Equivalent Key Recovery Attack on H^2-MAC Instantiated with MD5*

Wei Wang [1,2]
[1]School of Computer Science and Technology, Shandong University, Jinan 250101, China
[2]Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan 250100, China
weiwangsdu@sdu.edu.cn

Abstract

This paper presents the first equivalent key recovery attack on H^2-MAC-MD5, which conduces to a selective forgery attack directly. H^2-MAC is similar with HMAC except that the outer key is omitted. For HMAC-MD5, since the available differential paths are pseudo-collisions, all the key recovery attacks are in the related-key setting, while our attack on H^2-MAC-MD5 gets rid of this restriction. Based on the distinguisher of HMAC-MD5 proposed by Wang et al., a pair of intermediate chaining variables, i.e., the equivalent keys $(\tilde{K}, \tilde{K}')$, is detected which fulfils the specific conditions on (IV, IV') of the pseudo-collision. Then the inner key recovery attack on HMAC-MD5 explored by Contini and Yin is adopted to recover $(\tilde{K}, \tilde{K}')$. Consequently, the adversary can compute the valid MAC value of $M_0 \| M^*$ effortlessly, where M_0 is a fixed one-block message, and M^* can be any bit string.
Keywords: *Cryptanalysis, H^2-MAC-MD5, Distinguishing attack, Equivalent key recovery attack*

1 Introduction

A Message Authentication Code (MAC) algorithm is a keyed hash function, which takes a secret key K and a message M of arbitrary length as input, and produces a short tag as output. MAC is used to ensure the integrity and authenticity of messages, and has been widely used in internet security protocols (e.g., SSL/TLS, SSH, IPsec, etc.) and the financial sector for debit and credit transaction.

There are mainly three earlier methods to construct MAC algorithms, which are secret prefix, secret suffix and secret envelope [14] methods. The secret prefix method prepends a secret K to the message M before the hashing operation, and is the basic design unit for HMAC, which is a commonly-used, widely standardized MAC construction nowadays [1]. HMAC can take advantage of the modern cryptographic hash functions directly, and is provable secure under the two assumptions that the keyed compression function of the hash function and the key derivation function in HMAC are PRFs (Pseudo Random Functions)

*Supported by Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20100131120015) and Independent Innovation Foundation of Shandong University (Grant No. 2010TS069).

[2]. For an iterated Merkle-Damgård hash function H, HMAC is defined by

$$HMAC(K, M) = H(IV, (K_0 \oplus opad) \| H(IV, (K_0 \oplus ipad) \| M)),$$

where M is the message, K is the secret key shared between the originator and the intended receiver, IV is the initial value of H, K_0 is K's completion to a single block of H, *opad* and *ipad* are two fixed one-block constants. It is noted that, HMAC has to make an access to the secret key twice in the process, which causes some inconvenience.

H^2-MAC, which was proposed by Kan Yasuda in Information Security Conference (ISC) 2009 [22], is a new secret-prefix MAC, which is almost the same as HMAC except the second call to the secret key is omitted, i. e., the outer key is replaced by the fixed IV of the hash function. In this way, the secret key is only accessed once in the computation of H^2-MAC, and H^2-MAC gets rid of the disadvantage of the secret key management without losing the original advantage of HMAC. However, the absence of the outer key also causes negative sides, which do not exist in HMAC. As pointed out by the designer, the security of H^2-MAC requires its compression function be a secure PRF, and the adversary can easily compute any extension of that message locally once one of the intermediate chaining variables (ICVs) is leaked, i. e., the ICVs are equivalent to the secret key, and the security of H^2-MAC heavily depends on the secure computation of them. Thus, it is worth investigating into the impact of the outer key on security.

This paper studies the above question by cryptanalysis of H^2-MAC instantiated with a concrete hash function of MDx family, especially MD5 [13]. It is obvious that the secret key of H^2-MAC preforms in a similar way as the inner key of HMAC. Thus the inner key recovery attack on HMAC instantiated with specific hash functions can be applied to H^2-MAC directly, and the equivalent key of H^2-MAC is recovered.

In 2005, Wang informally discussed the impact of high probability collision path on early hash based MAC constructions like the secret prefix, secret suffix, and secret envelope methods [16]. Then a series of work had been proposed [6, 7, 8, 9, 12, 15, 17, 23]. When it comes to corresponding H^2-MAC, all the inner key recovery attacks can work with the same complexity.

While for HMAC-MD5, the situation becomes interesting. As the available differential paths are pseudo-collisions [5], which consist of two different IVs, (IV, IV'), all the key recovery attacks are in the related-key setting [7, 8, 12, 15]. One recent work presented a distinguishing attack on HMAC-MD5 without related keys, which is extended to a partial key recovery attack on MD5-MAC [19]. This motivates us to explore the first equivalent key recovery attack on H^2-MAC-MD5. To discard the related-key setting, we construct the distinguisher similar with [19], except that the length of the chosen messages has to be 3 blocks after padding in order to recover the equivalent key. The first blocks guarantee the existence of a one-block pair (M_0, M_0') which produces the related IV pair (IV, IV') of the pseudo-collision, the second blocks help to detect the specific (M_0, M_0'), and the third make sure that the padding information is not contained in the second block so that we can change any bits of the second block in the key recovery phase. Once the related (IV, IV') of the pseudo-collision is achieved by (M_0, M_0'), they become the intermediate variables of H^2-MAC-MD5 actually, and can be recovered by the bit flipping algorithm introduced in [7]. The complexity of our equivalent key recovery attack is 2^{97} queries. Consequently, we can compute the valid MACs for any extension of M_0 or M_0'.

This paper is organized as follows. Section 2 recalls the related backgrounds and definitions. Section 3 constructs a concrete distinguisher of H^2-MAC-MD5 first, and then

extends it to an equivalent key recovery attack, which leads to a selective forgery directly. Finally, Section 4 concludes this paper.

2 Preliminaries

We first explain the notations related to this paper, then present brief descriptions of MD5, the pseudo-collision of MD5 and H^2-MAC in this section.

2.1 Notations

H	:	a concrete hash function, such as MD5
$\tilde{H}$	:	a hash function without message padding
h	:	a compression function
IV	:	the initial value for H
ICV_k	:	the intermediate chaining variables for the k-th iteration of H
K	:	a secret key
$\tilde{K}$	:	an equivalent secret key
$x\|y$	:	the concatenation of two bit strings x and y
$\oplus$	:	the bitwise exclusive OR
$\lll s$	:	the left-rotation by s-bit

2.2 Brief Description of MD5

MD5 takes a message of arbitrary length as input and produces a 128-bit hash value by iterating a compression function h [13]. First, the input message M with length l is padded to a multiple of 512 bits. Then the padded message is divided into 512-bit blocks $(M_0, M_1, \cdots, M_{b-1})$, and processed by Merkle-Damgård iterative structure. For the k-th $(1 \le k \le b)$ iteration, the compression function $h(ICV_{k-1}, M_{k-1})$ performs in the following manner:

1. The 512-bit message block M_{k-1} is split into 16 message words $(m_0, m_1, \cdots, m_{15})$, and the 128-bit output of the $(k-1)$-th iteration, ICV_{k-1}, is split into four registers (A_0, B_0, C_0, D_0), where ICV_0 is the fixed initial value IV.
2. For $0 \le i \le 63$, compute

$$\begin{aligned} A_{i+1} &= D_i, B_{i+1} = B_i + (A_i + f(B_i, C_i, D_i) + w_i + c_i) \lll s_i, \\ C_{i+1} &= B_i, D_{i+1} = C_i, \end{aligned}$$

   where w_i is one of the 16 message words, c_i and s_i are step-dependent constants, f is a round-dependent Boolean function.
3. Output: $ICV_k = (A_0 + A_{64}, B_0 + B_{64}, C_0 + C_{64}, D_0 + D_{64})$, where $+$ means the addition modular 2^{32}.

After all the b blocks are processed, ICV_b is the hash value of the message M, i. e., $H(IV, M) = ICV_b$.

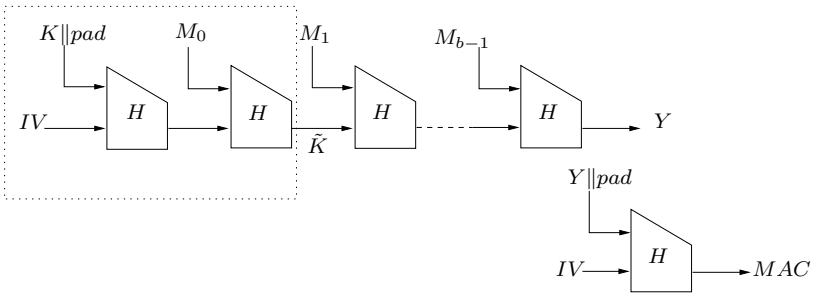

Figure 1. H^2-MAC Algorithm

2.3 Pseudo-collisions of MD5

All the previous attacks on HMAC-MD5 are based on the dBB pseudo-collisions found by den Boer and Bosselaers [5], so does this paper. The dBB pseudo-collisions satisfy the following relations:

$$h(IV, M) = h(IV', M), \quad (1)$$
$$IV \oplus IV' = (2^{31}, 2^{31}, 2^{31}, 2^{31}) = \Delta^{\mathrm{MSB}}, \quad (2)$$
$$\mathrm{MSB}(B_0) = \mathrm{MSB}(C_0) = \mathrm{MSB}(D_0), \quad (3)$$

where $IV = (A_0, B_0, C_0, D_0)$, M is a one-block message, and MSB means the most significant bit. The probability of the dBB pseudo-collision is 2^{-46}.

We adopt the definitions presented in [19]. The dBB conditions means relations (2) and (3) on the intermediate variables, and a collision of two-block messages $(x\|y, x'\|y)$ is called a dBB-collision if

1. (ICV_1, ICV_1') satisfies the dBB conditions, where $ICV_1 = h(IV, x)$, and $ICV_1' = h(IV, x')$.
2. $(x\|y, x'\|y)$ is a colliding pair, i. e., $H(IV, x\|y) = H(IV, x'\|y)$.

2.4 Brief Description of H^2-MAC

In ISC 2009, a new MAC construction, H^2-MAC [22], was presented, which is similar to the widely used HMAC algorithm [1]. For a message M of arbitrary length, and an n-bit secret key K, where n equals to the length of the MAC value, H^2-MAC is computed as follows (See Fig. 1):

$$H^2\text{-MAC}(K, M) = H(IV, H(IV, K\|pad\|M)),$$

where H is a concrete hash function, and *pad* is a fixed constant chosen according to H, in order to make sure that the length of $K\|pad$ is one block.

Compared with HMAC, H^2-MAC can also utilize current hash functions directly, moreover, it achieves higher performance and reduces the cost of managing the secret key since K is accessed only once. However, the absence of the outer key also leads to negative sides, such as the adversary can perform a selective forgery attack as soon as one of the intermediate chaining variables is leaked, and the security is based on a stronger assumption that the compression function must be a secure PRF.

Define an equivalent key $\tilde{K}$ as the intermediate chaining variable after $K\|pad\ \|M_0$ is processed, i. e., $\tilde{K} = \tilde{H}(IV, K\|pad\|M_0)$, where $\tilde{H}$ is a hash function without message padding. Once $\tilde{K}$ is recovered, the adversary can compute the valid MAC value of message $M_0\|M^*$ without knowing the secret key K, where M^* is an arbitrary message.

We denote H^2-MAC initiated with MD5 as H^2-MAC-MD5, and this paper presents an attack to recover the equivalent key $\tilde{K}$ of H^2-MAC-MD5 without related-key setting.

3 Equivalent Key Recovery Attack on H^2-MAC-MD5

We first present a distinguishing attack, which distinguishes H^2-MAC-MD5 from H^2-MAC instantiated with a random function. Then an equivalent key recovery attack is proposed based on the distinguisher. Finally, a selectively forgery attack can be performed. The complexity of the equivalent key recovery attack is 2^{97} MAC queries, which is dominated by the distinguisher.

3.1 Distinguishing Attack on H^2-MAC-MD5

The distinguishing attack on H^2-MAC-MD5 is inspired by Wang et al.'s attack on HMAC-MD5 which introduced new ideas to detect the inner near-collisions [19]. However, our distinguisher makes use of messages of three blocks instead of two, in order to proceed the key recovery attack.

To take advantage of the dBB pseudo-collision and get rid of the related-key setting, the key point is to replace the specific (IV, IV') of the pseudo-collision with the intermediate chaining variables (ICV, ICV') satisfying the dBB conditions, and find a method to identify such a pair. We adopt the definitions described in [19]. Suppose that we get a collision of H^2-MAC with the form $(M_0\|M_1\|M_2, M_0'\|M_1'\|M_2')$, where $M_1 = M_1'$, $M_2 = M_2'$, and all the padding information appears in M_2. According to the definition of ICV_k in Section 2.1, the intermediate chaining variable after $M[i]$ is processed is ICV_{i+2}, and the corresponding on for $M'[i]$ is ICV'_{i+2}. Denote $ICV_i \oplus ICV_i'$ as ΔICV_i. The collisions caused by $(M_0\|M_1\|M_2, M_0'\|M_1\|M_2)$ are divided into the following types:

- If $\Delta ICV_2 = 0$, it is an *Internal collision*.
- Else, $\Delta ICV_2 \neq 0$, it is an *External collision*. Moreover,
  - If (ICV_2, ICV_2') satisfies the dBB conditions, and $\Delta ICV_3 = 0$, it is a *dBB-collision*.

The core of the attack is to guarantee the existence of (ICV_2, ICV_2') satisfying the dBB conditions by a one-block message pair (M_0, M_0'), and identify it by appending different M_1 (See Fig. 2). If such a (ICV_2, ICV_2') exists, another collision can be obtained with less complexity than the birthday attack [24]. In this way, the adversary can identify the specific (ICV_2, ICV_2'), and conclude that the H^2-MAC is instantiated with MD5 rather than a random function. In a word, the adversary needs to distinguish the dBB-collision from others. The details are as follows:

1. Generate a structure S that composes of 2^{89} three-block messages, where

$$S = \{M_0\|M_1\|M_2|\ M_0 \text{ is chosen randomly, } M_1 \text{ and } M_2 \text{ are constants, and } M_2 \text{ contains all the padding information}\}.$$

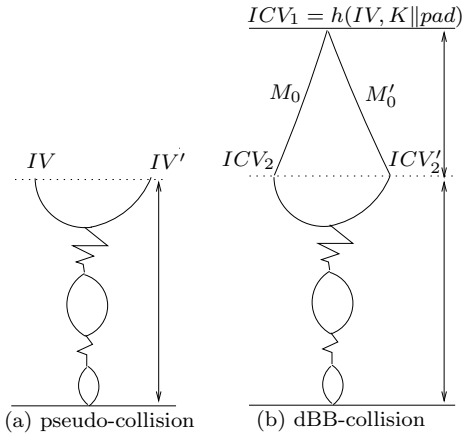

Figure 2. Distinguishing Attack Based on a Pseudo-Collision Path

Query the MAC with each message in S, and collect all colliding pairs $(M_0\|M_1\|M_2, M_0'\|M_1\|M_2)$ by the birthday attack.

2. For all colliding pairs collected in step 1, append another $\tilde{M}_1\|\tilde{M}_2$ to (M_0, M_0'), and query the two MACs for the new pair $(M_0\|\tilde{M}_1\|\tilde{M}_2, M_0' \|\tilde{M}_1\|\tilde{M}_2)$. If their MACs still collide, it is an internal collision. And the others are external collisions.
3. For each external collisions, detect the dBB-collisions. Append 2^{47} different $\tilde{M}_1\|\tilde{M}_2$ to (M_0, M_0'), respectively, and obtain their MACs. Among the 2^{47} new message pairs, if one collision is found, the corresponding (ICV_2, ICV_2') satisfies the dBB conditions. Then, append a randomly chosen $\tilde{M}_2$ to $M_0\|M_1$, and query their MACs. If they still collide, there must be $\Delta ICV_3 = 0$, and a dBB collision is identified. The adversary concludes that the H^2-MAC is instantiated with MD5.
4. If there is no dBB-collision found after all external collisions are sieved, the H^2-MAC is based on a random function.

Complexity evaluation:

For 2^{89} randomly chosen messages, there is expected to produce $2^{89} * 2^{89}/2 = 2^{177}$ pairs, so that step 2 collects $2^{177}/2^{128} = 2^{49}$ internal collisions, 2 dBB-collisions and $2^{49} * 3 \approx 2^{50.6}$ non-dBB external collisions. Thus, the complexity of step 3 is $2^{50.6} * 2^{47} = 2^{97.6}$ queries, which dominates the complexity of the whole distinguishing attack. To sum up, the complexity of our attack is about $2^{97.6}$ MAC computations, and the data complexity is about $2^{97.6}$ chosen messages.

3.2 Recovering the Equivalent Key $\tilde{K}$

As defined in Section 2.4, the equivalent key $\tilde{K}$ is the same as ICV_2 appeared in the distinguisher, and can be taken as the inner key of HMAC for fixed M_0. Thus, we can combine the above distinguisher with the bit flipping algorithm proposed by Contini and Yin [7] to construct an equivalent key recovery attack on H^2-MAC-MD5 without the related-key setting.

Suppose $(M_0\|M_1\|M_2, M_0'\|M_1\|M_2)$ is a dBB-collision detected by the distinguisher, the corresponding (ICV_2, ICV_2') satisfies the dBB conditions, i. e., $(\tilde{K}, \tilde{K}')$ fulfills the dBB conditions, which performs as the two related inner keys in the different attack of [7], so

that it can be recovered in the same way. We just recall their main idea as follows:

1. Obtain a message $M_1\|M_2$ which causes a dBB-collision under the fixed $(\tilde{K}, \tilde{K}')$. We can utilize the one obtained in the distinguishing attack.
2. Determine 80 bits of the intermediate registers $R_{14} = (A_{14}, B_{14}, C_{14}, D_{14})$ involved in the computation of $h(\tilde{K}, M_1)$. To do this, the adversary modifies certain bits of M_1 to create many new message pairs $(M_0\|M_1^*\|M_2, M_0'\|M_1^*\|M_2)$ smartly, and deduces the corresponding bits of R_{14} according to whether any new pairs collides or not. Since M_2 contains all the padding information, we can change M_1 randomly without affecting M_2. For more details, please refer to [7].
3. Compute candidates of $(\tilde{K}, \tilde{K}')$. Guess the other 48 bits of R_{14} and compute backwards. For $i = 13, 12, \cdots, 0$, compute

$$\begin{aligned} B_i &= C_{i+1}, C_i = D_{i+1}, D_i = A_{i+1}, \\ A_i &= (B_{i+1} - B_i) \ggg s_i - f(B_i, C_i, D_i) - w_i - c_i. \end{aligned}$$

   In this way, $R_0 = (A_0, B_0, C_0, D_0)$ is derived, which is a candidate of $\tilde{K}$. And the corresponding candidate of $\tilde{K}'$ can be obtained by the dBB conditions.
4. Finally, there will be 2^{48} candidates for $(\tilde{K}, \tilde{K}')$, and the the correct one can be identified according to the equitation $\tilde{H}(\tilde{K}, M_1\|M_2) = \tilde{H}(\tilde{K}', M_1\|M_2)$.

Complexity evaluation:
The complexity of step 1 depends on the distinguishing attack, which is about $2^{97.6}$ queries and 2^{89} table lookups, and step 2 and step 3 takes the same as the inner-key recovery attack of Contini and Yin, which is 2^{45} MD5 operations. Therefore, the complexity of the equivalent key recovery attack is $2^{97.6}$ queries.

Remark. From the above analysis, it is noted that to proceed the key recovery attack, the first thing is to construct a pair $(\tilde{K}, \tilde{K}')$ meeting the dBB conditions, and identify it. As soon as we detect such a pair caused by (M_0, M_0'), we can construct a dBB-collision in the following way:

1. Randomly choose 2^{47} two-block messages $M_1\|M_2$, where the padding information only appears in M_2.
2. Obtain the MAC value of 2^{47} pairs $(M_0\|M_1\|M_2, M_0'\|M_1\|M_2)$. Among them, two collisions are expected to exist because the probability of a dBB-collision is 2^{-46}.

The complexity of the above steps is 2^{47} queries.

Therefore, in the distinguishing attack, the adversary can adopt the distinguisher proposed by [19] to find a pair $(\tilde{K}, \tilde{K}')$ satisfying dBB conditions, and utilize the above technique to obtain a dBB collision, which is required in step 1 of the equivalent key recovery attack. In this way, the complexity of the whole attack is dominated by the distinguishing attack, which is reduced to 2^{97} queries.

3.3 Selective Forgery Attack

Selective forgery attack means that an adversary is able to create a valid MAC σ for a message M of his choice, where M has not been signed or MACed in the past by the legitimate signer/MAC generator, and must be fixed before the start of the attack.

Once the adversary recovered the equivalent key $\tilde{K}$, he can perform the selective forgery immediately, and only takes 1 MAC computation. From

$$H^2\text{-MAC}(K, M_0\|M^*) = H(IV, H(IV, K\|pad\|M_0\|M^*)) = H(IV, \tilde{K}\|M^*),$$

where M_0 is the one that leads to $\tilde{K}$ in the distinguishing attack, and M^* can be any arbitrary messages, the adversary can compute the legal MAC value of the message $M_0\|M^*$ without knowing K. Similarly, the legal MAC value of message $M_0'\|M^*$ can be obtained because of the recovery of $\tilde{K}'$.

4 Conclusions

In this paper, we further study the impact of the outer key from the cryptanalysis side, and insight about the gap in security between H^2-MAC and HMAC when they are applied with one of the current hash functions, MD5.

Since H^2-MAC is equivalent to HMAC without the outer key, the inner key recovery attacks on HMAC instantiated with any concrete hash functions can be applied to H^2-MAC immediately. While for MD5, the situation becomes interesting. Since there only exists a distinguishing attack on HMAC-MD5 without related keys [19] in published literatures. In contrast, we present an equivalent key recovery attack on H^2-MAC-MD5 getting rid of the related-key restriction. It is noted that our results accord with the security proof associated with H^2-MAC, which is provable secure on the assumption that the underlying compression function is a secure Pseudo-Random Function (PRF) [22], while HMAC's security based on a weaker-than-PRF property of the compression function [2].

Inspired by Wang et al.'s idea of detecting the inner near-collision of HMAC-MD5 [19], the adversary can guarantee the existence of a pair of intermediate chaining variables, i. e., the equivalent keys $(\tilde{K}, \tilde{K}')$, which satisfies the specific conditions on (IV, IV') of the pseudo-collision, by one-block message M_0, and identify it by appending different one-block messages M_1. If such $(\tilde{K}, \tilde{K}')$ is identified, the equivalent key recovery attack can be processed, where the bit flipping algorithm proposed by Contini and Yin [7] is adopted. The complexity of the equivalent key recovery attack is 2^{97} queries, which is dominated by the distinguishing attack. Moreover, the equivalent key recovery attack leads to a selective forgery attack, since the adversary can compute the valid MAC value of any extension of M_0 effortlessly , where M_0 is the one-block message corresponding to $\tilde{K}$.

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.: New Proofs for NMAC and HMAC: Security without Collision-Resistance. In: Dwork. C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 602–619. Springer, Heidelberg (2006)

[3] Biham, E., Chen, R.: Near-Collisions of SHA-0. In: Franklin, M.K. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 290–305. Springer, Heidelberg (2004)

[4] Biham, E., Chen, R., Joux, A., Carribault, P., Lemuet, C., Jalby, W.: Collisions of SHA-0 and Reduced SHA-1. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 36–57. Springer, Heidelberg (2005)

[5] den Boer, B., Bosselaers, A.: Collisions for the Compression Function of MD5. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 293–304. Springer, Heidelberg (1994)

[6] Chabaud, F,: Joux, A.: Differential Collisions in SHA-0. In Krawczyk, H. (ed.) CRYPTO 1998, LNCS, vol. 1462, pp. 56–71. Springer, Heidelberg (1998)

[7] 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)

[8] Fouque, P.-A., Leurent, G., Nguyen, P.Q.: Full Key-Recovery Attacks on HMAC/NMAC-MD4 and NMAC-MD5. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 13–30. Springer, Heidelberg (2007)

[9] 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, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 242–256. Springer, Heidelberg (2006)

[10] Preneel, B., and 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)

[11] Rechberger, C., Rijmen, V.: On Authentication with HMAC and Non-Random Properties. In: Dietrich, S., Dhamija, R. (eds.) Financial Cryptography 2007. LNCS, vol. 4886, pp. 39–57. Springer, Heidelberg (2007)

[12] Rechberger, C., Rijmen, V.: New Results on NMAC/HMAC when Instantiated with Popular Hash Functions. Journal of Universal Computer Science, 14(3), 347–376 (2008)

[13] Rivest, R.L.: The MD5 Message Digest Algorithm. Request for Comments (RFC 1321), Network Working Group (1992)

[14] Tsudik, G.: Message Authentication with One-Way Hash Functions. ACM Comput. Commun. Rev., 22 (5), 29–38 (1992)

[15] Wang, L., Ohta, K., Kunihiro, N.: New Key-Recovery Attacks on HMAC/NMAC-MD4 and NMAC-MD5. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 237–253. Springer, Heidelberg (2008)

[16] Wang, X.: What's the Potential Danger Behind the Collisions of Hash Functions, 2005. ECRYPT Conference on Hash Functions, Krakow, available via http://www.ecrypt.eu.org/stvl/hfw/.

[17] Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the Hash Functions MD4 and RIPEMD. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 1–18. Springer, Heidelberg (2005)

[18] Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)

[19] Wang, X., Yu, H., Wang, W., Zhang, H., Zhan, T.: Cryptanalysis on HMAC/NMAC-MD5 and MD5-MAC. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 121–133. Springer, Heidelberg (2009)

[20] Wang, X., Yu, H., Yin, Y.L.: Efficient Collision Search Attacks on SHA-0. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 1–16. Springer, Heidelberg (2005)

[21] Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)

[22] Yasuda, K.: HMAC without the "Second" Key. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 443–458. Springer, Heidelberg (2009)

[23] Yu, H., Wang, G., Zhang, G., Wang, X.: The Second-Preimage Attack on MD4. In: Desmedt, Y., Wang, H., Mu, Y., Li, Y. (Eds.) CANS 2005, LNCS vol. 3810, pp. 1–12. Springer, Heidelberg (2005)

[24] Yuval, G.: How to Swindle Rabin. Cryptologia, 3, 187–190 (1979)