10. Xiaoyun Wang, Wei Wang, Keting Jia, Meiqin Wang. New Distinguishing Attack on MAC using Secret-Prefix Method. In: O. Dunkelman (Ed.): FSE 2009, LNCS 5665, pp. 363-374, 2009.

11. Xiaoyun Wang, Hongbo Yu, Wei Wang, Haiha Zhang, Tao Zhan. Cryptanalysis on HMAC/NMAC-MD5 and MD5-MAC. In: Joux, A. (ed.): EUROCRYPT 2009, LNCS 5479, Springer, Heidelberg, pp. 121-133, 2009.
12. Zheng Yuan, Wei Wang, Keting Jia, Guanfwu Xu, Xiaoyun Wang. New Birthday Attacks on Some MACs Based on Block Ciphers. In: Shai Halevi: Advances in Cryptology-Crypto 2009, LNCS 5677, Springer-Verlag, Berlin Heidelberg New York, pp. 209-230, 2009.
13. Yuval, G.: How to swindle rabin. Cryptologia 3, 187-190 (1979)