JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 24, 1473-1484 (2008)

Analysis of All-or-Nothing Hash Functions*

PIN LIN[1,3], WENLING WU[1], CHUANKUN WU[1] AND TIAN QIU[2,3]
[1]*The State Key Laboratory of Information Security*
Institute of Software
Chinese Academy of Sciences
[2]*National Key Laboratory of Integrated Information System Technology*
[3]*Graduate School of Chinese Academy of Sciences*
Beijing 100190, P.R. China
E-mail: ping_linux@163.com
E-mail: {wwl; ckwu}@is.iscas.ac.cn; qiutian@ios.cn

The most popular method to construct hash functions is to iterate a compression function on the input message. This method is called Merkle-Damgård method. Most hash functions used in practice such as MD4, MD5, SHA-0, SHA-1 are based on this method. However this method is not always the best. For example, this method can not resist multi-collision attack. Recently some modifications of this method are proposed. These modified methods are based on Merkle-Damgård method and some improvements are made. A hash function based on All-or-Nothing property is one of these improvements. All-or-nothing property is an encryption mode for block ciphers. It has the property that one must decrypt all cipher blocks to determine any plain-text block. All-or-nothing hash function is a kind of hash function constructed with the all-or-nothing property. The authors of it claim that it is more secure than those common hash functions. In this paper, we will show that this is not true and there are still some flaws on this improved method.

Keywords: hash functions, compression functions, random oracle, all-or-nothing, block cipher

1. INTRODUCTION

1.1 Preliminaries on Hash Functions

Cryptographic hash functions are very important primitives in cryptography. They are widely used in many applications such as message authentication codes (MAC), manipulation detection code (MDC) and digital signature schemes. A cryptographic hash function processes an arbitrary-length input and has a fixed-length output. The most common method to construct hash functions is to iterate a compression function on the input message. This method was proposed by Merkle [2] and Damgård [1] independently. It is called Merkle-Damgård method (briefly MD method). The method is described as follows:

$h_0 = IV$

Received January 8, 2007; revised October 8, 2007; accepted November 12, 2007.
Communicated by Wen-Guey Tzeng.
* This work was supported by National Natural Science Foundation of China (grant No. 90604036), Major State Basic Research Development Program of China (973 Program, grant No. 2004CB318004) and National High-Tech Research and Development Program of China (863 Program, grant No. 2007AA01Z470).

$$h_i = f(h_{i-1}, m_i),\ 1 \le i \le l \qquad (1)$$
$$H(M) = h_l$$

where H denotes the hash function, f denotes the compression function, $M = (m_1, \ldots, m_l)$ denotes the whole input message which is divided into l blocks, m_i denotes the message block processed by the compression function at the ith step. h_i is called a chain value which is the intermediate value before the final output. Before being hashed, the messages need to be padded using an unambiguous padding rule and divided into some fixed-length blocks. Usually the length in binary of the message is padded, and the length of padded message is the multiple of the length of one block. The padding rule mentioned above is called MD-Strengthening. Lai *et al*. prove that if a hash function based on MD method has no MD-Strengthening padding rule, it is not secure [17]. A secure hash function must satisfy three conditions listed as follows [13].

1. Collision resistant: it should be computationally infeasible to find a pair $m \neq m'$ of inputs to the hash function H such that $H(m) = H(m')$.
2. 2nd pre-image resistant: for a given m, it should be computationally infeasible to find $m \neq m'$ such that $H(m) = H(m')$.
3. Pre-image resistant: it should be computationally infeasible, for a given value y, to find m such that $H(m) = y$.

Here H denotes the hash function and m, m' denote the two different messages to be processed. Merkle and Damgård have proved that if the compression function is collision resistant, the hash function constructed with it is also collision resistant. Subsequently, Black *et al*. prove that if the compression function is pre-image resistant and 2nd pre-image resistant, the hash function constructed with it is pre-image resistant and 2nd pre-image resistant [15]. Assuming the output length of a hash function is n bits, then the security of an ideal hash function can be scaled by the following conditions:

1. Given an output of a hash function, the complexity to find a pre-image should be $O(2^n)$.
2. Given a message and its hash value, the complexity to find a second pre-image should be $O(2^n)$.
3. The complexity to find a collision pair should be $O(2^{n/2})$.

The three conditions respectively mean the hash function is collision resistant, second pre-image resistant and pre-image resistant. For second pre-image attack, the time complexity is known as $O(2^n)$ before the recent results of Kelsey and Schneier [14]. Kelsey *et al*. have shown that the complexity of second pre-image attack on the hash function based on MD method is only $O(2^{n/2})$. These attacks mentioned above are called generic attacks on hash functions because these attacks don't depend on any particular hash function. The complexity of these generic attacks is the upper bound, so if there are no attacks on a hash function better than the generic attacks, the hash function is called an ideal hash function.

1.2 Attacks on Hash Functions Based on MD Method

Most hash functions used in practice are based on MD method such as MD5 [18], SHA-0 [19], SHA-1 [20] *etc*. These hash functions are called dedicated-designed hash functions because the compression functions of these hash functions are specially designed. These hash functions are very fast but the compression functions need to be carefully designed and the security cannot be proved. Recently the weakness of some dedicated-designed hash functions such as SHA-0 have been found. Wang *et al*. [5-8] have shown that finding collisions on these hash functions can be much faster than the generic attacks *i.e.* these hash functions are not collision resistant. For example, the output length of SHA-0 is 160 bit, if SHA-0 is ideal, the complexity to find a collision for it should be $O(2^{80})$, but Wang *et al*. improve the complexity to $O(2^{39})$. Another kind of hash function is the block-cipher-based hash functions whose compression functions are constructed with block ciphers. The security of block-cipher-based hash functions can be proved in the ideal model. Preneel *et al*. consider all 64 block-cipher-based hash functions which are called PGV scheme [21], and give the security analysis of these hash functions. However, the focus of [21] is on attacks not on strict proofs. In 2002, Black *et al*. proved the security of these schemes in the black-box model and divided these schemes into three groups [15]. Most block-cipher-based hash functions is the variety of PGV schemes, for example, those schemes in [13]. For the MD method itself, Joux has shown that there is an attack on this method named multi-collision attack which can find more than two collisions much faster than expected [4]. Assuming the output length of hash functions is n bit, if the MD method is ideal, the complexity to find t collisions for hash functions based on it should be $O(2^{\frac{(t-1)n}{t}})$, however, Joux shows that the complexity to find 2^t collisions is only $O(t2^{\frac{n}{2}})$. It means that there is some flaw in the MD method. Multi-collision attack is described in detail in the next section. All the attack mentioned above are based on message manipulation. To avoid these attacks, some modifications of MD method are proposed. One way is to increase the output size of compression functions, another is to preprocess the input messages before hashing it. The former has been independently proposed by Lucks [9], Hirose [10] and Nandi [11]. The latter also has some results [16]. Although multi-collision attack was proposed after [3], the schemes in [3] have similar properties to those in [16] *i.e.* they firstly convert the original messages into pseudo messages and then process these pseudo messages to get the final output. After all, the schemes in [3] are not proposed to resist multi-collision attack, so one scheme in [3] exists some flaw under multi-collision attack and needs to be modified.

1.3 Result of This Paper

In this paper, we first review the results proposed by Shin *et al*. in [3], where the authors propose three schemes and give the security analysis of these schemes, then we show that the security analysis of these schemes is not correct and give the correct security bound. We also show that one of the three schemes is not multi-collision attack resistant which contradict the claim in [3] that all schemes can resist the attacks using message manipulation technique, and we give a modification of this scheme.

2. ANALYSIS OF THE ALL-OR-NOTHING SCHEMES

2.1 Description of the Schemes

In the design of hash functions, Shin *et al*. propose three schemes which use the all-or-nothing property to preprocess the input message and then use the traditional hash functions based on MD method to get the final hash value.

In 1997, a new encryption mode for block ciphers called all-or-nothing encryption [12] was proposed by Rivest. This encryption mode has the property that if one wants to determine any one block Of the plain text, the entire cipher text must be received and decrypted. In [3], all-or-nothing mode is described as follows:

1. Let the input message block be $m_1, m_2, \ldots, m_s$.
2. Choose at random a key K for a block cipher used to transform the message.
3. Compute the output sequence $m'_1, m'_2, \ldots, m'_{s'}$ for $s' = s + 1$ as follows:
   (1) $m'_i = m_i \oplus E(K, i)$, for $i = 1, 2, 3, \ldots, s$.
   (2) Let $m'_{s'} = K \oplus h_1 \oplus \ldots \oplus h_s$,
   where $h_i = E(K_0, m'_i \oplus i), i = 1, 2, \ldots, s$,
   where K_0 is a fixed, publicly-known encryption key.

Here E denotes a block cipher. $m'_i = m_i \oplus E(K, i)$ is called the package transform which converts the original messages into pseudo messages. The key K of the block cipher for the package transform is publicly known, and does not need to be the same as the block cipher at step (2) which encrypts the output sequence. It is easy to see that the package transform is invertible. The plain text can be recovered as follows:

1. $K = m'_{s'} \oplus h_1 \oplus \ldots \oplus h_s$.
2. $m_i = m'_i \oplus E(K, i), i = 1, 2, \ldots, s$.

If any block of the output sequence is unknown, the randomly chosen key K cannot be computed, and so it is infeasible to compute any message block. This property is used in [3] to construct three schemes, where hash functions or some one way functions are used substitute the block cipher to transform the message into pseudo messages.

The following notations are used in [3]:

– n: the length of an output of a hash function.
– k: the block size of a hash function.
– m: an input message.
– m': a pseudo-message resulting from the all-or-nothing transform.
– K: a randomly chosen k-bit key.
– K_p: a fixed and publicly-known k-bit key.
– $h()$: an arbitrary hash function.
– $h_x(y)$: hash an input y with an initial value x.
– H_i: the chain value for the ith step.
– IV: initial value of a hash function.
– $\oplus$: bitwise *XOR*.

– ||: concatenation.
– $\overline{Z}$: transform n-bit Z to k-bit.

Using all-or-nothing property and notations mentioned above, three schemes are constructed in [3] as follows.

The first scheme:

(1) Partition the input message m into t k-bit blocks, $m_1, m_2, \ldots, m_t$.
(2) Generate a random k-bit key K.
(3) Compute the input value as follows: $H_0 = IV$, $m'_{t+1} = K$.
For $i = 1$ to t {
$m'_i = m_i \oplus f(K, H_{i-1}, i)$
$m'_{t+1} = m'_{t+1} \oplus g(K_p, m'_i, i)$
$H_i = h_{H_{i-1}}(m'_i)$
}
$H_{t+1} = h_{H_t}(m'_{t+1})$
(4) Output $(m' \| H_{t+1})$.

Here f and g are two different block ciphers as all-or-nothing scheme described above.

The second scheme:

(1) Partition the input message m into t n-bit block, $m_1, m_2, \ldots, m_t$.
(2) Generate a k-bit random key K.
(3) Compute the input value as follows:

$$m'_0 = IV,\ m'_i = m_i \oplus h_{m'_{i-1}}(K \oplus i),\ i = 1, 2, \ldots, t.$$

(4) Compute the last pseudo-message block m'_{t+1}:

$$m'_{t+1} = K \oplus \{\overline{h_{m'_1}(K_p \oplus 1 \oplus \ldots \oplus h_{m'_t}(K_p \oplus t)}\}.$$

(5) Output $(m' \| h_{IV}(m'_{t+1}))$.

The Third scheme:

(1) Partition the input message m into t n-bit block, $m_1, m_2, \ldots, m_t$.
(2) Generate a k-bit random key K.
(3) Compute

$$m'_0 = IV,\ m'_i = m_i \oplus h_{m'_i}(K \oplus \overline{m_{i-1} \| i}),\ i = 1, 2, \ldots, t.$$

(4) Compute the last pseudo-message block, m'_{t+1} as follows:

$$MD = h_{K_p}(m'_1 \| \ldots \| m'_t \| h_{IV}(K_p)),\ m'_{t+1} = K\{\overline{MD}\}.$$

(5) Output $(m' \| h_{MD}(m'_{t+1}))$.

It is claimed in [3] that their schemes prevent the attacks using message modification technique. It is also claimed that the pre-image and second pre-image security bound of the schemes is $O(2^{n/2+lt})$, where n denotes the output length, l denotes the length of each message block and t denotes the number of input message blocks. We show that the above claims are not correct. In the following of this paper, we construct some attacks on these schemes and give the new security bound of these schemes.

2.2 Security Analysis of All-or-Nothing Schemes as Hash Functions

Our attacks on these schemes are based on the multi-collision attack proposed by Joux [4]. Assuming there is an algorithm to find a collision for the compression function of the target hash functions, that Is, given as input a chain value H, the algorithm can output two different message blocks m and m' such that $f(H, m) = f(H, m')$. To illustrate the idea, we use an example in [4] to show how four collisions can be obtained using this algorithm. We first call the algorithm on initial value-IV to obtain two different randomly selected blocks, m_0 and m'_0 that yield a collision, *i.e.* $f(IV, m_0) = f(IV, m'_0) = H_1$, where H_1 denotes the chain value, and then use the algorithm on H_1 to find two other random blocks m_1 and m'_1 such that $f(H_1, m_1) = f(H_1, m'_1)$. Then we have 4 collisions on the hash function as follows.

$$f(f(IV, m_0), m_1) = f(f(IV, m_0), m'_1) = f(f(IV, m'_0), m_1) = f(f(IV, m'_0), m'_1).$$

This algorithm can use the generic birthday attack and should be effective for any input value. It should be noted that the algorithm is effective even if the compression function is ideal *i.e.* random oracle. In this paper we use the multi-collision attack to construct attacks on the schemes in [3] and give the correct complexity for the pre-image attack on them.

Now we construct a pre-image attack on the first scheme. The attack can be described as follows:

1. Given the random key K, for each step

   $$H_i = f_{H_{i-1}}(m'_i) \ (i = 1, 2, \ldots, t)$$

   use the algorithm mentioned above to find a collision such as

   $$f_{H_{i-1}}(m'_i) = f_{H_{i-1}}(m''_i)$$

   where m'_i and m''_i are different pseudo messages, then for different pseudo message blocks, we have the same H_i. It is easy to see that after t steps, the algorithm is called for t times and there are 2^t pseudo messages which have the same H_t.
2. Use these pseudo message blocks found in step 1 to find the plain messages. Compute

   $$m_i = m'_i \oplus f(K, H_{i-1}, i)$$

   to get the plain message and Compute

   $$m'_{t+1} = m'_{t+1} \oplus g(K_p, m'_i, i)$$

to get the last pseudo message block. Then we will have 2^t messages that have the same H_t value and different 2^t pseudo messages

3. If t is equal to the output length of the hash function n, then hash the 2^n pseudo messages to find a pre-image in these plain messages.

Theorem 1 The complexity of pre-image attack for the first scheme is $O(2^n)$.

Proof: In step 1 of this attack, the complexity to find one collision for one step of the hash function is $O(2^{n/2})$, then after t steps, the complexity to find one collision for all steps is $O(t2^{n/2})$. Then there are 2^t pseudo messages that have the same output H_t (see Fig. 1). If t is equal to the output length n of the hash function, then there are 2^n pseudo messages. Using these pseudo messages, we can get the corresponding plain messages, even if the hash function is an ideal hash function, because there are 2^n pseudo messages *i.e.* 2^n last block, then the probability to find the pre-image of $H_{t+1} = h_{H_t}(m'_{t+1})$ is overwhelming, where m'_{t+1} is the last block of one pseudo message. In the attack, the hash function is used $t2^{n/2} + 2^n$ times, and because $2^n \gg 2^{n/2}$, the complexity to find its pre-image is $O(2^n)$. ❑

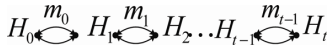


Fig. 1. Find multi-collisions.

The pre-image attack for the second scheme is similar to the first scheme and can be described as follows:

1. Given the random key K, for each hash function $h_{m'_i}(K_p \oplus i)$ (i = 1, 2, …, t), instead of fixing chain value as in the fist scheme, we fix the input message $K_p \oplus i$ and use the algorithm to find a free-start collision such as

   $$h_{m'_i}(K_p \oplus i) = h_{m''_i}(K_p \oplus i)$$

   where m'_i and m''_i are different pseudo messages. Then for different pseudo message blocks, we have the same value of $h_{m'_1} \oplus h_{m'_2} \oplus \ldots \oplus h_{m'_t}$. It is easy to see that after using the algorithm for t times, there are 2^t pseudo messages which have the same value.
2. use these pseudo message blocks found in step 1 to find the plain messages. Compute

   $$m_i = m'_i \oplus h_{m'_i}(K \oplus i)$$

   to get the plain messages, and we will get 2^t messages that have the same $h_{m'_1} \oplus h_{m'_2} \oplus \ldots \oplus h_{m'_t}$ value.
3. If t is equal to the output length of the hash function n, then there are 2^n pseudo messages to find a pre-image in these plain messages.

Theorem 2 The complexity of pre-image attack for the second scheme is $O(2^n)$.

Proof: In step 1 of this attack, the complexity to find one free-start collision for one hash function is $O(2^{n/2})$, which is the same as the collision attack. After t steps, the complexity

to find one free-start collision for each hash function is $O(t2^{n/2})$. Hence there are 2^t pseudo messages that have the same $h_{m_1'} \oplus h_{m_2'} \oplus \ldots \oplus h_{m_t'}$ value. If t is equal to the output length n of the hash function, then there are 2^n pseudo messages. Using these pseudo messages, we can get the corresponding plain messages. Assuming the hash function is an ideal hash function, because there are 2^n pseudo messages that have the same last block, the probability to find the pre-image of $h_{IV}(m')$ is overwhelming, where m' is the pseudo message . In the attack, the hash function is used for $t2^{n/2} + 2^n$ times, so the complexity to find its pre-image is $O(2^n)$. ❑

The attack for the third scheme can be described as follows:

1. Given the random key K, for the hash function

   $$h_{K_p}(m_1' \| \ldots \| m_t' \| h_{IV}(K_p)) \ (i = 1, 2, \ \ldots, t)$$

   use the algorithm to find a collision in each step such as

   $$h_{K_p}(m_i') = h_{K_p}(m_i'')$$

   where m_i' and m_i'' are different pseudo messages. Then for different pseudo message blocks, we have the same MD. It is easy to see that after using the algorithm for t times, there are 2^t pseudo messages which have the same MD.
2. Use these pseudo message blocks found in step 1 to get the plain messages by computing

   $$m_i = m_i' \oplus h_{m_{i-1}'}(K \oplus \overline{m_{i-1} \| i}).$$

   Then we will have 2^t messages that have the same MD value.
3. If t is equal to the output length of the hash function n, then there are 2^n messages to find the pre-image.

Theorem 3 The complexity of pre-image attack for the third scheme is $O(2^n)$.

Proof: In step 1 of this attack, the complexity to find one collision for one step of the hash function is $O(2^{n/2})$, then after t steps, the complexity to find one collision for all steps is $O(t2^{n/2})$. Then there are 2^t pseudo messages that have the same output MD (see Fig. 1). If t is equal to the output length n of the hash function, then there are 2^n pseudo messages. Using these pseudo messages, we can get the corresponding plain messages, even if the hash function is an ideal hash function, because there are 2^n pseudo messages that have the same MD, then the probability to find the pre-image of $h_{MD}(K \oplus \{\overline{MD}\})$ is overwhelming. In the attack, the compression function is used for $t2^{n/2} + 2^n$ times, so the complexity to find its pre-image is $O(2^n)$. ❑

It is easy to see that the complexity to find collisions for these schemes is $O(2^{n/2})$ if the compression function is ideal. It should be noted that for any hash function, $O(2^n)$ is an upper bound for pre-image attack and $O(2^{n/2})$ is an upper bound for collision attack, and the bound is reached when the hash function is an ideal random oracle. Our attacks

need the number of message blocks $t \geq n$, otherwise the attacks cannot be implemented.

From the analysis above, it is shown that the complexity analysis in [3] is not correct. These schemes prevent the particular message manipulation attack and may improve the complexity to find collisions and pre-images on the implemented hash functions but cannot improve the complexity over the bound for the ideal hash functions. The complexity to find collision and pre-image depends on the output length of the hash functions but not on the number of message blocks or the message block length. It is well known that the ideal hash functions cannot be implemented in practice. Therefore the complexity for any hash functions in practice is lower than the bound for the ideal hash functions.

2.3 Security of All-or-Nothing Schemes under Multi-Collision Attack

The purpose to construct the schemes in [3] is to prevent the attacks based on message modification, because most of the known attacks on hash functions use message manipulation. Multi-collision attack is also a kind of attack using message manipulation. Although the result in [4] is later than the one in [3] and the purpose of the schemes in [3] is not multi-collision resistant, the attacks in [4] cannot be applied to two schemes in [3]. For the first scheme and the second scheme in [3], multi-collision attack cannot be directly applied to them. However, the third scheme is not multi-collision attack resistant. Similar to the pre-image attack above, we construct an attack to find multi-collisions for this scheme. It is described as follows:

1. Given the random key K, for hash function

   $$h_{K_p}(m'_1 \| \ldots \| m'_t \| h_{IV}(K_p)), i = 1, 2, \ldots, t$$

   use the algorithm to find a collision such that

   $$h_{K_p}(m'_1) = h_{K_p}(m''_1)$$

   where m'_i and m''_i are different pseudo messages. Then for different pseudo message blocks, we have the same value of MD. It is easy to see that after using the algorithm for t times, there are 2^t pseudo messages which have the same MD.
2. use these pseudo message blocks found in step 1 to find the plain messages. Then we will have 2^t messages that have the same MD value.
3. Using these messages, we can obtain 2^t collisions on this scheme because the hash value is

   $$h_{MD}(m'_{t+1}), m'_{t+1} = K \oplus \{\overline{MD}\}.$$

Theorem 4 The complexity to find 2^t collisions for the third schemes is $O(t2^{n/2})$.

Proof: In step 1 of this attack, the complexity to find one collision for one step of the hash function is $O(2^{n/2})$, then after t steps, the complexity to find one collision for all steps is $O(t2^{n/2})$. There are 2^t pseudo messages that have the same output MD (see Fig. 1). Using these pseudo messages, we can get the corresponding plain messages, even if the hash function is an ideal hash function, because there are 2^t pseudo messages that have

the same *MD*, then the probability to find 2^t collisions of $h_{MD}(K \oplus \{\overline{MD}\})$ is overwhelming. Therefore the complexity to find 2^t collisions is $O(t2^n)$. ❑

To prevent this attack, we substitute $h_{MD}(m'_{t+1})$ with $h_{MD}(m')$. Then like the other two schemes, multi-collision attack also cannot be directly applied to this scheme. The authors of [3] claimed that the three schemes could resist message manipulation attack, but the third one cannot resist the multi-collision attack which is a kind of message manipulation attack. Actually, the schemes in [3] are not constructed to resist multi-collision attack, but we find the first and the second scheme can resist the attack. This idea is similar to the one in [16].

3. CONCLUSION

In this paper, we review the three schemes in [3]. It has been claimed that the pre-image and second pre-image security bound of the schemes is $O(2^{n/2+lt})$, where n denotes the output length, l denotes the length of each message block and t denotes the number of input message blocks. We have shown that the security bound of these schemes cannot exceed the ideal security bound and give the correct complexity bound. Although it has been claimed that these schemes prevent the attacks using message modification technique, we find one of the schemes cannot resist the multi-collision attack. Actually, the schemes in [3] provide a good idea to generate the MAC and encrypting the message simultaneously. However, the efficiency of these schemes is very low, because the hash functions have to process the messages twice. Therefore, constructing efficient schemes will be the further research.

REFERENCES

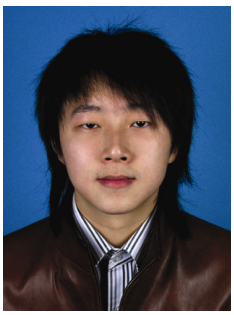
1. I. B. Damgård, "A design principle for hash functions," in *Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, LNCS 435, 1989, pp. 416-427.
2. R. Merkle, "One way hash functions and DES," in *Proceedings of Advance in Cryptology*, LNCS 435, 1989, pp. 428-446.
3. S. U. Shin, K. H. Rhee, and J. W. Yoon, "Hash functions and the MAC using all-or-nothing property," in *Proceedings of the 2nd International Workshop on Practice and Theory in Public Key Cryptography*, LNCS 1560, 1999, pp. 263-275.
4. A. Joux, "Multicollisions in iterated hash functions," *Application to Cascaded Constructions*, *Advances in Cryptology – CRYPTO*, LNCS 3152, 2000, pp. 306-316.
5. X. Wang, X. Lai, D. Feng, H. Chen, and X. Yu, "Cryptanalysis of the hash functions MD4 and RIPEMD," in *Proceedings of Advances in Cryptology – EUROCRYPT*, LNCS 3494, 2005, pp. 1-18.
6. X. Wang, H. Yu, and Y. Yin, "Efficient collision search attacks on SHA-0," in *Proceedings of Advances in Cryptology – Crypto*, LNCS 3621, 2005, pp. 1-16.
7. X. Wang, Y. Yin, and H. Yu, "Finding collisions in the full SHA-1," in *Proceedings of Advances in Cryptology – CRYPTO*, LNCS 3621, 2005, pp. 17-36.
8. X. Wang and H. Yu, "How to break MD5 and other hash functions," in *Proceedings*

of Advances in Cryptology – EUROCRYPT, LNCS 3494, 2005, pp. 19-35.
9. S. Lucks, "A failure-friendly design principle for hash functions," in *Proceedings of Advances in Cryptology – ASIACRYPT*, LNCS 3788, 2005, pp. 474-494.
10. S. Hirose, "Provably secure double-block-length hash functions in a black-box model," in *Proceedings of the 7th International Conference on Information Security and Cryptology*, LNCS 3506, 2005, pp. 330-342.
11. M. Nandi, W. Lee, K. Sakurai, and S. Lee, "Security analysis of a 2/3-rate double length compression function in black-box model," in *Proceedings of the 12th International Workshop on Fast Software Encryption*, LNCS 3557, 2005, pp. 243-254.
12. R. Rivest, "All-or-nothing encryption and the package transform," in *Proceedings of the 4th International Workshop on Fast Software Encryption*, LNCS 1267, 1997, pp. 210-218.
13. L. Knudsen and F. Muller, "Some attacks against a double length hash proposal," in *Proceedings of Advances in Cryptology – ASIACRYPT*, LNCS 3788, 2005, pp. 462-473.
14. J. Kelsey and B. Schneier, "Second preimages on *n*-bit hash functions for much less than 2*n* work," in *Proceedings of Advances in Cryptology – EUROCRYPT*, LNCS 3494, 2005, pp. 474-490.
15. J. Black, P. Rogaway, and T. Shrimpton, "Black-box analysis of the block-cipher based hash function constructions from PGV," in *Proceedings of Advances in Cryptology – CRYPTO*, LNCS 2442, 2002, pp. 320-335.
16. N. Kauer, T. Suarez, and Y. Zheng, "Enhancing the MD-strengthening and designing scalable families of one-way hash algorithms," http://eprint.iacr.org/2005/397.
17. X. Lai and J. Massey, "Hash functions based on block ciphers," in *Proceedings of Advances in Cryptology – EUROCRYPT*, LNCS 658, 1993, pp. 55-70.
18. R. L. Rivest, "The MD5 message-digest algorithm," RFC1321, Internet Activity Board, Internet Privacy Task Force, 1992.
19. FIPS 180-1, Secure Hash Standard, Federal Information Processing Standard, Publication 180-1, NIST, 1995.
20. FIPS 180-2, Secure Hash Standard, Federal Information Processing Standard, Publication180-2, NIST, 2003.
21. B. Preneel, R. Govaerts, and J. Vandewalle, "Hash functions based on block ciphers: a synthetic approach," in *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology*, LNCS 773, 1994, pp. 368-378.

Pin Lin is now a Ph.D. working at the State Key Laboratory of Information Security, Graduate University of Chinese Academy of Sciences. His current research interest is the design and analysis of hash functions.

Wenling Wu is now a professor at the State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences. She received her B.S. degree and M.S. degree in Maths from Northwest University in 1987 and 1990, respectively. She received her Ph.D. degree in Cryptography from Xidian University in 1997. From 1998 to 1999 she was a postdoctoral fellow in the Institute of Software, Chinese Academy of Science. Her current research interests include theory of cryptography, mode of operation, block cipher, stream cipher and hash function.

Chuankun Wu was teaching at Xidian University since January 1988. He was promoted by Xidian University as a Lecturer in 1990, an Associate professor in 1992, and a full professor in 1995. In September 1995, he became a postdoctoral fellow in Australia, then from 1997 a research fellow, and from 2000 a Lecturer in the Department of Computer Science, Australian National University. Since 2003, He has joined the Institute of Software, Chinese Academy of Sciences. He has got many awards while he was in China, including China Government Special Subsidy awarded in 1993. He founded and served as a program co-chair for 2001, 2002 and 2003 International Workshop on Cryptology and Network Security (CANS) which has become one of the influential international conferences since 2005, and has served as a program committee member for many international conferences. He is an associate editor of IEEE Communications Letters, a member of International Association of Cryptologic Research, and a senior member of IEEE since 2000.



Tian Qiu was born in 1979. He is a Ph.D. candidate working at the Institute of Software, the Chinese Academy of Sciences. His current research interests include semantic web service discovery.