Attacks on JH Hash Function

Yiyuan Luo and Xuejia Lai

luoyiyuan@gmail.com

Abstract. JH hash function is one of the five finalists in NIST SHA-3 competition. JH-s is based on a $2n$ bit permutation and the final output is truncated to s bits, where n is 512 and s can be 224,256,384 and 512. Lee and Hong proved that JH-s is optimal collision resistance without length padding to the last block.

In this paper we present significant collision and preimage attacks on JH-s without length padding to the last block. For collision and preimage attack, the adversary needs $2^{s/4+1}$ and $2^{s/2+1}$ queries to the underlying permutation respectively. Thus for JH-224, the attacker only needs 2^{57} compression function queries to mount a collision attack. Our attack exploits structure flaws in the design of JH. The attack is easily applied to JH's variant MJH.

1 Introduction

Cryptographic hash function is one of the most important primitives in cryptography [8]. A hash function maps from message of arbitrary length to a fixed length. A hash function usually consists of iteration of a compression function. One first designs a fixed domain compression function and then extends the domain to an arbitrary domain by iterating that function.

Since some popular hash functions such as MD5 [9] and SHA-1 [4] have been attacked [11,10], NIST has launched a competition for a new hash function standard SHA-3. JH is one of the five finalists in the SHA-3 competition [12]. As the sponge construction [1], JH's compression function uses a single large fixed $2n$-bit permutation, then the chopMD [3] domain extension is applied to the compression function. The hash value is the last s bits of the output of the last block compression function. In the design of JH, n is 512 and s can be 224,256,384 and 512. After JH, Lee and Stam proposed a variant MJH based on a an (n, n) blockcipher [7]. The hash value of MJH is $2n$ bits. In the compression function of JH, the message block is added to both the input and output of the permutation to get the chain value, while in Sponge compression function, the message block is only added to the input of the permutation.

Previous security results for JH and MJH. In the provable security literature, the underlying primitives are assume to be ideal, thus the fixed permutation is assumed to be an ideal permutation. The first provable security result for mode of JH is its indifferentiability[2]. Bhattacharyya *et al.* proved that JH-s is indifferentiable from a random oracle up to $O(2^{n/3})$ queries to the ideal permutation when $s \leq n$.

In [6], Lee and Hong proved that JH-s without length padding to the last block is collision resistance up to $O(2^{s/2})$ queries and claimed that JH-s is optimal collision resistance in the ideal permutation model when $s \leq n$.

In [7], the designers proved that MJH without length padding to the last block is collision resistance up to $O(2^{\frac{2n}{3}-\log n})$ queries.

In [5], Hong and Kwon make a collision attack with time complexity 2^{124} on MJH for $n = 128$ and the preimage attack with time complexity $2^{3n/2+2}$.

Our contribution. In this paper we present significant collision and preimage attacks on JH-s without length padding to the last block. Note that the collision resistant proof for JH-s is in this mode. For collision and preimage attack, the adversary needs $2^{s/4+1}$ and $2^{s/2+1}$ queries to the underlying permutation respectively. Thus for JH-224, the attacker only needs 2^{57} compression function queries to mount a collision attack. Our attack exploits structure flaws in the design of JH. We show that JH is weaker than Sponge construction since the message block is both added to the input and the output of the permutation.

The attack is easily extended to JH's variant MJH. For the $2n$-bit MJH, the adversary needs $3 \times 2^{n/2}$ queries to the blockcipher to find a collision and $3 \times 2^{n-1}$ queries to the blockcipher to find a (2nd) preimage.

2 Preliminaries

General Notation. For two bitsrings x and y, $x \parallel y$ denotes the concatenation of x and y. A blockcipher E with n-bit block and n-bit keysize is called an (n, n) blockcipher.

Information Theoretic Model. In the information theoretic model, the adversary is computationally unbounded but is given up to q queries to the underlying ideal primitive. The advantage of the adversary is related to the query times q. Almost every security proof in the hash function literature uses this model.

3 The JH hash function

Let F be a $2n$ bit permutation, the compression function of JH depicted in Fig. 1 is defined as:

$$f(h_{i-1}, g_{i-1}, m_i) = F(h_{i-1} \oplus m_i \parallel g_{i-1}) \oplus (0^n \parallel m_i)$$

where $h_{i-1}, g_{i-1}, m_i \in \{0,1\}^n$.

JH uses the chopMD mode of operation. The initial value is fixed as (IV_0, IV_1) and the message M is first padded into l message blocks, then the usual Merkle-Damgård iteration is applied to F to compute the last chain value (h_l, g_l). The final output is is the last s bits of g_l.

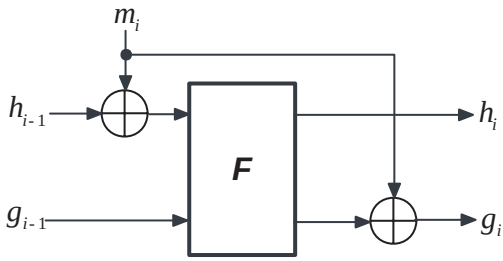

Fig. 1. The compression function of JH.

4 Attacks on the high level structure of JH

In this section we first give attacks on the high level structure of JH. The attack can be easily applied to the normal JH function. Let L be a linear transform on $2n$. The high level compression function of JH can be denoted as

$$h_i = F(h_{i-1} \oplus m_i) \oplus L(m_i)$$

where $h_{i-1}, h_i, m_i \in \{0,1\}^{2n}$. Let $chop_s$ be the last s bits of a $2n$-bit value and h_0 be a fixed initial value IV. The 2-block high level of JH can be depicted in Fig.2.

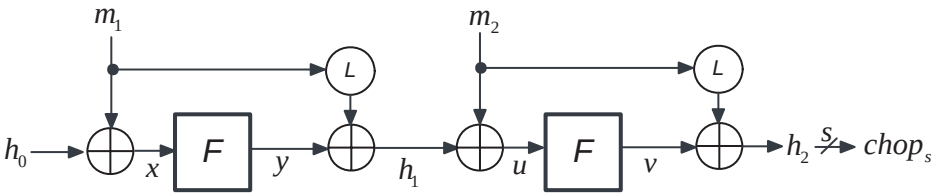

Fig. 2. The 2-block high level structure of JH.

4.1 Collision Attack

We assume F be an ideal permutation and the adversary never makes repeat queries. That is to say, the adversary never makes queries that she already knows the result. In the attack we use two blocks of message (m_1, m_2) and fix the initial value as $h_0 = IV$ where IV is a $2n$-bit constant. As shown in Fig.2, we let (x, y) be the input-output queried pairs in the first block query and (u, v) be the

input-output queried pairs in the second block query. Thus we have

$$\begin{aligned} m_1 &= h_0 \oplus x \\ h_1 &= L(m_1) \oplus y \\ m_2 &= h_1 \oplus u \\ h_2 &= L(m_2) \oplus v. \end{aligned}$$

The attack is described as follows:

1. Set m_1 to be a constant.
2. Choose r random distinct values of x and make queries to F, we thus get r distinct random pairs of (x, y). Since $h_1 = y \oplus L(m_1)$, we get r distinct random values of h_1.
3. Choose r random distinct values of u and make queries to F, we thus get r distinct random pairs of (u, v).
4. For each pair of (u, v), we can compute $m_2 = h_1 \oplus u$ and $h_2 = v \oplus L(m_2)$. Since there are r distinct values of h_1, we can get r random values of m_2 and h_2.
5. There are total r pairs of (u, v), thus we can get r^2 random values of h_2.
6. If the final hash value is truncated to s bits where $s \leq 2n$. Let r be $2^{s/4}$, thus we can get $r^2 = 2^{s/2}$ random values of h_2 and $chop_s$. According to the birthday paradox, there exists two pairs of (h_1, m_2) colliding at $chop_s(h_2)$ with probability 0.39.
7. The adversary needs $2 \times 2^{s/4} = 2^{s/4+1}$ queries to the permutation F to find a collision with probability 0.39.

4.2 Preimage and Second Preimage Attack

For preimage and second preimage attack, we need to find a preimage for a s-bit value. It is easy to see that if we let $r = 2^{s/2}$, at last we can get $r^2 = 2^s$ random values of $chop_s$, since $chop_s$ is s bits, with high probability we can find a (second) perimage.

Thus the adversary needs $2 \times 2^{s/2} = 2^{s/2+1}$ queries to the permutation F to find a preimage with high probability.

5 Attacks on 2-block JH

The 2-block JH-s hash function is shown in Fig.3. Let (h_0, g_0) be a fixed initial value (IV_0, IV_1). From the figure, we have

$$\begin{aligned} (h_1, g_1) &= F(h_0 \oplus m_1 \parallel g_0) \oplus (0^n \parallel m_1) \\ (h_2, g_2) &= F(h_1 \oplus m_2 \parallel g_1) \oplus (0^n \parallel m_2). \end{aligned}$$

As in the figure, the output of a query (x, g_0) to the first block F is denoted as (h_1, y), and the output of a query (u, g_1) to the second block F is denoted as (h_2, v). The final output of JH-s is the last s bits of g_2 and denoted as $chop_s$.

The collision and (2nd) preimage attack is described as follows:

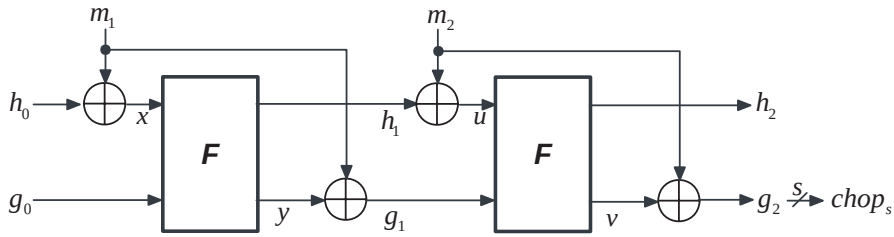

Fig. 3. The 2-block structure of JH-s. All wires carry n-bit values.

– Collision attack:
  1. $h_0 \parallel g_0$ is fixed to the initial value. Set m_1 to be a constant.
  2. Choose r random distinct values of x and make queries $x \parallel g_0$ to F, we thus get r distinct random values of $h_1 \parallel y$. Since $g_1 = y \oplus m_1$, we get r distinct random values $(h_1^i \parallel g_1^i), 1 \leq i \leq r$.
  3. Choose r random distinct values $u^i, 1 \leq i \leq r$, make queries $(u^i \parallel g_1^i), 1 \leq i \leq r$ to F, we thus get r distinct random values $h_2^i \parallel v^i), 1 \leq i \leq r$.
  4. For each pair of $(u^i \parallel g_1^i, h_2^i \parallel v^i), 1 \leq i \leq r$, we can compute $m_2 = h_1 \oplus u^i$ and $g_2 = v^i \oplus m_2$. Since there are r random values of h_1, we can get r random values of m_2 and g_2.
  5. There are total r pairs of $(u^i \parallel g_1^i, h_2^i \parallel v^i)$, thus we can get r^2 random values of g_2.
  6. If the final hash value is truncated to s bits where $s \leq n$. Let r be $2^{s/4}$, thus we can get $r^2 = 2^{s/2}$ random values of h_2 and $chop_s$. According to the birthday paradox, there exists two pairs colliding at $chop_s$ with probability 0.39.
  7. The adversary needs $2 \times 2^{s/4} = 2^{s/4+1}$ queries to the permutation F to find a collision with probability 0.39.
– Preimage attack: For preimage and second preimage attack, we need to find a preimage for a s-bit value. It is easy to see that if we let $r = 2^{s/2}$, we can get $r^2 = 2^s$ random values of $chop_s$, since $chop_s(h_2)$ is only s bits, with probability close to 1 we can find a (second) perimage.
  Thus the adversary needs $2 \times 2^{s/2} = 2^{s/2+1}$ queries to the permutation F to find a (2nd) preimage with probability close to 1.

6 Attacks on JH's variant MJH

MJH hash function is proposed by Lee and Stam [Lee2011]. It is a variant of JH hash function. It uses two calls to a (n, n)-bit blockcipher E to implement the underlying primitive F, while F needn't to be a permutation. Let σ be an involution on $\{0,1\}^n$ with no fixed point, and let $\theta \neq 0, 1$ be a constant in $\mathbb{F}_{2^n}$,

the primitive F is defined as

$$\begin{aligned} F[\sigma,\theta] : \{0,1\}^{2n} &\longrightarrow \{0,1\}^{2n} \\ (x_L \parallel x_R) &\longrightarrow (y_L \parallel y_R) \\ y_L &= E_{x_R}(x_L) \oplus x_L \\ y_R &= \theta \cdot (E_{x_R}(\sigma(x_L)) \oplus \sigma(x_L)) \oplus x_L. \end{aligned}$$

By applying the JH transform, the compression function of MJH is the same as in Fig. 1. Then MJH uses Merkle-Damgård mode without length padding to the last block to calculate the final $2n$-bit hash value.

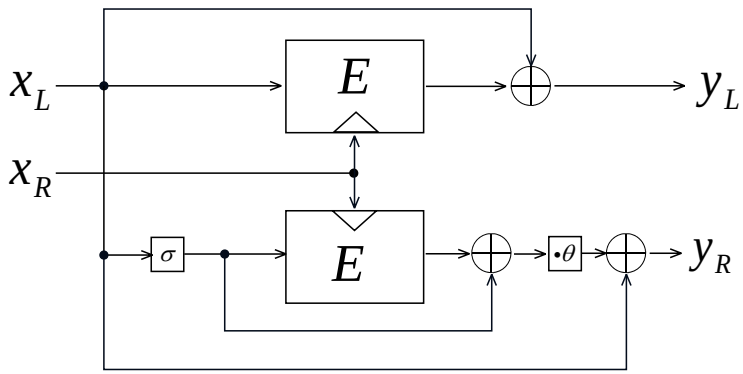

Fig. 4. The F primitive in MJH, where $(y_L, y_R) = F(x_L, x_R)$. All wires carry n-bit values. E is an (n, n) blockcipher. σ is an involution and θ is a constant in $\mathbb{F}_{2^n} \setminus \mathbb{F}_2$.

Due to the involution property, the adversary can get a pair of $(x_L \parallel x_R, y_L \parallel y_R)$ when she makes a query $E_{x_R}(x_L)$ to the upper blockcipher and a query $E_{x_R}(\sigma(x_L))$ to the lower blockcipher. That is to say, for each query $x_L \parallel x_R$ to the primitive F, the adversary can get two pairs of $(x_L \parallel x_R, y_L \parallel y_R)$ by making two blockcipher queries.

Since MJH outputs $2n$ bits as the hash value, thus the collision attack and (2nd) preimage attack is a little different from the attack on JH. The attack is similar as in Fig. 3 and described as follows.

– Collision attack:
  1. $h_0 \parallel g_0$ is fixed to the initial value. Set m_1 to be a constant.
  2. Choose $r = 2^{n/2-1}$ random distinct values of x and make queries $x \parallel g_0$ to F, we thus get $2^{n/2}$ random values of $h_1 \parallel y$ by $2^{n/2}$ queries to the blockcipher. Since $g_1 = y \oplus m_1$, we get $2^{n/2}$ random values $(h_1^i \parallel g_1^i), 1 \leq i \leq r$.

3. Choose $2^{n/2}$ random distinct values $u^i, 1 \leq i \leq 2^{n/2}$, make queries $(u^i \parallel g_1^i), 1 \leq i \leq 2^{n/2}$ to F, we thus get $2^{n/2+1}$ random values $h_2^i \parallel v^i, 1 \leq i \leq 2^{n/2+1}$ by $2^{n/2+1}$ blockcipher queries. Due to birthday paradox, with probability $1 - e^{-\frac{(2^{n/2+1})^2}{2 \times 2^n}} \approx 0.86$ we can obtain a pair $(u^i \parallel g_1^i, u^j \parallel g_1^j), 1 \leq i < j \leq 2^{n/2+1}$ colliding at h_2.
4. For the value $u^i \parallel g_1^i$, we can compute $m_2 = h_1 \oplus u^i$ and $g_2 = v^i \oplus m_2$. Since there are $2^{n/2}$ random values of h_1, we can get $2^{n/2}$ random values of m_2 and g_2. For the value $u^j \parallel g_1^j$, we can also get $2^{n/2}$ random values of m_2 and g_2. Thus with probability $1 - e^{-\frac{1}{2}} \approx 0.39$ a match will be found for these two sets.
5. The adversary needs $3 \times 2^{n/2}$ queries to the blockcipher to find a collision with probability $0.86 \times 0.39 \approx 0.34$.

– Preimage attack: For preimage and second preimage attack, we need to find a preimage for a $2n$-bit value. It is easy to see that if we let $r = 2^{n/2-1}$, after find a hitting at h_2, we can get 2^n random values of g_2. Since g_2 is only n bits, with probability close to 1 we can find a (second) perimage.
Thus the adversary needs $3 \times 2^{n-1}$ queries to the blockcipher to find a (2nd) preimage with probability close to 1.

7 Conclusion

In this paper we have presented collision and preimage attacks on JH-s. For collision and preimage attack, the adversary needs $2^{s/4+1}$ and $2^{s/2+1}$ queries to the underlying permutation respectively. Though our attack fails if the length is padded to the last message block or it uses the first s bits of the output as the hash value, the attack exploits structure flaws in the design of JH. JH it is weaker than Sponge since the message block is both added to the input and the output of the permutation. The attack is easily extended to JH's variant MJH. Through our analysis, the security of JH-s is at most n bits when $s \leq 2n$. It shows that previous security proofs of JH and MJH are flawed.

References

1. G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche. On the indifferentiability of the sponge construction. In *Advances in Cryptology - EUROCRYPT'08*, volume LNCS 4965, pages 181–197, Istanbul, Turkey, 2008. Springer-Verlag.
2. Rishiraj Bhattacharyya, Avradip Mandal, and Mridul Nandi. Security analysis of the mode of JH hash function. In *FSE 2010*, volume LNCS 6147, pages 168–191. Springer-Verlag, 2010.
3. J. S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-damgård revisited: How to construct a hash function. In *Advances in Cryptology - CRYPTO'05*, volume LNCS 3621, pages 430–448. Springer-Verlag, 2005.
4. FIPS. FIPS 180-1 Secure Hash Standard. Federal Information Processing Standard (FIPS), Publication 180-1, National Institute of Standards and Technology, US Department of Commerce, Washington D.C, 1995.

5. Deukjo Hong and Kwon. Cryptanalysis of some double-block-length hash modes of block ciphers with n-bit block and n-bit key. http://eprint.iacr.org/2013/174, 2013.
6. Jooyoung Lee and Deukjo Hong. Collision resistance of the JH hash function. *IEEE Transaction on Information Theory*, 58(3):1992–1995, 2012.
7. Jooyoung Lee and Martijn Stam. MJH: A faster alternative to MDC-2. In *CT-RSA 2011*, volume LNCS 6558, pages 213–236. Springer-Verlag, 2011.
8. A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.
9. R. L. Rivest. The MD5 message digest algorithm. In *Request for Comments (RFC) 1321*. Internet Activities Board, Internet Privacy Task Force, 1992.
10. Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full SHA-1. In Victor Shoup, editor, *Advances in Cryptology - CRPTO'05*, volume LNCS 3621, pages 17–36, Santa Barbara, CA, USA, 2005. Springer-Verlag.
11. Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT'05*, volume LNCS 3494, pages 19–35, Aarhus, Denmark, 2005. Springer-Verlag.
12. Hongjun Wu. The hash function JH. http://www3.ntu.edu.sg/home/wuhj/research/jh/jh-round3.pdf, 2011.