Distinguishing and Forgery Attacks on ALRED and Its AES-based Instance ALPHA-MAC*

Zheng Yuan[1,2], Keting Jia[3], Wei Wang[3], and Xiaoyun Wang**[2,3]

[1] Beijing Electronic Science and Technology Institute, Beijing 100070,China
[2] Center for Advanced Study, Tsinghua University, Beijing 100084, China
xiaoyunwang@mail.tsinghua.edu.cn
[3] Key Laboratory of Cryptographic Technology and Information Security, Ministry of Education, Shandong University, Jinan 250100, China

Abstract. In this paper, we present new distinguishers of the MAC construction ALRED and its specific instance ALPHA-MAC based on AES, which is proposed by Daemen and Rijmen in 2005. For the ALRED construction, we describe a general distinguishing attack which leads to a forgery attack directly. The complexity is $2^{64.5}$ chosen messages and $2^{64.5}$ queries with success probability 0.63. We also use a two-round collision differential path for ALPHA-MAC, to construct a new distinguisher with about $2^{65.5}$ queries. The most important is that the new distinguisher can be used to recover the internal state, which is an equivalent secret subkey, and leads to a second preimage attack. Moreover, the distinguisher on ALRED construction is also applicable to the MACs based on CBC and CFB encryption mode.
Keywords: Distinguishing attack, Forgery attack, ALRED construction, ALPHA-MAC, AES

1 Introduction

Message Authentication Code (MAC) is a fixed length information used to ensure data integrity and authenticity, and is widely used in Internet community such as IPsec, SNMP, SSL, etc. MAC takes a secret key and a message of arbitrary length as inputs, and outputs a short digest. Many research groups have presented various approaches to construct MAC functions, for example, MAA [7], UMAC [3], OMAC [9], TMAC [12], XCBC [4], RMAC [10], NMAC [1], and HMAC [13], etc.

The MAC construction ALRED was introduced by Daemen and Rijmen in FSE 2005, and an efficient instance based on AES is called ALPHA-MAC [6]. The ALRED construction is an iterative MAC function using components of block ciphers. The secret key, which is used as the key of the block cipher, is applied in the initialization and the final transformation, respectively. The internal state

* This work is supported by the National Outstanding Young Scientist (No.60525201), China Postdoctoral Science Foundation Funded Project (No. 20080430423) and National 973 Program of China (No.2007CB807902)
** To whom correspondence should be addressed.

is changed by consecutive injections of message blocks. The ALPHA-MAC is the ALRED construction instantiated with AES [5]. Since the AES algorithm has been widely used in the real world, the ALPHA-MAC can be easily implemented, which is a factor 2.5 more efficient than the popular CBC-MAC with AES.

The designers [6] proved that the ALRED construction is as strong as the underlying block cipher with respect to key recovery and any forgery attack not involving internal collisions. Furthermore, for ALPHA-MAC, they showed that any colliding messages of the same size have to be at least 5 blocks long. Recently, Huang et al. [8] exploited the algebraic properties of the AES, constructed internal collisions, and found second preimages for ALPHA-MAC, on the assumption that a key or an internal state is known. Biryukov et al. [2] proposed a side-channel collision attack on ALPHA-MAC recovering its internal state, and mounted a selective forgery attack.

The main contribution of this paper is to present interesting distinguishing attacks on the ALRED construction and ALPHA-MAC, which can lead to forgery attack directly. More important, the distinguishing attack on ALPHA-MAC can be applicable to recover the internal state, which results in a second preimage attack.

There are two kinds of distinguishing attacks on MACs. Preneel and van Oorschot [14] introduced a general distinguishing attack to identify iterated MACs from a random function. Using the birthday paradox, they detect the internal collision by appending the same one-block message. Another kind is to distinguish the cryptography primitive embedded in the MAC construction from a random function [11]. Recently, new techniques to identify the underlying hash functions of MACs are presented in [15,16]. Wang et. al. [16] presented distinguishing attacks on HMAC/NMAC-MD5 and MD5-MAC, and recovered partial subkey of the MD5-MAC. Their distinguisher makes use of inner near-collisions, which can leak more information than inner collisions.

Our distinguishing attacks are enlightened by Wang et al.'s work, which can detect the inner near-collision with some specific differences. First, we describe a distinguishing attack on the ALRED construction. By the birthday attack [17], there exists a collision differential path with some specific differences in the state, which is an inner near-collision, and can be recognized with probability 1 by appending another message pair with the same difference. Second, we present a new distinguisher for ALPHA-MAC based on a two-round collision differential path of AES. Furthermore, combining the structural features of the ALPHA-MAC with the algebraic properties of AES, the differential path can be used to recover the internal state, which is an equivalent subkey. With the recovered subkey, we can obtain the second preimage of the MAC for any given message M and its MAC value. The complexity of all the attacks is up to $2^{65.5}$ choose messages and $2^{65.5}$ queries with success rate 0.63. Moreover, the distinguishing attack on the ALRED construction is also applicable to the MACs based on CBC and CFB encryption mode.

The paper is organized as follows. In section 2, we list the notations used in this paper and give a short description of the ALRED construction and ALPHA-

MAC. Section 3 shows our new distinguishing attack and forgery attack on the ALRED construction. New distinguishing and recovery attack on AES-based ALPHA-MAC are introduced in the section 4. Finally, We conclude the paper in Section 5.

2 Backgrounds and Notations

In this section, we define the notations, and give a brief description of the ALRED construction and ALPHA-MAC.

2.1 Notation

f	:	the iteration function
x_i	:	the message word
y_i	:	the state after the i-th iteration
k	:	the secret key
C	:	the output of MAC taking secret key K and message M as input
ΔA	:	the XOR difference of A and A'
n	:	the length of the state
l_w	:	the length of the message word
l_m	:	the length of the MAC output
$M\|N$	:	the concatenation of M and N
$S(a)$	:	the output of S-box with a as input

2.2 The ALRED Construction

The MAC construction ALRED [6] bases on an iterated block cipher. The length of the secret key equals to that of the underlying block cipher, and the message length is a multiple of l_w bits.

Denote the i-th message word as x_i, and the state after the i-th iteration as y_i. For message $M = (x_1, x_2, \cdots, x_t)$, the construction is as follows.

1. Apply the block cipher to the state of all-zero block, i. e.,
$$y_0 = \mathrm{Enc}_k(0).$$
2. Perform the following iteration f for each message word: (a) *Injection layout*: Map the bits of the message word to an *injection input* that has the same dimensions as a sequence of r round keys of the block cipher. (b) Apply a sequence of r block cipher round functions to the state, replacing the round keys with the injection input.
$$y_i = f(y_{i-1}, x_i), i = 1, 2, \cdots, t.$$
3. Apply the block cipher to the state y_t, and truncate the first l_m bits of the state as the output. The final output C is
$$C = Trunc(\mathrm{Enc}_k(y_t)).$$

2.3 A Brief Description of ALPHA-MAC

ALPHA-MAC [6] is a specific instance of the ALRED construction with AES as the underlying block cipher, where $l_w = 32$ and $r = 1$. Similar with AES, the ALPHA-MAC supports key length of 128, 192 and 256 bits. The ALPHA-MAC function is depicted in Figure 1.

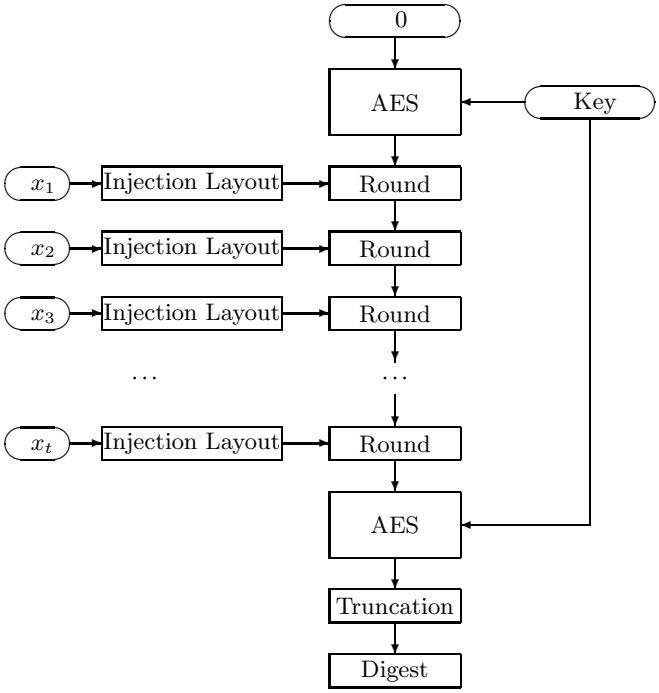

Fig. 1. The Construction of ALPHA-MAC

The message padding method appends a single 1 followed by the minimum number of 0 bits such that the length of the result is a multiple of 32. For AES-128, the injection layout places the 4 bytes of each message word $x_i = (x_{i,0}, x_{i,1}, x_{i,2}, x_{i,3})$ into a 4×4 array with the form:

$$\begin{pmatrix} x_{i,0} & 0 & x_{i,1} & 0 \\ 0 & 0 & 0 & 0 \\ x_{i,2} & 0 & x_{i,3} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

which acts as the corresponding 128-bit round key. The ALPHA-MAC round function consists of the four basic transformations of AES in the following order:

– AddRoundKey (AK): add the injection input to the state by XOR operation.
– SubBytes (SB): operate a non-linear byte substitution on each byte of the state independently using an 8×8 S-box.
– ShiftRows (SR): cyclically shift left the bytes in the last three rows of the state with different number of bytes, 1 for the second, 2 for the third and 3 for the fourth row.
– MixColumns (MC): multiply each column of the state with a matrix.

In this paper, we assume there is no truncation on the final output, i. e., $l_m = 128$.

2.4 The Outline of the Related Works

We recall the general distinguishing-R attack on all iterated MACs proposed by Preneel and van Oorschot [14], and the distinguishing-H attack on HMAC/NMAC-MD5 and MD5-MAC introduced by Wang et al. [16]. These attacks motivate us to explore new distinguish attacks on MACs based on block cipher.

Preneel et al. proposed a general forgery attack on MAC by birthday paradox, which is applicable to all deterministic iterated MACs, including MAA and CBC-MAC. They detected all the colliding pairs among $2^{(n+1)/2}$ known text-MAC pairs by birthday attack [17], where n is the bit length of the chaining variable. For each searched collisions, i. e., $MAC(k, M) = MAC(k, M')$, they appended one-block message N to identify whether it is an internal collision, according to the equation $MAC(k, M||N) = MAC(k, M'||N)$ holds or not. Once an internal collision is recognized, they can query the MAC with $M\|N'$, then they carried out a forgery, i. e., a new message $M'\|N'$ with a valid MAC. But the method can't distinguish the cryptographic primitives embedded in the MAC.

Wang et al. [16] introduced another interesting idea which can distinguish HMAC/NMAC-MD5 without the related-key setting and implement partial key recovery attack on MD5-MAC. The main idea of the distinguishing attack is: Firstly, they collect enough two-block message pairs $(M\|N, M'\|N)$ to guarantee the appearance of an expected internal near-collision in the first iteration. Then detect such a near-collision by changing the second block with another message N'. Once the expected inner near-collision is identified, the MAC is based on MD5.

3 Distinguishing and Forgery Attack on MAC Construction ALRED

In this section, we present distinguishing and forgery attack on ALRED construction. Enlightened by Wang et al.'s idea, we can get proper output difference as an inner near-collision by the birthday paradox. When the MAC construction is ALRED rather than a random function, this kind of inner near-collision can be detected with probability 1 by substituting the last different message pair with another message pair with the same difference. Based on this detected inner near-collision, the forgery attack can be constructed immediately.

3.1 Distinguishing Attack on ALRED Construction

The iteration part of ALRED construction is based on the round function of block cipher, where the round key is substituted for the injection input. The core of our distinguisher is to detect Δy_{j-1}, which is the output difference of $(j-1)$-th iteration. According to the operation between the injection input and the state involved in the iteration f, the message word difference Δx_j may extinguish Δy_{j-1}, and lead to a collision at the final output. The form of the difference depends on the operation between the injection input and the state involved in the round function, e. g., for ALRED based on IDEA or RC6, the operation is modular addition, while for others, it is XOR. The rest of this paper takes the XOR operation.

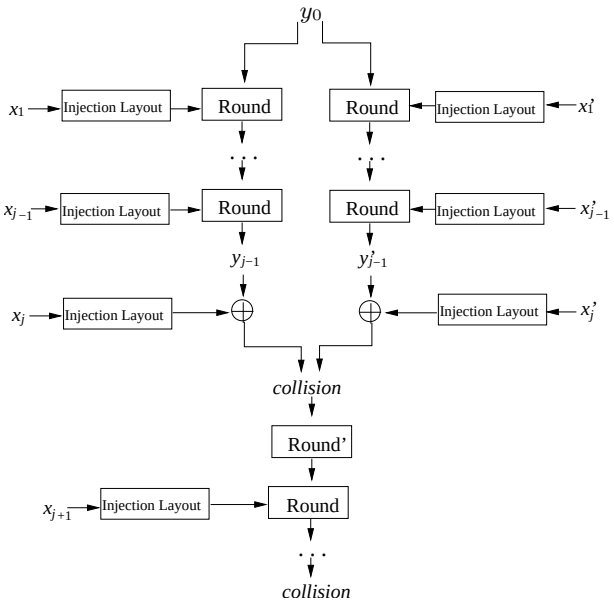

Fig. 2. The Distinguisher with XOR Operation

As shown in Fig. 2, there is an inner near-collision after round $j-1$. When $\Delta x_j = \Delta y_{j-1}$, there will be $x_j \oplus y_{j-1} = x'_j \oplus y'_{j-1}$, which is an internal collision, and can be propagated to the output. If the construction is ALRED with XOR operation, replace the (x_j, x'_j) with different $(\overline{x_j}, \overline{x'_j})$, where $\Delta\overline{x_j} = \Delta x_j$, a collision still happens. According to this property, the distinguisher is constructed as follows:

1. Randomly choose a structure $T = \{M^i | M^i = (x_1^i, x_2^i, \cdots, x_t^i)\}$ composed of $2^{(n+1)/2}$ different messages, and query the corresponding MAC value C^i.

2. By birthday paradox, a collision $C^a = C^b$ can be obtained. The corresponding message words are M^a and M^b.
3. Suppose x_j is the first unequal word in M^a and M^b counting backwards, i. e. $M^a = (x_1^a, \cdots, x_j^a, x_{j+1}, \cdots, x_t)\}$, $M^b = (x_1^b, \cdots, x_j^b, , x_{j+1}, \cdots, x_t)\}$. We replace (x_j^a, x_j^b) with another $(\overline{x_j^a}, \overline{x_j^b})$, where $\overline{x_j^a} \oplus \overline{x_j^b} = x_j^a \oplus x_j^b$. Query the MACs with $(\overline{M^a}, \overline{M^b})$, where $\overline{M^a}$ $=\{x_1^a, \cdots, x_{j-1}^a, \overline{x_j^a}\}$ and $\overline{M^b}$ $=\{x_1^b, \cdots,$ $x_{j-1}^b, \overline{x_j^b}\}$.
   - If $\overline{C^a} = \overline{C^b}$, we conclude that the MAC is ALRED construction.
   - Else, it is a random function.

Note that t should be large enough to guarantee there is an inner near-collision at round $j-1, j \leq t$.

This attack requires about $2^{(n+1)/2}$ chosen messages and works with a probability of 0.63 by the birthday paradox. If we double the number of chosen text-MAC pairs, the success rate can be increased to 0.98.

Remark. With regard to MACs based on CBC encryption mode for block ciphers, e.g. CBC-MAC, OMAC, TMAC, etc., the iteration of MAC is defined as follows:

$$H_i = f(H_{i-1}, x_i) = E_k(H_{i-1} \oplus x_i).$$

The above attack can be applied similarly. Besides, the method also works for the MACs based on CFB mode, i. e.,

$$H_i = f(H_{i-1}, x_i) = E_k(H_{i-1}) \oplus x_i.$$

3.2 Forgery Attack on ALRED Construction

Once the inner near-collision is identified, we replace message words with the same difference, and achieve a new collision pair. Hence, the forgery attack is easily realized with the same complexity and success rate as the distinguishing attack. The details are as follows:

Suppose (M^a, M^b) is the colliding pair detected in the above distinguishing attack. We query the MAC oracle about $\widetilde{M^a}$, where $\widetilde{M^a} = (x_1^a, \cdots, x_{j-1}^a, \widetilde{x_j^a}, s)$, and s is an arbitrary message string. Then we construct the forgery of $\widetilde{M^b} = (x_1^b, \cdots, x_{j-1}^b, \widetilde{x_j^a} \oplus \Delta x_j, s)$ with one query.

4 Recovery the Equivalent Subkey of ALPHA-MAC

The above distinguisher is applicable to distinguish the ALPHA-MAC from a random function, however, we introduce a new distinguisher in this section, where the expected collision implies an inner near-collision with some specific differences. With this distinguisher, we can recover an internal state, which results in the derivation of the equivalent subkey, i. e., the state y_0 (See Fig. 2).

4.1 Some Useful Properties of AES

This section presents a two-round collision differential path of AES, and summarizes some useful properties based on it. The two-round differential path will be used to recover the inner state in Section 4.3.

For $i = 1, \cdots, t$, denote

$$\begin{pmatrix} y_{i-1,0} & y_{i-1,1} & y_{i-1,2} & y_{i-1,3} \\ y_{i-1,4} & y_{i-1,5} & y_{i-1,6} & y_{i-1,7} \\ y_{i-1,8} & y_{i-1,9} & y_{i-1,10} & y_{i-1,11} \\ y_{i-1,12} & y_{i-1,13} & y_{i-1,14} & y_{i-1,15} \end{pmatrix} \oplus \begin{pmatrix} x_{i,0} & 0 & x_{i,1} & 0 \\ 0 & 0 & 0 & 0 \\ x_{i,2} & 0 & x_{i,3} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\text{SB}} \begin{pmatrix} z_{i,0} & z_{i,1} & z_{i,2} & z_{i,3} \\ z_{i,4} & z_{i,5} & z_{i,6} & z_{i,7} \\ z_{i,8} & z_{i,9} & z_{i,10} & z_{i,11} \\ z_{i,12} & z_{i,13} & z_{i,14} & z_{i,15} \end{pmatrix},$$

where y_{i-1} is the output of round $i-1$, and ($x_{i,0}$,0,$x_{i,1}$,0,0,0,0,0,$x_{i,2}$,0,$x_{i,3}$,0,0,0,0,0) is the injection input to round i which acts as the round key. Suppose (y_{t-2}, x_{t-1}, x_t) and $(y'_{t-2}, x'_{t-1}, x'_t)$ follow the two-round collision differential path as depicted in Fig. 3.

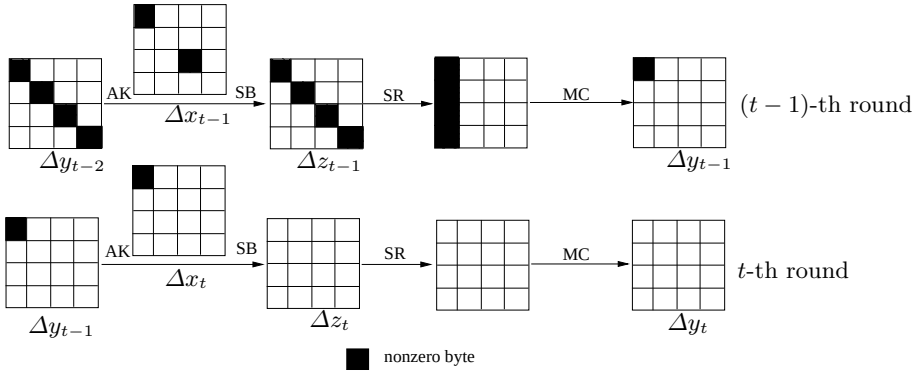

Fig. 3. Two-Round Collision Differential Path

From the differential path, we can see that there is only one nonzero byte in Δy_{t-1}, which equals to $\Delta x_{t,0}$. Because MC is a linear transformation, and SR has no impact on the value of difference, we can compute the output differences of four S-boxes in the $(t-1)$-th round:

$$(\Delta z_{t-1,0}, \Delta z_{t-1,5}, \Delta z_{t-1,10}, \Delta z_{t-1,15})^T = MC^{-1}(\Delta x_{t,0}, 0, 0, 0)^T. \qquad (1)$$

Since the branch number of MC transformation in AES is 5 [5], there are four nonzero bytes in Δz_{t-1}. It is noted that given (y_{t-2}, y'_{t-2}) and (x_t, x'_t), there will be a collision if and only if $(\Delta z_{t-1,0}, \Delta z_{t-1,5}, \Delta z_{t-1,10}, \Delta z_{t-1,15})$ satisfies equation (1), and the difference of other bytes in Δz_{t-1} are zero. Thus, we have the following property:

Property 1. Provided that (y_{t-2}, x_{t-1}, x_t) and $(y'_{t-2}, x'_{t-1}, x'_t)$ following the two-round collision differential path, randomly choose 2^8 different pairs of $(\overline{x_{t-1,0}}, \overline{x'_{t-1,0}})$ to replace $(x_{t-1,0}, x'_{t-1,0})$, respectively, and compute 2^8 corresponding pairs of $(\overline{y_t}, \overline{y'_t})$, there exists one collision of $(\overline{y_t}, \overline{y'_t})$ on average.

Proof. Since only $(x_{t-1,0}, x'_{t-1,0})$ changes, all bytes in Δz_{t-1} remain the same except $\Delta z_{t-1,0}$, where $\Delta z_{t-1,0} = S(y_{t-2,0}, x_{t-1,0}) \oplus S(y'_{t-2,0}, x'_{t-1,0})$. Thus, in order to obtain a new collision, $(\overline{x_{t-1,0}}, \overline{x'_{t-1,0}})$ must satisfy

$$S(y_{t-2,0} \oplus \overline{x_{t-1,0}}) \oplus S(y'_{t-2,0} \oplus \overline{x'_{t-1,0}}) = \Delta z_{t-1,0}.$$

From the distribution table of S-box in AES, we can observed that, there are 2^7 pairs corresponding to each output difference on average. Hence, one randomly chosen pair leads to the expected output difference $\Delta z_{t-1,0}$ with probability $2^7/2^{15} = 2^{-8}$, there is excepted to have a collision among the 2^8 $(\overline{y_t}, \overline{y'_t})$. □

This property is used in the new distinguishing attack to identify the ALPHA-MAC from a random function. For ALPHA-MAC, the inner state y_{t-2} is unknown, however, we can recover two bytes of y_{t-2} by the next two properties.

Property 2. Suppose (y_{t-2}, x_{t-1}, x_t) and $(y'_{t-2}, x'_{t-1}, x'_t)$ follow the two-round collision differential path, where (y_{t-2}, y'_{t-2}) are unknown, we can compute the value $(y_{t-2,0}, y'_{t-2,0})$ with about 2^{16} XOR operations and 2^{10} chosen messages.

Proof. If (y_{t-2}, x_{t-1}, x_t) and $(y'_{t-2}, x'_{t-1}, x'_t)$ follow the two-round differential path, there will be

$$S(y_{t-2,0} \oplus x_{t-1,0}) \oplus S(y'_{t-2,0} \oplus x'_{t-1,0}) = \Delta z_{t-1,0}. \tag{2}$$

According to Property 1, we get

$$S(y_{t-2,0} \oplus \overline{x_{t-1,0}}) \oplus S(y'_{t-2,0} \oplus \overline{x'_{t-1,0}}) = \Delta z_{t-1,0}. \tag{3}$$

Thus, we can guess all 2^{16} possibilities of $(y_{t-2,0}, y'_{t-2,0})$ and filter out the wrong ones by equation (2) and (3).

If there are more than one solution left, we replace the messages as described in Property 1, and get another pair $(\overline{x_{t-1,0}}, \overline{x'_{t-1,0}})$ with

$$S(y_{t-2,0} \oplus \overline{x_{t-1,0}}) \oplus S(y'_{t-2,0} \oplus \overline{x'_{t-1,0}}) = \Delta z_{t-1,0} \tag{4}$$

which leads to a collision, and achieve the right value of $(y_{t-2,0}, y'_{t-2,0})$.

The time complexity is dominated by the exhaustive search of 2^{16} possibilities of $(y_{t-2,0}, y'_{t-2,0})$, and the number of chosen messages to get equations (3) and (4) is 2^{10}. □

In a similar way, we can recover the value $(y_{t-2,10}, y'_{t-2,10})$ by replacing the $(x_{t-1,10}, x'_{t-1,10})$ with 2^9 different pairs of $(\overline{x_{t-1,10}}, \overline{x'_{t-1,10}})$.

Property 3. Suppose (y_{t-2}, x_{t-1}, x_t) and $(y'_{t-2}, x'_{t-1}, x'_t)$ follow the two-round collision differential path, where (y_{t-2}, y'_{t-2}) are unknown, we can compute the value $(y_{t-2,10}, y'_{t-2,10})$ with about 2^{16} XOR operations and 2^{10} chosen messages.

4.2 Distinguishing Attack on ALPHA-MAC

Similar with the distinguisher for ALRED construction, the new distinguisher on ALPHA-MAC is based on the identification of an inner near-collision

$$\Delta y_{t-1} = \begin{pmatrix} \Delta x_{t,0} & 0\ 0\ 0 \\ 0 & 0\ 0\ 0 \\ 0 & 0\ 0\ 0 \\ 0 & 0\ 0\ 0 \end{pmatrix},$$

just the same as the Δy_{t-1} in Fig. 3. There exists such an inner near-collision with this form by the birthday paradox, and we can detect it by the new distinguisher. Moreover, combining with the properties introduced in Section 4.1, the detected difference can lead to the recovery of the inner state, which results in the recovery of y_0. It is noted that, $y_0 = Enc_k(0)$, is equivalent to a prefix subkey.

Reference [6] claims that an extinguishing differential in ALPHA-MAC spans at least 5 message words, and given the state value y_{i-1}, the map from the sequence of four message words $(x_i, x_{i+1}, x_{i+2}, x_{i+3})$ to the state value before iteration $i+4$ is a bijection. Hence, we choose a structure composed of $2^{64.5}$ messages with t-word length, where $t \geq 6$ in order to guarantee the map form $(x_1, \cdots, x_{t-1})$ to y_{t-1} is a random function, and the expected inner near-collision can be obtained by birthday attack. It is recommended to choose $t = 9$.

We construct two structures as follows:

$$T_1 = \{M = (x_1, x_2, \cdots, x_{t-1}, x_t)\},$$
$$T_2 = \{M = (x_1, x_2, \cdots, x_{t-1}, x_t \oplus (\eta, 0, 0, 0))\},$$

where $(x_1, x_2, \cdots, x_{t-2}, x_{t-1,0}, x_{t-1,3})$ are randomly chosen, $x_{t-1,1}$, $x_{t-1,2}$, x_t and η are fixed, i. e., we choose $\Delta x_{t-1}, \Delta x_t$ as shown in Fig. 3. The distinguisher works in the following manner:

1. Query the MAC with all the $2^{65.5}$ messages in structure T_1 and T_2, and obtain the corresponding MACs.
2. According to the birthday paradox, a collision can be obtained, i. e., $C_1^a = C_2^b$, where C_1^a belongs to the MAC values of T_1, and C_2^b belongs to the MAC values of T_2. Suppose corresponding messages are M^a and M^b, denote the word differences of $(x_{t-1}^a - x_{t-1}^b)$ and $(x_t^a - x_t^b)$ as Δx_{t-1} and Δx_t, respectively. Randomly choose another different pair of $(\overline{M^a}, \overline{M^b})$, where

$$\overline{M^a} = (x_1^a, \cdots, x_{t-1}^a, \overline{x_t^a}),\ \overline{M^b} = (x_1^b, \cdots, x_{t-1}^b, \overline{x_t^b}),\ \Delta\overline{x_t} = \Delta x_t.$$

   Query the MAC with the new message pair $(\overline{M^a}, \overline{M^b})$.
   If they still collide, the MAC algorithm is ALRED-MAC, and goto step 3. Otherwise, we conclude that the MAC is a random function.
3. Replace $(x_{t-1,0}^a, x_{t-1,0}^b)$ with 2^8 different $(\overline{x_{t-1,0}^a}, \overline{x_{t-1,0}^b})$. Query the MACs on the new messages. Examine whether there is at least a new collision among them.

– If a collision appears, the ALRED construction is concluded as a ALPHA-MAC by Property 1.
– Otherwise, is based on a random function.

Complexity Analysis. The complexity is $2^{65.5}$ queries and $2^{65.5}$ chosen messages in step 1. There is only 2 queries in step 2, and 256 queries in step 3. So the total complexity is dominated by step 1, which is about $2^{65.5}$ queries and $2^{65.5}$ chosen messages.

Success Rate. The probability that there is a collision among the two structures is 0.63 according to the birthday paradox. Once the collision pair is found, the conclusion of the attack is correct according to the property of AES. Therefore, the success rate is 0.63. We can improve the success rate to 0.98 by doubling the size of each structure.

4.3 Internal State Recovery of ALPHA-MAC

In this section, we show a technique which can recover the internal state with the help of the new distinguisher presented above. When the inner near-collision is identified, we can deduce the input and output difference of two S-boxes in the $(t-1)$-th round, which allows us to recover the corresponding byte $(y_{t-2,0}, y_{t-2,10})$ according to Property 1 and 2. Combining with the relation between the round function of AES, we explore equations between the internal states, and can obtain 8 bytes of the state y_{t-3}. The rest 8 bytes can be recovered by exhaustive search.

We depict the process of the state recovery in Fig. 4, where $*$ denotes the difference that can be computed, ? stands for the unknown difference, and 0 means the same. The details of the recovery attack are as follows:

$$\Delta y_{t-3} = \begin{pmatrix} * & ? & * & ? \\ ? & * & ? & * \\ * & ? & * & ? \\ ? & * & ? & * \end{pmatrix}$$

$$\xleftarrow{AK^{-1}\ SB^{-1}} \Delta z_{t-2} = \begin{pmatrix} * & ? & * & ? \\ ? & * & ? & * \\ * & ? & * & ? \\ ? & * & ? & * \end{pmatrix} \xleftarrow{SR^{-1}\ MC^{-1}} \Delta y_{t-2} = \begin{pmatrix} * & 0 & 0 & 0 \\ 0 & ? & 0 & 0 \\ 0 & 0 & * & 0 \\ 0 & 0 & 0 & ? \end{pmatrix}$$

$$\xleftarrow{AK^{-1}\ SB^{-1}} \Delta z_{t-1} = \begin{pmatrix} * & 0 & 0 & 0 \\ 0 & * & 0 & 0 \\ 0 & 0 & * & 0 \\ 0 & 0 & 0 & * \end{pmatrix} \xleftarrow{SR^{-1}\ MC^{-1}} \Delta y_{t-1} = \begin{pmatrix} \Delta x_{t,0} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Fig. 4. Recovery of the Difference of Internal State

1. **Recovering** $(y^a_{t-2,0}, y^b_{t-2,0}, y^a_{t-2,10}, y^b_{t-2,10})$.
Once the ALPHA-MAC is recognized by the distinguisher presented in Section 4.2, we get $(y^a_{t-2}, x^a_{t-1}, x^a_t)$ and $(y^b_{t-2}, x^b_{t-1}, x^b_t)$, $(y^a_{t-2}, \overline{x^a_{t-1}}, x^a_t)$ and $(y^b_{t-2}, \overline{x^b_{t-1}}, x^b_t)$, which follow the two-round differential path (See Fig. 3), respectively. Then we achieve the value $(y^a_{t-2,0}, y^b_{t-2,0})$ and $(y^a_{t-2,10}, y^b_{t-2,10})$ according to Property 1 and 2. Hence, we get the values $\Delta y_{t-2,0}$ and $\Delta y_{t-2,10}$.
2. **Recovering** $(y^a_{t-3,0}, y^b_{t-3,0}, y^a_{t-3,2}, y^b_{t-3,2}, y^a_{t-3,8}, y^b_{t-3,8}, y^a_{t-3,10}, y^b_{t-3,10})$.
Apply MC^{-1} and SR^{-1} to the state y_{t-2}, we get the following differences:

$$(\Delta z_{t-2,0}, \Delta z_{t-2,5}, \Delta z_{t-2,10}, \Delta z_{t-2,15})^T = MC^{-1}(\Delta y_{t-2,0}, 0, \Delta y_{t-2,8}, 0)^T, \quad (5)$$

$$(\Delta z_{t-2,2}, \Delta z_{t-2,7}, \Delta z_{t-2,8}, \Delta z_{t-2,13})^T = MC^{-1}(\Delta y_{t-2,2}, 0, \Delta y_{t-2,10}, 0)^T \quad (6)$$

For the $(y^a_{t-2}, x^a_{t-1}, x^a_t)$ and $(y^b_{t-2}, x^b_{t-1}, x^b_t)$ detected in the distinguisher, we first recover $(y^a_{t-3,0}, y^b_{t-3,0})$.

(a) As $\Delta z_{t-2,0}$ is deduced from equation (5), there will be

$$S(y^a_{t-3,0} \oplus x^a_{t-2,0}) \oplus S(y^b_{t-3,0} \oplus x^b_{t-2,0}) = \Delta z_{t-2,0}. \quad (7)$$

From the distribution table of S-box, for a fixed output difference, there will be about 2^8 corresponding input, thus we can sieve 2^8 possible $(y^a_{t-3,0}, y^b_{t-3,0})$.

(b) We explore a new technique to discover the correct $(y^a_{t-3,0}, y^b_{t-3,0})$ among the 2^8 sieved values. For each left $(y^a_{t-3,0}, y^b_{t-3,0})$, query the MAC with about 2^8 different $(\overline{M^a}, \overline{M^b})$, where

$$\overline{M^a} = \{x^a_1, \cdots, x^a_{t-3}, x_{t-2}, x_{t-1}, x^a_t\}, \overline{M^b} = \{x^b_1, \cdots, x^b_{t-3}, x'_{t-2}, x'_{t-1}, x^b_t\}.$$

are constructed as follows, the correct $(y^a_{t-3,0}, y^b_{t-3,0})$ is expected to cause a collision.
For each sieved $(y_{t-3,0}, y'_{t-3,0})$, replace $(x^a_{t-2,0}, x^b_{t-2,0})$ with the exact pair $(x_{t-2,0}, x'_{t-2,0})$, where

$$x_{t-2,0} = x^b_{t-2,0} \oplus y_{t-3,0} \oplus y'_{t-3,0}, \ x'_{t-2,0} = x^a_{t-2,0} \oplus y_{t-3,0} \oplus y'_{t-3,0}.$$

−If $(y_{t-3,0}, y'_{t-3,0}) = (y^a_{t-3,0}, y^b_{t-3,0})$, the two inputs to the S-box are

$$x_{t-2,0} \oplus y^a_{t-3,0} = x^b_{t-2,0} \oplus y^a_{t-3,0} \oplus y^b_{t-3,0} \oplus y^a_{t-3,0} = x^b_{t-2,0} \oplus y^b_{t-3,0},$$
$$x'_{t-2,0} \oplus y^b_{t-3,0} = x^a_{t-2,0} \oplus y^a_{t-3,0} \oplus y^b_{t-3,0} \oplus y^b_{t-3,0} = x^a_{t-2,0} \oplus y^a_{t-3,0},$$

respectively, and the corresponding outputs are

$$S(x_{t-2,0} \oplus y_{t-3,0}) = z_{t-2,0} = z^b_{t-2,0}, S(x'_{t-2,0} \oplus y'_{t-3,0}) = z'_{t-2,0} = z^a_{t-2,0}.$$

In this way, the value $\overline{\Delta z_{t-2,0}} = \Delta z_{t-2,0}$, which means $\overline{\Delta z_{t-2}} = \Delta z_{t-2}$ since other bytes remain the same. Then (y_{t-3}, x_{t-2}) and (y'_{t-3}, x'_{t-2}) can lead to the same Δy_{t-2} where

$$x_{t-2} = (x_{t-2,0}, x^a_{t-2,1}, x^a_{t-2,2}, x^a_{t-2,3}), x'_{t-2} = (x'_{t-2,0}, x^b_{t-2,1}, x^b_{t-2,2}, x^b_{t-2,3}).$$

It is noted that the value $z_{t-2,0}$ only affects four bytes of y_{t-2}, which is the bytes of the first column $(y_{t-2,0}, y_{t-2,4}, y_{t-2,8}, y_{t-2,12})$, and the other bytes remain the same. So that the 2nd to 4th column of $\overline{\Delta y_{t-1}}$ are the same with Δy_{t-1}. Thus, there will be a collision at (y_t, y'_t) if and only if

$$S(y_{t-2,0} \oplus x_{t-1,0}) \oplus S(y'_{t-2,0} \oplus x'_{t-1,0}) = \Delta z_{t-1,0}. \tag{8}$$

Replace $(x^a_{t-1,0}, x^b_{t-1,0})$ with 2^8 different pairs $(x_{t-1,0}, x'_{t-1,0})$, there is excepted to have one satisfy equation (8).
Therefore, query the MAC with the corresponding 2^8 different $(\overline{M^a}, \overline{M^b})$, there will be one colliding pair on average.
−Else $(y_{t-3,0}, y'_{t-3,0}) \neq (y^a_{t-3,0}, y^b_{t-3,0})$, the two inputs to the S-box are

$$x_{t-2,0} \oplus y^a_{t-3,0} = x^b_{t-2,0} \oplus y_{t-3,0} \oplus y'_{t-3,0} \oplus y^a_{t-3,0},$$
$$x'_{t-2,0} \oplus y^b_{t-3,0} = x^a_{t-2,0} \oplus y_{t-3,0} \oplus y'_{t-3,0} \oplus y^b_{t-3,0}.$$

The following equation holds with probability 2^{-8}.

$$S(x_{t-2,0} \oplus y_{t-3,0}) \oplus S(x'_{t-2,0} \oplus y_{t-3,0}) = \Delta z_{t-2,0}.$$

Thus, $\overline{\Delta z_{t-2}}$ changes, so that $\overline{\Delta y_{t-2,4}} = 0$,$\overline{\Delta y_{t-2,8}} = 0$, and $\overline{\Delta y_{t-2,12}} = 0$ hold with probability 2^{-24}. Therefore, query the MAC with the 2^8 different $(\overline{M^a}, \overline{M^b})$, no collision will happen.

In this way, we can sieve the right $(y^a_{t-3,0}, y^b_{t-3,0})$. In the Appendix, we describe a concrete algorithm to recover $(y^a_{t-3,0}, y^b_{t-3,0})$, where only the correct solution is sieved. The value of $(y^a_{t-3,2}, y^b_{t-3,2})$, $(y^a_{t-3,8}, y^b_{t-3,8})$ and $(y^a_{t-3,10}, y^b_{t-3,10})$ can be recovered in a similar way.

3. **Recovering** $(y^a_{t-3,5}, y^b_{t-3,5}, y^a_{t-3,7}, y^b_{t-3,7}, y^a_{t-3,13}, y^b_{t-3,13}, y^a_{t-3,15}, y^b_{t-3,15})$.
According to the form of the injection input, we get the following equations:

$$\Delta z_{t-2,5} = S(y^a_{t-3,5}) \oplus S(y^b_{t-3,5}), \tag{9}$$
$$\Delta z_{t-2,7} = S(y^a_{t-3,7}) \oplus S(y^b_{t-3,7}), \tag{10}$$
$$\Delta z_{t-2,13} = S(y^a_{t-3,13}) \oplus S(y^b_{t-3,13}), \tag{11}$$
$$\Delta z_{t-2,15} = S(y^a_{t-3,15}) \oplus S(y^b_{t-3,15}). \tag{12}$$

From all the deduced value of $(y^a_{t-2,0}, y^b_{t-2,0}, y^a_{t-2,10}, y^b_{t-2,10})$, the next equations derived the round function hold.

$$y^a_{t-2,0} = 3S(y^a_{t-3,0} \oplus x^a_{t-2,0}) \oplus 2S(y^a_{t-3,5}) \oplus S(y^a_{t-3,10} \oplus x^a_{t-2,3}) \oplus S(y^a_{t-3,15}) \tag{13}$$
$$y^b_{t-2,0} = 3S(y^b_{t-3,0} \oplus x^b_{t-2,0}) \oplus 2S(y^b_{t-3,5}) \oplus S(y^b_{t-3,10} \oplus x^b_{t-2,3}) \oplus S(y^b_{t-3,15}) \tag{14}$$
$$y^a_{t-2,10} = S(y^a_{t-3,2} \oplus x^a_{t-2,1}) \oplus S(y^a_{t-3,7}) \oplus 3S(y^a_{t-3,8} \oplus x^a_{t-2,2}) \oplus 2S(y^a_{t-3,13}) \tag{15}$$
$$y^b_{t-2,10} = S(y^b_{t-3,2} \oplus x^b_{t-2,1}) \oplus S(y^b_{t-3,7}) \oplus 3S(y^b_{t-3,8} \oplus x^b_{t-2,2}) \oplus 2S(y^b_{t-3,13}) \tag{16}$$

We can see that, there are only four unknown bytes $(y^a_{t-3,5}, y^a_{t-3,15}, y^b_{t-3,5}, y^a_{t-3,15})$ in equations (9), (10), (13) and (14), where $(S(y^a_{t-3,5}), S(y^a_{t-3,15}),$

$S(y^b_{t-3,5}), S(y^a_{t-3,15}))$ can be easily computed. Then we obtain the corresponding input. Using equations (11), (12), (15) and (16) to recover ($y^a_{t-3,7}$, $y^a_{t-3,13}, y^b_{t-3,7}, y^a_{t-3,13}$) in the same way.

To sum up, we recover 8 bytes of y^a_{t-3} and 8 bytes of y^b_{t-3}, respectively.

4. **Recovering the internal state y_0.**
   For each solution derived, guess all the 2^{64} possible values for the rest bytes of y^a_{t-3}, compute backwards, i. e., do the decryption with $(x^a_{t-4}, \cdots, x^a_1)$ as the decryption subkey, and obtain 2^{64} values of y_0. For each y_0, compute the corresponding y^b_{t-3} with $(x^b_1, \cdots, x^b_{t-3})$ to filter out the wrong guesses.
   If there are more than one y_0 left, using the distinguisher to get another colliding pair, and repeat the recovery attack until there is only one value left. Two colliding pairs is enough to sieve the right y_0.

The complexity of this attack is dominated by the distinguishing attack and the final exhaustive search, which is about $2^{65.5}$ chosen messages and $2^{65.5}$ queries.

Second Preimage for ALPHA-MAC

Once the internal state y_0 is recovered, we can apply Huang et al.'s attack [8] to find the second preimages for ALPHA-MAC directly, or Biryukov et al.'s attack [2] to construct a selective forgery attack.

5 Conclusions

In this paper, distinguishing and forgery attacks on the ALRED construction and its specific instance ALPHA-MAC are presented. The complexity of the attacks is dominated by the birthday attack instead of exhaustive search. We construct distinguisher to detect inner near-collisions with specific differences rather than collisions, from which more information can be derived. Especially for ALPHA-MAC, our distinguishing attack can be converted into a recovery attack of the internal state, which equals to a subkey. The recovered equivalent subkey can result in the second preimage attack on ALPHA-MAC. It is to say, for any given text-MAC pair (M, C), we can obtain another $M' \neq M$ with the same C. Moreover, these attacks are also applicable to the MACs based on CBC and CFB encryption mode.

References

1. M. Bellare, R. Canetti, H. Krawczyk, Keying Hash Functions for Message Authentication, CRYPTO 1996, LNCS 1109, pp. 1-15, 1996.
2. A. Biryukov, A. Bogdanov, D. Khovratovich, T. Kasper, Collision Attacks on AES-Based MAC: ALPHA-MAC, CHES 2007, LNCS 4727, pp. 166-180, 2007.
3. J. Black, S. Halevi, H. Krawczyk, T. Krovetz, P. Rogaway, UMAC: Fast and Secure Message Authentication, CRYPTO 1999, LNCS 1666, pp. 216-233, 1999.
4. J. Black, P. Rogaway, CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions, CRYPTO 2000, LNCS 1880, pp. 197-215, 2000.
5. J. Daemen, V. Rijmen, AES Proposal : Rijndae. The First Advanced Encryption Standard Candidate Conference. NIST AES Proposal, 1998.

6. J.Daemen, V. Rijmen, A New MAC Construction Alred and A Specific Instance Alpha-MAC. FSE 2005, LNCS 3557, pp. 1-17, 2005.
7. D. W. Davies, A Message Authenticator Algorithm Suitable for A Mainframe Computer. CRYPTO 1984, LNCS 196, pp. 393-400, 1985.
8. J. Huang, J. Seberry, W. Susilo, On the Internal Structure of Alpha-MAC. VIETCRYPT 2006, LNCS 4341, pp. 271-285, 2006.
9. T. Iwata, K. Kurosawa, OMAC: One-Key CBC MAC, FSE 2003, LNCS 2887, pp. 129-153, 2003.
10. E. Jaulmes, A. Joux, F. Valette, On the Security of Randomized CBC-MAC beyond the Birthday Paradox Limit: A New Construction, FSE 2002, LNCS 2365, pp. 237-251, 2002.
11. J. Kim, A. Biryukov, B. Preneel, S. Hong, On the Security of HMAC and NMAC Based on HAVAL, MD4, MD5, SHA-0, and SHA-1. SCN 2006, LNCS 4116, pp. 242-256, 2006.
12. K. Kurosawa, T. Iwata, TMAC: Two-Key CBC-MAC, CT-RSA 2003, LNCS 2612, pp. 265-273, 2003.
13. NIST, FIPS 198, The Keyed-Hash Message Authentication Code (HMAC), 2002.
14. B. Preneel, P. Oorschot, MDx-MAC and Building Fast MACs from Hash Functions. CRYPTO 1995, LNCS 963, pp. 1-14, 1995.
15. X. Wang, W. Wang, K. Jia, M. Wang, New Distinguishing Attack on MAC using Secret-Prefix Method. Submitted to FSE 2009.
16. X. Wang, H. Yu, W. Wang, H. Zhang, T. Zhan, Cryptanalysis on HMAC/NMAC-MD5 and MD5-MAC. Submitted to EUROCRYPT 2009.
17. G. Yuval, How to Swindle Rabin. *Cryptologia*, vol. 3, pp. 187-189, 1979.

Appendix: An Algorithm to Recover the Internal State $(y^a_{t-3,0}, y^b_{t-3,0})$

Provide that $(x^a_1, \cdots, x^a_{t-1}, x^a_t)$ and $(x^b_1, \cdots, x^b_{t-1}, x^b_t)$ are the colliding pairs obtained in the distinguishing attack on ALPHA-MAC. We describe an algorithm to recover the internal value of $(y^a_{t-3,0}, y^b_{t-3,0})$.

Algorithm: Recovering the Internal State $(y^a_{t-3,0}, y^b_{t-3,0})$
INPUT:

$T_3 = \{(in, in') | S(in) \oplus S(in') = \Delta z_{t-2,0}\}$
$T_4 = \{(y_{t-3,0}, y'_{t-3,0}) | S(x^a_{t-2,0} \oplus y_{t-3,0}) \oplus S(x^b_{t-2,0} \oplus y'_{t-3,0}) = \Delta z_{t-2,0})\}$
$T_5 = \{\emptyset\}$

OUTPUT: $(y^a_{t-3,0}, y^b_{t-3,0})$.

1. For $(y_{t-3,0}, y'_{t-3,0}) \in T_4$,
   $T_4 = T_4 \backslash (y_{t-3,0}, y'_{t-3,0})$, deleting the element $(y_{t-3,0}, y'_{t-3,0})$ from T_4;
   Compute $\gamma = y_{t-3,0} \oplus y'_{t-3,0}$; $x_{t-2,0} = x^b_{t-2,0} \oplus \gamma$; $x'_{t-2,0} = x^a_{t-2,0} \oplus \gamma$;
   Let $x_{t-2} = (x_{t-2,0}, x^a_{t-2,1}, x^a_{t-2,2}, x^a_{t-2,3})$;
   $x'_{t-2} = (x'_{t-2,0}, x^b_{t-2,1}, x^b_{t-2,2}, x^b_{t-2,3})$.
   For $x_{t-1,0}$ from 0 to 255 do
   For $x'_{t-1,0}$ from 0 to 255 do
   Let $x_{t-1} = (x_{t-1,0}, x^a_{t-1,1}, x^a_{t-1,2}, x^a_{t-1,3})$;
   $x'_{t-1} = (x'_{t-1,0}, x^b_{t-1,1}, x^b_{t-1,2}, x^b_{t-1,3})$.
   Query the ALPHA-MAC with $\overline{M^a}, \overline{M^b}$, where

$$\overline{M^a} = \{x^a_1, \cdots, x^a_{t-3}, x_{t-2}, x_{t-1}, x^a_t\},$$
$$\overline{M^b} = \{x^b_1, \cdots, x^b_{t-3}, x'_{t-2}, x'_{t-1}, x^b_t\}.$$

   Let $\overline{C^a} = \text{ALPHA-MAC}(\overline{M^a})$, $\overline{C^b} = \text{ALPHA-MAC}(\overline{M^b})$.
   If $\overline{C^a} = \overline{C^b}$ then $T_5 = T_5 \cup (y_{t-3,0}, y'_{t-3,0})$, and goto 1.
2. For $(y_{t-3,0}, y'_{t-3,0}) \in T_5$ do
   Let $x_{t-2,0} = in \oplus y_{t-3,0}$, $x'_{t-2,0} = in' \oplus y_{t-3,0}$, where $in \neq y_{t-3,0} \oplus x^a_{t-3,0}$, $in' \neq y'_{t-3,0} \oplus x^b_{t-3,0}$.
   For $x_{t-1,0}$ from 0 to 255 do
   For $x'_{t-1,0}$ from 0 to 255 do
   Let $x_{t-1} = (x_{t-1,0}, x^a_{t-1,1}, x^a_{t-1,2}, x^a_{t-1,3})$;
   $x'_{t-1} = (x'_{t-1,0}, x^b_{t-1,1}, x^b_{t-1,2}, x^b_{t-1,3})$.
   Query the ALPHA-MAC with $\overline{M^a}, \overline{M^b}$, where

$$\overline{M^a} = \{x^a_1, \cdots, x^a_{t-3}, x_{t-2}, x_{t-1}, x^a_t\},$$
$$\overline{M^b} = \{x^b_1, \cdots, x^b_{t-3}, x'_{t-2}, x'_{t-1}, x^b_t\}.$$

   Let $\overline{C^a} = \text{ALPHA-MAC}(\overline{M^a})$, $\overline{C^b} = \text{ALPHA-MAC}(\overline{M^b})$.
   If $\overline{C^a} \neq \overline{C^b}$ then $T_5 = T_5 \backslash (y_{t-3,0}, y'_{t-3,0})$, and goto 2.
   Else return $(y_{t-3,0}, y'_{t-3,0})$.