On the Security of H^2-MAC*

Fanbao Liu[1], Tao Xie[1], and Changxiang Shen[2]

[1]School of Computer, National University of Defense Technology, Changsha, 410073, Hunan, P. R. China, Email: liufanbao@gmail.com
[2]School of Computer, Beijing University of Technology, 100124, Beijing, P. R. China

Abstract

H^2-MAC was proposed by Yasuda to increase efficiency over hash-based message authentication code (HMAC) by omitting its outer key, and keep the advantages and security of HMAC at the same time. We propose an efficient method to break H^2-MAC, by using a generalized birthday attack to recover the equivalent key, under the assumption that the underlying hash function is secure (collision resistance). We can successfully recover the equivalent key of H^2-MAC instantiated with any Merkle-Damgård hash function in about $2^{n/2}$ on-line message authentication code (MAC) queries and $2^{n/2}$ off-line MAC computations with good probability. We argue that the pseudo random function-affix (PRF-AX) assumption of the origin security proof of H^2-MAC, and we prove that the security of H^2-MAC is dependent on the collision resistance of the underlying hash function, instead of the PRF assumption.

Keywords: H^2-MAC, Equivalent Key Recovery, Pseudo Random Function, Collision Resistance, Birthday Paradox.

1 Introduction

HMAC [2, 3], a derivative of NMAC, is a practically and commonly used, widely standardized MAC construction. HMAC has two advantages. First, HMAC can directly make use of current hash functions, the most widely used ones are based on Merkle-Damgård construction [4, 5], as black boxes. Second, it is provable secure under the assumption that the compression function of the underlying hash function is a pseudo random function (PRF) [3].

For an iterated hash function H with Merkle-Damgård construction, HMAC is defined with secret prefix approach [6], by

$$\mathrm{HMAC}_{(k_{\mathrm{in}},\, k_{\mathrm{out}})}(M) = H(k_{\mathrm{out}}||H(k_{\mathrm{in}}||M)) \tag{1}$$

where M is an input message with arbitrary length, k_{in} and k_{out} are secret b-bit keys derived from a base key K.

However, HMAC has a drawback of managing its secret keys. It has to call the secret keys twice to complete the MAC computation[7, 8].

*This paper is the extended version of [1], which was first appeared in ISA 2012.

In ISC 2009, Yasuda proposed H^2-MAC [9], a variant of HMAC [2, 3] with single key, to remedy the drawback of managing its two secret keys of HMAC and keep HMAC's advantages and security at the same time. H^2-MAC is defined by removing the outer key of HMAC. Given a Merkle-Damgård hash function H mapping $\{0,1\}^*$ to $\{0,1\}^n$, and an arbitrary message $M \in \{0,1\}^*$, H^2-MAC is defined with secret prefix approach [6], as $H^2\text{-MAC}_{(K)}(M) = H(H(K||pad||M))$, where K is an n-bit key, and $pad \in \{0,1\}^{m-n}$ is a fixed constant.

H^2-MAC is proven to be a secure PRF (pseudo random function) under the assumption that the underlying compression function is a PRF-AX [9].

In 2011, Wang [10] proposed an equivalent key recovery attack to H^2-MAC instantiated with the broken MD5 [11, 12, 13], combining the technologies used in [14] and [15], with complexity about 2^{97} on-line MAC queries.

Our contributions. We propose the first equivalent key recovery attack that breaks the security of H^2-MAC instantiated with secure hash functions, without related key setting. The attack is based on the assumption that the underlying hash function is collision resistant (CR), which is a stronger notion than the PRF assumption of the underlying compression function in origin security proof in [9]. Hence, our attack is suitable to all of the H^2-MACs instantiated with secure[1] Merkle-Damgård hash functions, since our attack is based on the birthday paradox. This attack is also applicable to NMAC [16] and HMAC with two random keys, in a related key setting.

We notice that H^2-MAC applies the Merkle-Damgård hash functions directly in the outer hashing without any key, which can not hide the existence of collisions of the inner hashing. We break H^2-MAC by recovering its equivalent key through a generalized birthday attack with two groups. First, we get the corresponding MAC values of H^2-MAC by on-line queries in group G_1, using different 1-block messages. Second, we directly compute the values of $H(H(C||m))^2$, called H^2, in group G_2 through off-line, where Cs and ms can be both randomly generated. If the number of queries in G_1 is $2^{n/2}$ and the number of computations in G_2 is also $2^{n/2}$, then, there is a pair (m, m') of the inner hashing part of H^2-MAC and H^2 that equate with good probability [17]. Hence, the equivalent key of H^2-MAC is recovered by computing the corresponding value of H^2.

Since the on-line MAC queries to the H^2-MAC oracle can not be done concurrently, and the off-line computations of the H^2 can be completed in parallel, we can improve the attack efficiency by reducing the on-line queries and increasing the number of off-line computations at the same time, without reducing the success ratios of the attack. Moreover, once the off-line computation is done, it can be reused while recovering the equivalent key of other H^2-MACs with different keys.

We also discuss the origin security proof in [9], we argue that the PRF-AX assumption of the underlying hash function is no stronger than PRF, hence, we conclude that the origin proof of H^2-MAC is incorrect.

Finally, we prove that the security of the H^2-MAC is totally dependent on the collision resistance of the underlying hash function, instead of the PRF assumption.

Organization of this paper. We introduce some preliminaries and background, such as birthday paradox, in section two. In section three, We break the security of H^2-MAC by using a generalized birthday attack with two groups, based on the assumption that the

[1]In this paper, a secure hash function means it is CR.

[2]The secret key of H^2-MAC is replaced with a constant, for example, the IV of the underlying hash function.

underlying hash function is collision resistance. Further, in section four, we discuss the origin security proof of H^2-MAC, and we further claim that thus proof is incorrect. We prove that the security of H^2-MAC is totally dependent on the collision resistance of the underlying Merkle-Damgård hash function, instead of the PRF assumption. We conclude the paper in the last section.

2 Preliminaries

In this section, we first present some notations, and then we recall the birthday paradox in brief, and later we present a brief description of H^2-MAC, finally we recall the security proof of H^2-MAC.

2.1 Notations

Let h be a compression function mapping $\{0,1\}^n \times \{0,1\}^b \rightarrow \{0,1\}^n$, and let H be a concrete hash function mapping $\{0,1\}^* \rightarrow \{0,1\}^n$. Let IV be the initial chaining variable of H. Let k denote a secret key with b bits, K denote a secret key with n bits. $x||y$ denotes the concatenation of two bit strings x and y. $|G|$ denotes the number of elements of the set G. $\oplus$ means the bit wise exclusive OR. $pad(M)$ denotes the padding bits of M in Merkle-Damgård style. H^2 means that the secret key to H^2-MAC is replaced with a constant C or a known parameter to everybody, hence, H^2 can be also viewed as the double applications of the underlying hash function H.

2.2 Birthday Paradox

The famous birthday paradox is stated as follows [17]: "Let r be the number of the students in a classroom and let $q(r)$ be the probability that at least two students in this classroom have the same birthday. The minimal value of r is 23 for $q(r) \geq 1/2$."

A generalized variant. Given two groups G_1 with r elements, G_2 with s elements drawn uniformly and independently at random from $\{0,1\}^n$, find $x_1 \in G_1$ and $x_2 \in G_2$, such that $x_1 = x_2$.

The probability $\Pr(|G_1 \cap G_2| = i)$ that there are i distinct elements in the intersection of the two groups is denoted by $P(2^n, r, s, i)$. $P(2^n, r, s, i)$ converges towards a Poisson distribution $\wp_\lambda(i)$ with parameter λ, where $r \times s/2^n \rightarrow \lambda$, $r, s, 2^n \rightarrow +\infty$ [17].

A solution x_1, x_2 exists with good probability once $r \times s \geq 2^n$ holds, and if the list sizes are favorably chosen, the complexity of the optimal algorithm is $O(2^{n/2})$ [17, 18].

The birthday problem has numerous applications throughout cryptography and cryptanalysis, and the direct application is collision searching.

2.3 Brief Description of H^2-MAC

H^2-MAC [9] was proposed by Yasuda in ISC 2009, it is defined as

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

where K is an n-bit key. It is a reduced version of HMAC by removing the outer key.

Compared with HMAC, H^2-MAC can also utilize the underlying hash functions as black box, and it is also provable secure under the assumption that the underlying compression

function is a PRF-AF [9]. Moreover, H^2-MAC can achieve higher performance and simpler key management over HMAC, especially for short messages, since it only access the secret key once.

However, the key reduction of H^2-MAC introduces another security problem, as pointed out by the designer [9], once the intermediate chaining variable of the inner hashing (the equivalent key of H^2-MAC) is leaked, it can be used to perform a selective forgery attack. Hence, the security of H^2-MAC is also dependent on the secrecy of intermediate chaining variable.

2.3.1 Security Proof of H^2-MAC

To prove the security of H^2-MAC, the author proposed a notation of PRF-AX [9]. It is claimed that the notion of PRF-AX is almost the same as that of PRF except that an adversary is allowed to obtain a piece of additional information called *affix*. Given a keying compression function $h : \{0,1\}^{n+m} \rightarrow \{0,1\}^n$ via its IV as $h_K(\cdot) = h(K||\cdot)$. In the PRF-AX setting, the adversary is not only allowed to access to the $h_K(\cdot)$ oracle, but also to access to an additional *affix* oracle, which returns $h(IV||K||pad)$, to distinguish them from the random oracle $R : \{0,1\}^m \rightarrow \{0,1\}^n$ and a random string $r \xleftarrow{U} \{0,1\}^n$ where r is uniformly and randomly chosen.

The authors first proved that the multi-oracle $h \otimes \cdots \otimes h$ is a PRF-AX, under the assumption that the underlying hash function h is a PRF-AX. Second, $\tilde{H}^2$ is a secure PRF, if both $h \otimes \cdots \otimes h$ and h are PRF-AXs. Finally, they proved that H^2-MAC is a secure PRF, if $\tilde{H}^2$ is a PRF and h is a PRF-AX. Hence, they reduced the security of H^2-MAC to the security of the underlying hash function h, theoretically.

3 Breaking H^2-MAC Using Birthday Paradox

We call $I_K = H(K||pad||M)$ the inner hashing of H^2-MAC, $Oh = H(I_K)$ the outer hashing of H^2-MAC, respectively.

If we know the value of the inner intermediate chaining variable of H^2-MAC, $I_K = H(K||pad||M)$, we can construct any selective forgery attack to H^2-MAC. However, it seems that we can't get the value of inner hashing $H(K||pad||M)$ for the application of outer hashing Oh.

Attack principle. To find a way out, we notice that if we consider the inner intermediate chaining variable (the equivalent key) of H^2-MAC as an n-bit input x, then we can view H^2-MAC as a simple hash of $H(x)$. Intuitively, find a collision pair that satisfies $H(x) = H(x')$ is easier than recover the secret key of a MAC, even if H is collision resistant. However, we can't use the birthday attack with one group to recover the equivalent key, because we could not know the value of x and x', even a collision pair (x, x') is found.

Fortunately, we can use the generalized birthday attack with two groups. If we can get one collision pair (M, M') that not only satisfies $H^2\text{-MAC}_{(K)}(M) = H^2_{(c)}(M')$, but also satisfies $H(K||pad||M) = H(c||M')$, where c is a b-bit constant or a known parameter set by us[3]. Then, we get the very equivalent key K_e of H^2-MAC through the equation of $K_e = H(K||pad||M) = H(c||M')$. Finally, we already know the value of c and M', hence, K_e can be easily computed.

[3]Here, an inner collision happens between H^2-MAC and H^2.

So the equivalent key recovery attack to H^2-MAC is transformed to the problem of finding a collision pair in two groups, where the elements of one group must be computed by on-line query to H^2-MAC, and the elements of the other group can be computed directly through H^2 off-line. Thus problem is the generalized birthday attack with two groups (sometimes, it is also named as meet-in-the-middle attack).

Generalized Birthday Attack to Recover the Equivalent Key of H^2-MAC

Here, we apply the generalized birthday attack with two groups [17] to H^2-MAC and then recover its equivalent key $K_e = H(K||pad||M_0)$.

We use 1-block messages M_is to generate the corresponding H^2-MAC values, and use 1-block messages M'_js to generate the corresponding H^2 values, where $1 \leq i, j \leq 2^{n/2}$. The overall strategy of equivalent key recovery attack to H^2-MAC is shown as follows.

1. Generate a group one G_1 with $r = 2^{n/2}$ elements, by computing the corresponding values of $H(H(c||M'_j))$ for r different cs and M'_js, which can be randomly generated. Specifically, c can be a pre-chosen constant.
2. Generate a group two G_2 with $s = 2^{n/2}$ elements, by querying the corresponding values to H^2-MAC oracle with the secret key K for s different M_is, where M_is are randomly generated.
3. There is a pair (M_i, M'_j) that not only satisfies $H^2\text{-MAC}_K(M_i) = H_c^2(M'_j)$, but also satisfies $H(K||pad||M_i) = H(c||M'_j)$ (an inner collision between H^2 and H^2-MAC happens), with good probability [17].
4. Since $H(K||pad||M_i) = H(c||M'_j)$, and we know the value of c and M'_j, we can compute the value of $K_e = H(K||pad||M_i) = H(c||M'_j)$.
5. Let pad_0 and pad_1 be the padding bits of $K||pad||M_i$ and $K||pad||M_i||pad_0||x$, respectively, for arbitrary message x. Hence, we can directly generate the result of $H(K||pad||M_i||pad_0||x)$ by computing $y = h(K_e, x||pad_1)$, then we compute $H(y)$ further, finally we get the very value of $H^2\text{-MAC}(K||pad||M_i||pad_0||x)$.

Why inner collision. In the above attack, an inner collision must be found first. The problem is why an inner collision must happen. If we remove the outer hashing of H^2-MAC, we can directly observer that a collision pair (M_i, M'_j) will be found with good probability, after querying the oracle of $H(K||pad||M_i)$ and $H(c||M'_j)$ with enough times. We recall that the application of outer hashing of H^2-MAC can't hide the existence of such inner collision.

How to judge the inner collision. After a collision pair (M_i, M'_j) that satisfies $H^2\text{-MAC}_K(M_i) = H_c^2(M'_j)$ is found, we first generate the padding bits pad_0 for M_i and M'_j, where $pad_0 = pad(c||M'_j)$. Further, we randomly generate a message x, and append x to $M_i||pad_0$ and $M'_j||pad_0$, respectively. We query the corresponding MAC value on-line to the H^2-MAC oracle for $M_i||pad_0||x$, and we compute the corresponding value for $M_i||pad_0||x$ off-line using H^2. After that, we further check whether the equation $H^2\text{-MAC}_K(M_i||pad_0||x) = H_c^2(M'_j||pad_0||x)$ still holds. If so, (M_i, M'_j) is also an inner collision pair between H^2-MAC and H^2, the attack succeeds. Otherwise, (M_i, M'_j) is an outer collision pair, which will be simply discarded.

Success probability. We calculate the success probability of the above attack. We notice that $r = s = 2^{n/2}$ (hence $\lambda = r \cdot s/2^n = 1$), the probability sp of that at least one

inner collision happens is computed as

$$\begin{aligned} sp &= 1 - P(2^n, r, s, 0) = 1 - \wp_\lambda(0) + \varepsilon \\ &= 1 - e^{-1} + \varepsilon \geq 0.632 \end{aligned} \tag{3}$$

where $\varepsilon \leq 10^{-5}$ [17].

Complexity analysis. The elements of group G_1 computed by H^2 need $2^{n/2}$ off-line H^2 computations. The elements of group G_2 computed by H^2-MAC need $2^{n/2}$ on-line H^2-MAC queries. We can store the element values of both groups in hash tables. The above algorithm requires $O(2^{n/2})$ time and space to complete.

After an inner collision pair (M_i, M'_j) is found, we can apply $H_c^2(M'_j)$ to compute the equivalent key of the H^2-MAC. Eventually, we can use the recovered equivalent key k_e to launch any selective forgery attack to H^2-MAC without further on-line query, based on M_0. This claims that the security of H^2-MAC is totally broken, moreover, we point out that the security of H^2-MAC is solely dependent on the collision resistance of the underlying hash function, not the secrecy and strength of the used key.

However, it is interesting to notice that H^2-MAC is provable secure under the assumption of that the underlying compression function h is a PRF-AX [9], which means that collision resistance of the underlying hash function can be dropped. Thus proof and assumption obvious violate our result.

4 Some Optimizations over the attack

Here, we discuss some improvements over the equivalent key recovery attack to H^2-MAC, such as enlarging the success probability and achieving more parallelism in the attack.

4.1 Enlarging the success probability

In the above attack, the success probability of that at least one inner collision happens is at least 0.632, which is acceptable sometimes. However, we can enlarge the success probability, through doing more queries, since sp is determined by $P(2^n, r, s, 0)$, which converges towards a Poisson distribution with parameter λ.

For example, if we now want the success probability sp to be $sp \geq 1 - 10^{-4}$, by changing only r and s (but preserving $r = s$ both powers of 2), we can choose $r = s = 2^{n/2+2}$, and then $\lambda = r \cdot s/2^n = 16$, finally, we have

$$\begin{aligned} sp &= 1 - P(2^n, 2^{n/2+2}, 2^{n/2+2}, 0) = 1 - \wp_\lambda(0) + \varepsilon \\ &= 1 - e^{-16} + \varepsilon \geq 1 - 10^{-4} \end{aligned} \tag{4}$$

where $\varepsilon \leq 10^{-5}$.

4.2 Implementing more parallelism

We notice that the on-line queries to H^2-MAC can not be done concurrently, however, the off-line computations of H^2 can be executed in parallel. For a hash function with $n = 128$ bits result, the on-line queries of $2^{n/2} = 2^{64}$ times can not be completed in practical time. Even if the H^2-MAC oracle can reply at the speed of 10Gbit/second, this could require

continuous replying during 25,000 years. However, the 2^{64} off-line computation of H^2 is achievable by utilizing the parallelism computation. For example, if use 1 million PCs in Internet, this work may be done within 9 days.

In a extreme way, we hold the success probability of the equivalent key recovery attack to H^2-MAC with 0.632, but we reduce heavily the on-line queries to the H^2-MAC oracle, at the cost of increasing the off-line computation of H^2. We set $s = 2^{n/4}$ and $r = 2^{3 \cdot n/4}$ (hence $\lambda = r \cdot s/2^n = 1$). The success probability sp of that at least one inner collision happens is computed as

$$\begin{aligned} sp &= 1 - P(2^n, 2^{3 \cdot n/4}, 2^{n/4}, 0) = 1 - \wp_\lambda(0) + \varepsilon \\ &= 1 - e^{-1} + \varepsilon \geq 0.632 \end{aligned} \tag{5}$$

where $\varepsilon \leq 10^{-5}$, and $n = 128$.

However, this work still can not be done in practical, since $2^{3 \cdot n/4} = 2^{96}$ computation s of H^2 for $n = 128$ is still out of reach.

To do this attack more practically, we reduce the on-line queries to H^2-MAC to $s = 2^{n/2-10} = 2^{54}$ times, which can be done successfully. At the same time, we increase the off-line computations of H^2 to $r = 2^{n/2+10} = 2^{74}$ times, which is also reachable by utilizing computing parallelism. Since $\lambda = r \cdot s/2^n = 1$, the success probability sp of that at least one inner collision happens can be still computed as

$$\begin{aligned} sp &= 1 - P(2^n, 2^{n/2+10}, 2^{n/2-10}, 0) = 1 - \wp_\lambda(0) + \varepsilon \\ &= 1 - e^{-1} + \varepsilon \geq 0.632 \end{aligned} \tag{6}$$

where $\varepsilon \leq 10^{-5}$, and $n = 128$.

5 The Security Proof of H^2-MAC

5.1 The Re-measurement of PRF-AX

It is claimed that the notion of PRF-AX is almost the same as that of PRF except that an adversary is allowed to obtain *affix*. Given a keying compression function $h : \{0,1\}^{n+m} \to \{0,1\}^n$ via its IV as $h_K(\cdot) = h(K||\cdot)$. In the PRF-AX setting, the adversary is not only allowed to access to the $h_K(\cdot)$ oracle, but also to access to an additional *affix* oracle, which returns $h(IV||K||pad)$.

For example, an adversary queries the $h_K(\cdot)$ oracle for an arbitrary message x, he will get a response $h_K(x) = h(K||pad||x)$, also, he queries the *affix* oracle, gets additional information of $h(IV||K||pad)$. If we consider $V = h(K||pad)$ and $Z = h(IV||K||pad)$, $(K||pad)$ is the suffix of $(IV||K||pad)$, since h is a PRF and both queries of Z and V are prefix-free, the adversary will get no information of K from both of V and Z. From the aspect of secure MAC, which needs key protection, *affix* means nothing. So the property of PRF-AX is no more than PRF.

5.2 The revised Security Proof of H^2-MAC

We first recall the Theorem 3.1 of [19]; it says that H^* is a prefix-free VI-PRF (pf-VI-PRF), if the underlying compression function h is an FI-PRF. we list the detail of the Theorem as follow, where we change the notions to suit this paper.

Theorem 3.1 of [19]. Let h be a function family with $Dom(h) = \{0,1\}^b$, $Range(h) = \{0,1\}^n$, and key length n. Suppose h is $(t, q, 1, \varepsilon)$-secure and let $l \geq 1$. Then h^* is (t, q, l, ε)-secure against prefix-free distinguisher, where

$$t = t' - cq\cdot(l + n + b) \cdot (\text{Time}(h) + \log q) \quad (7)$$

$$\varepsilon = ql\varepsilon' \quad (8)$$

here, c is a specific, small constant whose value can be determined from the proof. For the details of this proof, please refer [19].

Based on this theorem, we can simply observe that the inner hashing $I_K = H(K||pad||M)$ of the H^2-MAC is a pf-VI-PRF[4].

Moreover, since the unknown I_K has fixed length (n bits), the outer hashing $Oh = H(I_K) = h(I_K)$ is an PRF (the assumption).

Combined with above results, we have the conclusion that H^2-MAC is a pf-VI-PRF, instead of a PRF. It means that we can not prove H^2-MAC is a secure MAC, from the aspect of proving that H^2-MAC is a PRF,

5.3 H^2-MAC is not a Secure MAC

We prove that the security of H^2-MAC is totally dependent on the (weak) collision resistance of the underlying hash function, instead of the PRF assumption.

If we assume that the compression function h is weak collision resistant (CR), then the Merkle-Damgård hash function H is also CR [5, 4], hence, H^2 is CR.

Since I_K is vulnerable to extension attack, if any intermediate chaining variable is leaked, then selective forgery can be made. An inner collision is enough to recover the equivalent key, for the outer hashing will not hide the existence of that. Hence, the security of H^2-MAC is totally dependent on the collision resistance of the underlying hash function, which is proved by our equivalent key recovery attack to H^2-MAC. We conclude H^2-MAC is merely a complex hash function to some extent, not a secure MAC.

6 Conclusion

We recover the equivalent key of H^2-MAC through applying a generalized birthday attack with two groups, based on the assumption of that the underlying hash function is CR. We can recover thus key in about $2^{n/2}$ on-line queries to H^2-MAC and $2^{n/2}$ off-line H^2 computations. Moreover, this attack can be further optimized, the success probability can be improved by doing more queries and computations, and the on-line queries can also be reduced to the extent of practicality by increasing the off-line computations, which can be done in parallel. Our attack shows that the security of H^2-MAC is totally dependent on the CR of the underlying hash function, instead of the PRF assumption, which claims that the security of H^2-MAC is totally broken. CR is a stronger basis than the assumption of that the underlying compression function is a PRF-AX in the origin paper to prove the security of H^2-MAC [9].

[4]Since h is a dual PRF, no matter keyed by the IV or the input message data.

Acknowledgement

We thank the anonymous reviewers for their valuable comments. This work was partially supported by the program "Core Electronic Devices, High-end General Purpose Chips and Basic Software Products" in China (No. 2010ZX01037-001-001), and supported by the 973 program of China under contract 2007CB311202, and by National Science Foundation of China through the 61070228 project.

References

[1] F. Liu, T. Xie, and C. Shen, "Equivalent key recovery attack to h2-mac," *International Journal of Security and its Applications*, vol. 6, no. 2, pp. 397 – 402, 2012.

[2] M. Bellare, R. Canetti, and H. Krawczyk, "Keying Hash Functions for Message Authentication," in *Advances in Cryptology CRYPTO' 96*, ser. Lecture Notes in Computer Science, N. Koblitz, Ed. Springer Berlin / Heidelberg, 1996, vol. 1109, pp. 1–15.

[3] M. Bellare, "New Proofs for NMAC and HMAC: Security Without Collision-Resistance," in *Advances in Cryptology - CRYPTO 2006*, ser. Lecture Notes in Computer Science, C. Dwork, Ed. Springer Berlin / Heidelberg, 2006, vol. 4117, pp. 602–619.

[4] I. Damgård, "A Design Principle for Hash Functions," in *Advances in Cryptology CRYPTO' 89 Proceedings*, ser. Lecture Notes in Computer Science, G. Brassard, Ed. Springer Berlin / Heidelberg, 1990, vol. 435, pp. 416–427.

[5] R. Merkle, "One Way Hash Functions and DES," in *Advances in Cryptology CRYPTO 89 Proceedings*, ser. Lecture Notes in Computer Science, G. Brassard, Ed. Springer Berlin / Heidelberg, 1990, vol. 435, pp. 428–446.

[6] G. Tsudik, "Message authentication with one-way hash functions," *SIGCOMM Comput. Commun. Rev.*, vol. 22, pp. 29–38, October 1992.

[7] G. Chun-Sheng and G. Ji-Xing, "Cryptanalysis of smart-vercauteren and gentry-halevi's fully homomorphic encryption," *International Journal of Security and its Applications*, vol. 6, no. 2, pp. 103 – 108, 2012.

[8] H. D. Gao, Y. J. Guo, J. Q. Cui, H. G. Hao, and H. Shi, "A communication protocol of rfid systems in internet of things," *International Journal of Security and its Applications*, vol. 6, no. 2, pp. 91 – 102, 2012.

[9] K. Yasuda, "HMAC without the "Second" Key," in *Information Security*, ser. Lecture Notes in Computer Science, P. Samarati, M. Yung, F. Martinelli, and C. Ardagna, Eds. Springer Berlin / Heidelberg, 2009, vol. 5735, pp. 443–458.

[10] W. Wang, "Equivalent Key Recovery Attack on H^2-MAC Instantiated with MD5," in *Information Security and Assurance*, ser. Communications in Computer and Information Science, T.-h. Kim, H. Adeli, R. J. Robles, and M. Balitanas, Eds. Springer Berlin Heidelberg, 2011, vol. 200, pp. 11–20.

[11] R. Rivest, "The MD5 Message-Digest Algorithm," RFC 1321 (Informational), Internet Engineering Task Force, Apr. 1992, updated by RFC 6151. [Online]. Available: `http://www.ietf.org/rfc/rfc1321.txt`

[12] X. Wang and H. Yu, "How to Break MD5 and Other Hash Functions," in *Advances in Cryptology EUROCRYPT 2005*, ser. Lecture Notes in Computer Science, R. Cramer, Ed. Springer Berlin / Heidelberg, 2005, vol. 3494, pp. 561–561.

[13] T. Xie, F. Liu, and D. Feng, "Could The 1-MSB Input Difference Be The Fastest Collision Attack For MD5?." Eurocrypt 2009, Poster Session, Cryptology ePrint Archive, Report 2008/391, 2008, `http://eprint.iacr.org/`.

[14] S. Contini and Y. Yin, "Forgery and Partial Key-Recovery Attacks on HMAC and NMAC Using Hash Collisions," in *Advances in Cryptology ASIACRYPT 2006*, ser. Lecture Notes in Computer Science, X. Lai and K. Chen, Eds. Springer Berlin / Heidelberg, 2006, vol. 4284, pp. 37–53.

[15] X. Wang, H. Yu, W. Wang, H. Zhang, and T. Zhan, "Cryptanalysis on HMAC/NMAC-MD5 and MD5-MAC," in *Advances in Cryptology - EUROCRYPT 2009*, ser. Lecture Notes in Computer Science, A. Joux, Ed. Springer Berlin / Heidelberg, 2009, vol. 5479, pp. 121–133.

[16] F. Liu, C. Shen, T. Xie, and D. Feng, "On the Security of NMAC and Its Variants," Cryptology ePrint Archive, Report 2011/649, 2011, `http://eprint.iacr.org/`.

[17] M. Girault, R. Cohen, and M. Campana, "A Generalized Birthday Attack," in *Advances in Cryptology EUROCRYPT 88*, ser. Lecture Notes in Computer Science, D. Barstow, W. Brauer, P. Brinch Hansen, D. Gries, D. Luckham, C. Moler, A. Pnueli, G. Seegmller, J. Stoer, N. Wirth, and C. Gnther, Eds. Springer Berlin / Heidelberg, 1988, vol. 330, pp. 129–156.

[18] D. Wagner, "A generalized birthday problem," in *Advances in Cryptology CRYPTO 2002*, ser. Lecture Notes in Computer Science, M. Yung, Ed. Springer Berlin / Heidelberg, 2002, vol. 2442, pp. 288–304.

[19] M. Bellare, R. Canetti, and H. Krawczyk, "Pseudorandom functions revisited: the cascade construction and its concrete security," *Foundations of Computer Science, Annual IEEE Symposium on*, vol. 0, p. 514, 1996.