Cryptanalysis of the LANE Hash Function

Shuang Wu, Dengguo Feng, and Wenling Wu

State Key Lab of Information Security, Institute of Software
Chinese Academy of Sciences
Beijing 100190, China
{wushuang,feng,wwl}@is.iscas.ac.cn

Abstract. The LANE[4] hash function is designed by Sebastiaan Indesteege and Bart Preneel. It is now a first round candidate of NIST's SHA-3 competition. The LANE hash function contains four concrete designs with different digest length of 224, 256, 384 and 512.

The LANE hash function uses two permutations P and Q, which consist of different number of AES[1]-like rounds. LANE-224/256 uses 6-round P and 3-round Q. LANE-384/512 uses 8-round P and 4-round Q. We will use LANE-n-(a,b) to denote a variant of LANE with a-round P, b-round Q and a digest length n.

We have found a semi-free start collision attack on reduced-round LANE-256-(3,3) with complexity of 2^{62} compression function evaluations and 2^{69} memory. This technique can be applied to LANE-512-(3,4) to get a semi-free start collision attack with the same complexity of 2^{62} and 2^{69} memory. We also propose a collision attack on LANE-512-(3,4) with complexity of 2^{94} and 2^{133} memory.

Keywords: hash function, collision attack, rebound attack, LANE, SHA-3 candidates.

1 Introduction

The SHA-3 competition hosted by NIST aims to find a new cryptographic hash standard as a replacement of SHA-2. 51 of the 64 submitted designs are accepted to entered the first round. The LANE hash function is one of the first round candidates.

The attacks on widely used hash standards such as MD5[2] and SHA-1[3] are based on differential analysis. Many of the first round candidates of SHA-3 competition use AES[1]-like SPN structures and claim to resist differential attacks.

Florian Mendel et al. have proposed a new tool of "Rebound"[5] attack for cryptanalysis of AES-based designs. The main idea of rebound attack is to take advantage of weakness implied by S-box's optimal non-linearity. Random input and output differences of an S-box match with surprisingly high probability of 1/2 and at least two values can be selected for each S-box. The complexity of one round in the traditional truncated differential path can be totally eliminated

M.J. Jacobson Jr., V. Rijmen, and R. Safavi-Naini (Eds.): SAC 2009, LNCS 5867, pp. 126–140, 2009.

at the cost of exhausting degrees of freedom of the active state values. In this paper, we analyze reduced LANE with rebound techniques. There are other parallel works of improved rebound techniques and applications in [7,8,9].

This paper is organized as follows. In section 2, we briefly describe the LANE hash function. In section 3, we discuss inner collisions of only two lanes in the first layer P. Then semi-free start collision attacks on reduced LANE-256 are described in section 4. Section 5 describes the attacks on reduced LANE-512. Section 6 is the conclusion.

2 Description of LANE Hash Function

The LANE hash function uses iterative MD structure with counters and output transformation. Digest values of LANE-224 and LANE-384 are truncated from LANE-256 and LANE-512 separately. Details of padding rules and output transformation are omitted here since they do not influence this attack.

In this section we briefly describe the compression function of LANE-256 and LANE-512. LANE-256/512 is an iterative hash function, whose compression function $f(H_{i-1}, M_i, C_i)$ processes a 512/1024-bit message block, a 512/1024-bit chaining value, a 64-bit counter, and outputs 256/512-bit digest length. The chaining state H_{i-1} and the message block M_i are expanded to six 256/512-bit blocks. Each block enters a different lane of P. Different lanes use different constants and counters. Output of the first three lanes and the last three lanes are XORed separately as input of the Q permutations. At last, output of both Q permutations are XORed as the next chaining value H_i. The structure of compression function in LANE is shown in Figure 1.

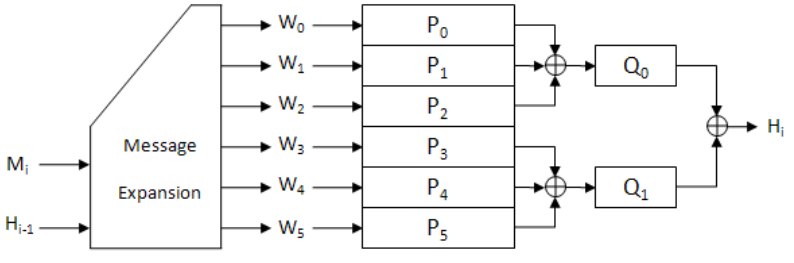

Fig. 1. The compression function of LANE

2.1 Message Expansion

The compression function of LANE-256 expands chaining value $H_{i-1} = h_0||h_1$ and message block $M_i = m_0||m_1||m_2||m_3$ to six 256-bit blocks $W_0, ...W_5$ as shown in equation 1.

$$\begin{array}{ll} W_0 = h_0 \oplus m_0 \oplus m_1 \oplus m_2 \oplus m_3 & \| \; h_1 \oplus m_0 \oplus m_2 \\ W_1 = h_0 \oplus h_1 \oplus m_0 \oplus m_2 \oplus m_3 & \| \; h_0 \oplus m_1 \oplus m_2 \\ W_2 = h_0 \oplus h_1 \oplus m_0 \oplus m_1 \oplus m_2 & \| \; h_0 \oplus m_0 \oplus m_3 \\ W_3 = h_0 & \| \; h_1 \\ W_4 = m_0 & \| \; m_1 \\ W_5 = m_2 & \| \; m_3 \end{array} \quad (1)$$

The message expansion in LANE-512 is analogous. The only difference is that all blocks are double-sized.

2.2 Permutations P and Q

P and Q in Figure 1 are permutations with AES-like state update rounds. LANE-256 uses 6-round P, 3-round Q and LANE-512 uses 8-round P and 4-round Q. Each round contains five steps. One round of state update operation in LANE-256 is shown in Figure 2. The difference in LANE-512 is that all operations are on four 4×4 matrices.

In this paper, we use S_i to denote the state value after the i-th round. Between S_i and S_{i+1}, the state values are denoted as S_i', S_i'', S_i''' and S_i'''' consecutively. $\triangle S_i$ is used to denote the XOR difference of state S_i.

The five steps of one round are:

- **SB**: the non-linear operation **SubBytes** applies an S-Box to each byte of the state. The S-box is the same as the one used in AES[1].
- **SR**: the cyclical permutation **ShiftRows** rotates the bytes of the i-th row leftwards by i positions.
- **MC**: the diffusion layer **MixColumns** multiplies each column by a MDS matrix which is the same as the one in AES.
- **AC**: the constants and counter additions **AddConstants** and **AddCounter** add the round constants and the counter to the states. We use **AC** to denote both of them. The last rounds of both P and Q don't have AC operations. Details of the AC operations are omitted here since they have nothing to do with our attacks.
- **SC**: the mixing operation between different 4×4 states **SwapColumns** reorders the columns in the state. LANE-256 and LANE-512 use different SwapColumns operations which are shown in Figure 3.

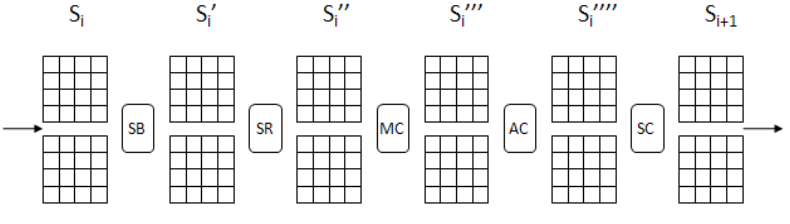

Fig. 2. One round of state update operation in LANE-256

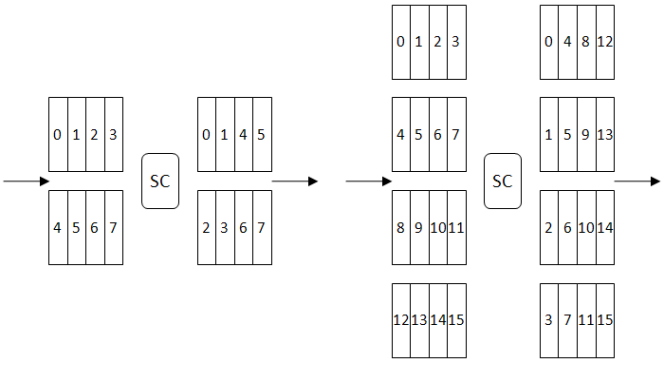

Fig. 3. SwapColumns operations used in LANE-256 and LANE-512

3 Construct Inner Collisions Using Rebound Techniques

The LANE hash function has six lanes in the first layer P. In this section, we are trying to construct collisions between only two lanes, namely the inner collisions. It's easy to see that two simultaneous inner collisions could directly lead to a full collision.

3.1 Optimal Differential Pattern for LANE

The message expansion used in LANE is based on a linear (6,3,4)-code over GF_4, which means for any possible differential path, there are at least four active lanes in the first layer P. Once the difference enters layer Q, there would be more active S-Boxes. So we want to eliminate all differences before they enter layer Q.

This is the best differential pattern for LANE with four active lanes P_1, P_2, P_4 and P_5. Two inner collisions in layer P ensure no difference enter layer Q as shown in Figure 4.

Let $\triangle m_0 = \triangle m_2 \neq 0$ and $\triangle h_0 = \triangle h_1 = \triangle m_1 = \triangle m_3 = 0$, we have four active lanes P_1, P_2, P_4 and P_5 and the differences in W are in the form of $(\triangle, 0)$ and $(0, \triangle)$. Differential paths with initial difference of $(\triangle, 0)$ and $(0, \triangle)$ behave

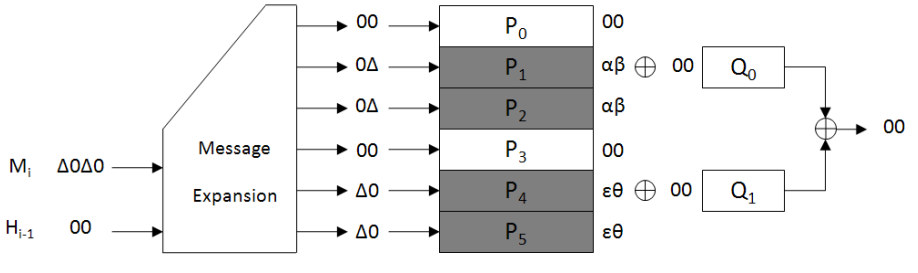

Fig. 4. Optimal differential pattern for LANE

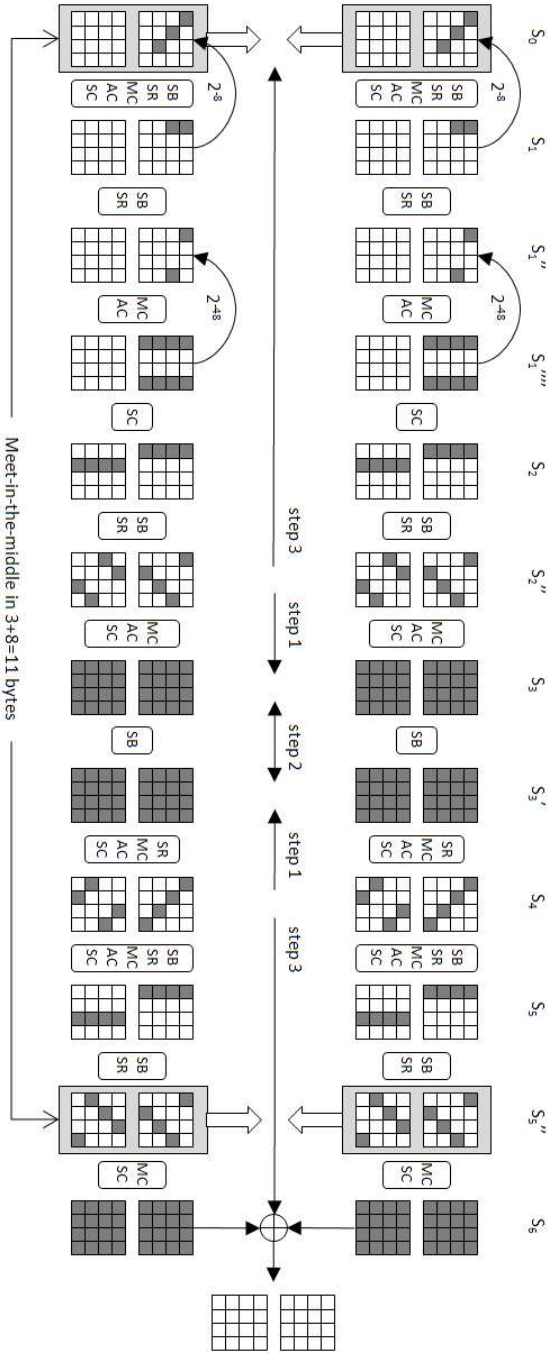

Fig. 5. Rebound differential path of an inner collision for LANE-256

in a similar way. So we only need to consider one type of differential path in the final attack. We will talk about this in section 4.2.

3.2 Rebound Differential Path of Inner Collision

In this section, we only consider an inner collision of two lanes. Using rebound techniques proposed by Florian Mendel et al. in [5], we can easily attack round 6 of layer P in LANE-256 with a complexity of 2^{100}. The differential path is shown in Figure 5.

In traditional truncated differential path, difference propagates from initial state to hash value in forward direction. In a rebound attack, we search for an inbound differential path in internal states first. Then the outbound part can be considered as two truncated differential paths in different directions - forward and backward. Since complexity of the inbound phase can be eliminated, we only need to consider probability of the outbound phase.

Here, we briefly describe the attack of inner collision. This is similar to the attack on Grøstl[5]. For more details of rebound attack, please refer to the original paper.

Step 1: We start from choosing random differences in both S_2'' and S_4. Then compute $\triangle S_3$ from $\triangle S_2''$ and $\triangle S_3'$ from $\triangle S_4$. These difference propagations $\triangle S_2' \rightarrow \triangle S_3$ and $\triangle S_3' \leftarrow \triangle S_4$ hold with probability of 1 because all operations between them are linear transformations SR, MC, AC and SC.

Step 2: We expect to find a match of possible differential character at the S-box in the third round with probability of 2^{-32}, because random difference in input and output of an S-Box matches with probability of $1/2$ and there are 32 active S-boxes. Once we have found a match, we get 2^{32} staring points (attempts) for the outbound phase, since we have at least two values for each S-Box match. So we can generate 2^{32} attempts with complexity of 2^{32}. For any $x \geq 32$ we can generate 2^x attempts with complexity of 2^x.

Step 3: Each starting point (attempt) can lead to our demanded differential pattern in S_0 with a probability of $2^{-48} \times 2^{-8} = 2^{-56}$. In other words, we can generate a successful attempt in one lane with complexity of 2^{56}.

Step 4: In order to find a match in the three bytes of difference in S_0 and eight bytes in S_5'' between two lanes, we need $2^{8\times(3+8)/2} = 2^{44}$ successful attempts in both lanes. So the complexity is $2^{56} \times 2^{44} = 2^{100}$ for a inner collision with the same initial difference in both lanes. The memory requirements of step 4 is $2 \times 2^{44} = 2^{45}$.

4 Semi-free Start Collision Attack on LANE-256-(3,3)

Even if we have successfully found two inner collisions of four lanes in layer P, we can not get a collision of full LANE. The problem is the message expansion since rebound attack require a full control of the state values. Four initial state values of the two inner collisions will probably lead to a contradiction since we

have a degree of freedom for only three states, namely (h_0, h_1), (m_0, m_1) and (m_2, m_3).

More precisely, from two inner collisions we get the exact values of W_1, W_2, W_4 and W_5. Recall equation (1), and we can see that W_4 and W_5 can determine values of m_0, m_1, m_2 and m_3. By selecting the values of h_0 and h_1, we can change the value of $(h_0 \oplus h_1 \oplus m_0 \oplus m_2 \oplus m_3, h_0 \oplus m_1 \oplus m_2)$ to W_1 which we have got from the first inner collision. Since all degrees of freedom are used, we have to leave W_2 satisfied by chance.

There are 256 bits left in W_2 along with the 24-bit initial difference. We need $2^{(256+24)/2} = 2^{140}$ inner collisions in both P_1, P_2 and P_4, P_5 to find a match in $256 + 24 = 280$ bits. So in both lanes of one inner collision, we need $2^{140/2} = 2^{70}$ times more attempts. The complexity of semi-free start collision attack on LANE-256 is $2^{100} \times 2^{70} = 2^{170} > 2^{128}$ which exceeds the birthday bound of 256-bit hash functions and this attack fails.

4.1 Rebound Differential Path with Partially Fixed State Values

We are inspired by Dmitry Khovratovich et al. of their meet-in-the-middle attacks on several SHA-3 candidates[6]. The idea is to fix values of certain bits to get an actually smaller size in the meet-in-the-middle part of the target state and lower the complexity.

If we fix some bytes in an AES state, they would be affected by other bytes in at most two rounds. We have got an observation that diffusion in LANE is not as efficient as in AES. Fixed values in certain positions of the initial state can proceed to the third round in both LANE-256 and LANE-512.

Combining this small observation and rebound techniques, we have found a solution for LANE-256-(3,3) as shown in Figure 6.

In this figure, one byte with a mark of "X" means its value can be pre-computed and fixed during the attack. In our attack, we let all the X bytes in S_0 to be zeros and calculate values of the following ones. When we choose differences in S_1' and S_2'', we also set the values of fixed bytes in S_1' and S_2'' to what we have pre-computed.

In the four active lanes of this attack, the round constants and counters are different. So the exact values of fixed bytes in S_1' and S_2'' are different in four lanes. But they would all lead to zero values in the certain positions of initial states.

We also let the values of marked bytes in h_0 and h_1 to be zeros. So when we have got W_1, W_2, W_4 and W_5 from two inner collisions, we calculate the values of the non-zero bytes of h_0, h_1, m_0, m_1, m_2 and m_3 from W_1, W_4 and W_5. Then there are only 128 bits of state values left unsatisfied in W_2 instead of 256 bits, since all zero bytes are already satisfied in advance.

4.2 Details of the Attack

In this attack, We use two inner collision differential paths with initial differences of the patterns $(\triangle, 0)$ and $(0, \triangle)$ separately. If we change the position of two 4×4 matrices in the initial state of one path, the differential path don't change

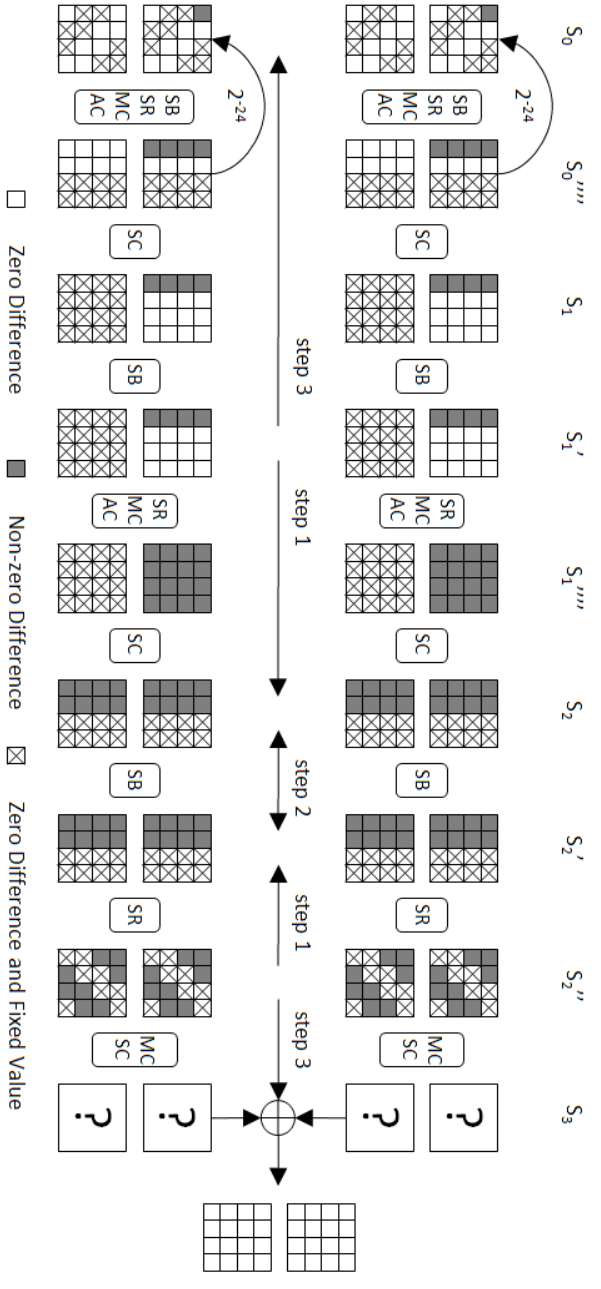

Fig. 6. Rebound differential path for LANE-256-(3,3)

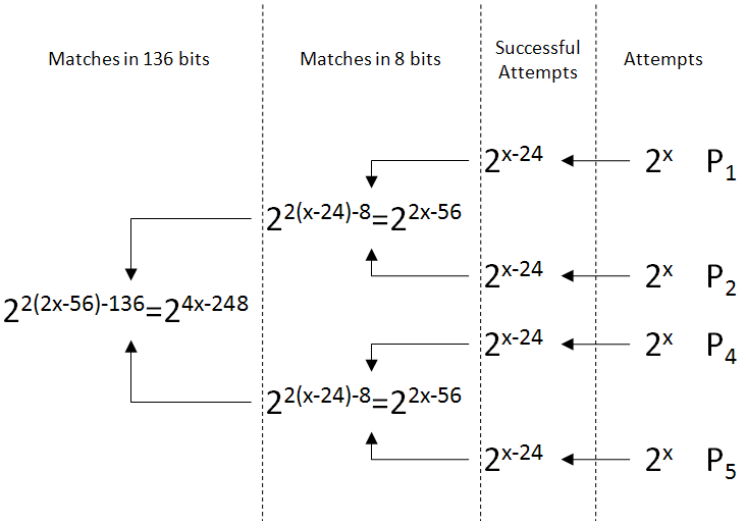

Fig. 7. Outline of the attack on LANE-256-(3,3)

substantially. Especially, the positions of fixed bytes don't change. So we consider these two differential paths equivalent and only need to analyze one of them in the following steps. Figure 7 shows outline of this attack.

This attack is described in six steps. Step 1 is the pre-computation. Steps 2 to 4 are the details in one lane of an inner collision. Step 5 is the meet-in-the-middle step of the initial difference between two lanes. Step 6 is the meet-in-the-middle step of the state values and the difference byte between two inner collisions except for the X bytes.

Step 1: Set all fixed bytes in the initial states in four lanes to zeros and compute the consecutive exact values of all fixed bytes in the following states.

Step 2: Choose random differences in both S_1' and S_2''. Here is a little difference from the attack above. We choose differences in S_2'' to be the same in both lanes of one inner collision. These two differences will remain the same when they proceed to S_3 because of the linear transformations from S_2'' to S_3. Even though we don't know the exact value of $\triangle S_3$ in both lanes, they must be the same and will offset each other before they enter layer Q.

Step 3: We expect to find a match of possible differential character at the S-box in the second round with probability of 2^{-32} as in the attack above. Once we have found a match, we get 2^{32} staring points (attempts) for the outbound phase. Now assume that we generated 2^x attempts with complexity of 2^x.

Step 4: We can find a successful attempt in one lane in every 2^{24} attempts. With 2^x attempts, we expect to find 2^{x-24} successful ones.

Step 5: Now we have 2^{x-24} successful attempts in both lanes, so we can find $2^{2(x-24)-8} = 2^{2x-56}$ matches in the only one byte difference in S_0. So we have got 2^{2x-56} inner collisions in P_1,P_2 and the same number of inner collisions in P_4,P_5. This step requires $4 \times 2^{x-24}$ memory.

Step 6: After we select the values of h_0 and h_1, there are 128 bits in W_2 and 8 bits in the initial difference unsatisfied. So we expect $2^{2(2x-56)-136} = 2^{4x-248}$ matches in these 136 bits. This step requires $2 \times 2^{2x-56}$ memory.

If $x = 62$, we expect to find a final match. Memory requirements of step 5 and step 6 are 2^{40} and 2^{69}. So the semi-free start collision attack on LANE-256-(3,3) has an overall complexity of 2^{62} and requires about 2^{69} memory.

5 Applications to LANE-512

We can also use rebound techniques to find inner collisions for LANE-512. By fixing certain bytes in the state values, we can find semi-free collision and collision attacks on LANE-512-(3,4).

5.1 Inner Collision of LANE-512

For LANE-512, we can proceed to round 8 of P in an inner collision attack of two lanes. The differential path will be shown in Figure 8 as an appendix. Details of this attack is similar to inner collision attack on LANE-256 in section 3.2.

For any given attempt, it is successful with probability of $2^{-24} \times 2^{-96} = 2^{-120}$, which means we can generate one successful attempt with complexity of 2^{120}. Then we have to match in $8 \times (16 + 16) = 256$ bits, and we need $2^{256/2} = 2^{128}$ successful attempts in both lanes. The complexity of inner collision on LANE-512 is $2^{120} \times 2^{128} = 2^{248}$ and the memory requirement of meet-in-the-middle step is $2 \times 2^{128} = 2^{129}$.

5.2 Semi-free Start Collision Attack and Collision Attack on LANE-512-(3,4)

Using the fixed bytes techniques, we can find a semi-free start collision of reduced LANE-512-(3,4) with a differential path shown in Figure 9 as an appendix. This attack is almost the same as the attack in section 4.2 with the same complexity of 2^{62} and 2^{69} memory.

As you can see in Figure 9, we can fix more bytes in the 3-round path for LANE-512. If we don't use the degrees of freedom in the initial chaining values h_0 and h_1, we have 16 more bytes in the final meet-in-the-middle part. The difference is that fixed bytes in W_1 and W_2 are not set to zeros in the marked positions. Recall equation 1, since now values of h_0 and h_1 are fixed, if we set fixed bytes of W_4 and W_5 to zeros, values of W_1 and W_2 in the marked positions are determined by the value of standard $IV = (h_0, h_1)$.

Assume that we have generated 2^x attempts in each lanes, we expect 2^{2x-56} inner collisions in both P_1, P_2 and P_4, P_5. The difference is now we have to match

$256 + 8 = 264$ bits. So we expect $2^{2\times(2x-56)-264} = 2^{4x-376}$ final matches with memory requirement of $2 \times 2^{2x-56}$. If $x = 94$, we expect to find one final match.

So, we have found a collision attack on LANE-512-(3,4). The complexity of collision attack on LANE-512-(3,4) is 2^{94} and memory requirement is 2^{133}.

5.3 Semi-free Start Collision Attack on LANE-512-(4,4)

If we want to attack more than three round in P, we can no longer use fixed values, since fixed values can only proceed to the third round. Without fixed values, we can attack LANE-512-(4,4) with a differential path shown in Figure 10 which is part of the one shown in Figure 8.

Assume that we have generated 2^x attempts in each lanes, and only 2^{x-120} of them will be successful ones. Then we expect $2^{2(x-120)-8} = 2^{2x-248}$ inner collisions in both P_1, P_2 and P_4, P_5. Here, we have to match $512 + 8 = 520$ bits. So we expect $2^{2\times(2x-248)-520} = 2^{4x-1016}$ final matches with memory requirement of $2 \times 2^{2x-248}$. If $x = 254$, we expect to find one final match.

So, the complexity of semi-free start collision attack on LANE-512-(4,4) is 2^{254} and memory requirement is 2^{261}. Though computational complexity is less than birthday bound, memory requirement of this attack is more than 2^{256}. This attack can be considered unsuccessful.

6 Conclusion

In this paper, we analyzed the LANE hash function using rebound and meet-in-the-middle techniques. We give several attacks on reduced variants of LANE-256 and LANE-512. Table 1 shows all the results of these attacks. Notation " † " in this table means the attack can be considered unsuccessful.

The memory requirements of all these attacks come from the meet-in-the-middle steps. But the memoryless variants seem not easy to be implemented in our attacks.

We can hardly attack more than three rounds of P with method of fixing certain bytes, since the fixed values can only proceed to the third round. Our attacks on reduced variants do not hurt collision resistance of full LANE.

Acknowledgments. The authors would like to thank the anonymous referees for their valuable comments. Furthermore, this work is supported by the National

Table 1. Results of collision attacks in this paper

hash function	P rounds	Q rounds	collision type	complexity	memory
LANE-256	6	-	inner collision of P	2^{100}	2^{45}
	3	3	semi-free start collision	2^{62}	2^{69}
LANE-512	8	-	inner collision of P	2^{248}	2^{129}
	3	4	semi-free start collision	2^{62}	2^{69}
			collision	2^{94}	2^{133}
	4	4	semi-free start collision	2^{254}	2^{261} †

High-Tech Research and Development 863 Plan of China (No. 2007AA01Z470), the National Natural Science Foundation of China (No. 60873259), and the National Grand Fundamental Research 973 Program of China (No. 2004CB318004).

References

1. National Institute of Standards and Technology: FIPS PUB 197, Advanced Encryption Standard (AES). Federal Information Processing Standards Publication 197, U.S. Department of Commerce (November 2001)
2. Wang, X., Yu, H.: How to break MD5 and other hash functions. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)
3. De Cannière, C., Rechberger, C.: Finding SHA-1 characteristics: General results and applications. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 1–20. Springer, Heidelberg (2006)
4. Indesteege, S., Preneel, B.: The LANE hash function, `http://www.cosic.esat.kuleuven.be/lane/`
5. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 260–276. Springer, Heidelberg (2009)
6. Khovratovich, D., Nikolić, Weinmann, R.: Meet-in-the-Middle Attacks on SHA-3 Candidates. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 228–245. Springer, Heidelberg (2009)
7. Mendel, F., Peyrin, T., Rechberger, C., Schläffer, M.: Improved Cryptanalysis of the Reduced Grøstl Compression Function, ECHO Permutation and AES Block Cipher. In: Jacobson, M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2009. LNCS, vol. 5867, pp. 16–35. Springer, Heidelberg (2009)
8. Lamberger, M., Mendel, F., Rechberge, C., Rijmen, V., Schläffer, M.: Rebound Distinguishers: Results on the Full Whirlpool Compression Function. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912. Springer, Heidelberg (to appear, 2009)
9. Matusiewicz, K., Naya-Plasencia, M., Nikolić, I., Sasaki, Y., Schläffer, M.: Rebound Attack on the Full LANE Compression Function. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912. Springer, Heidelberg (to appear, 2009)

Appendix

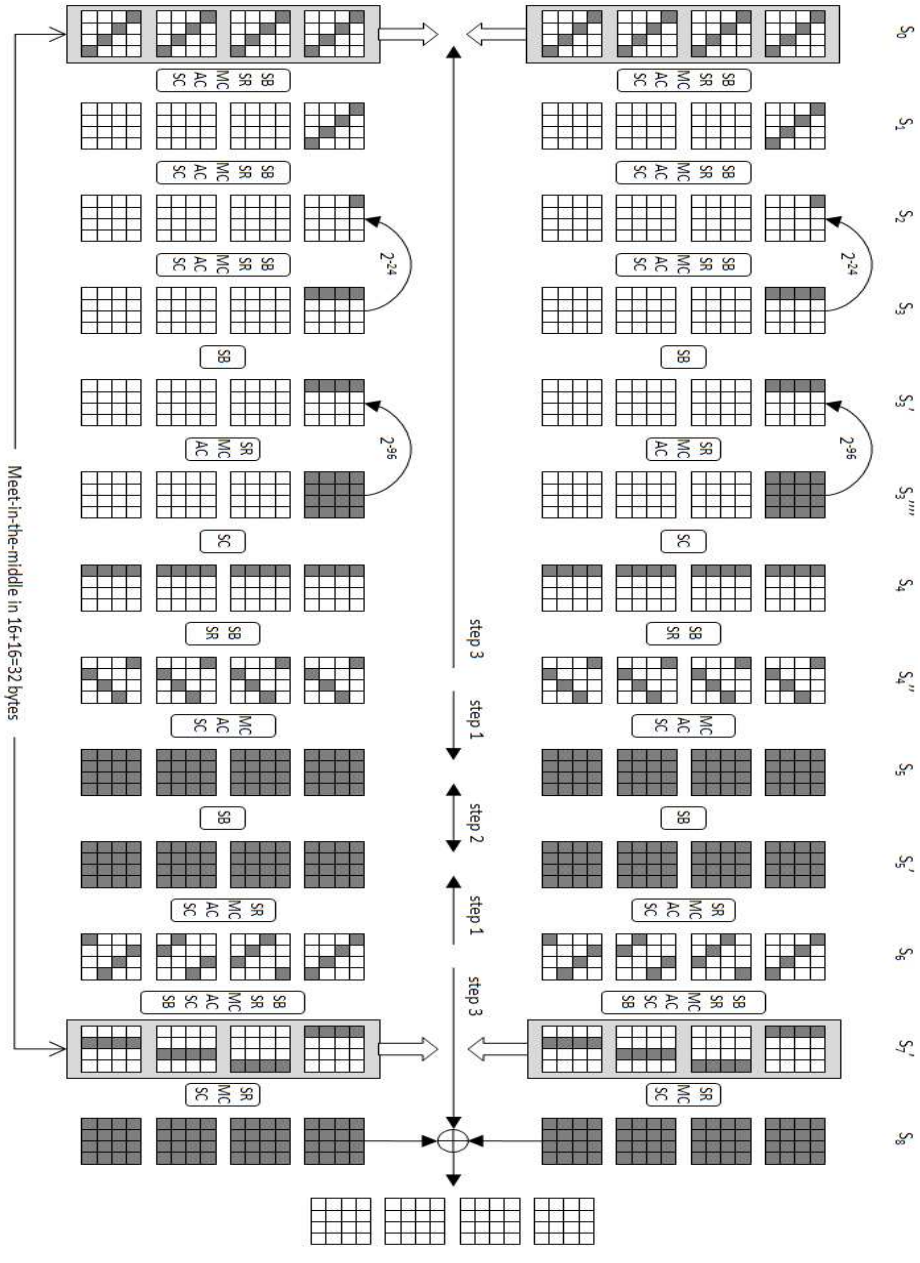

Fig. 8. Rebound differential path of inner collision for LANE-512

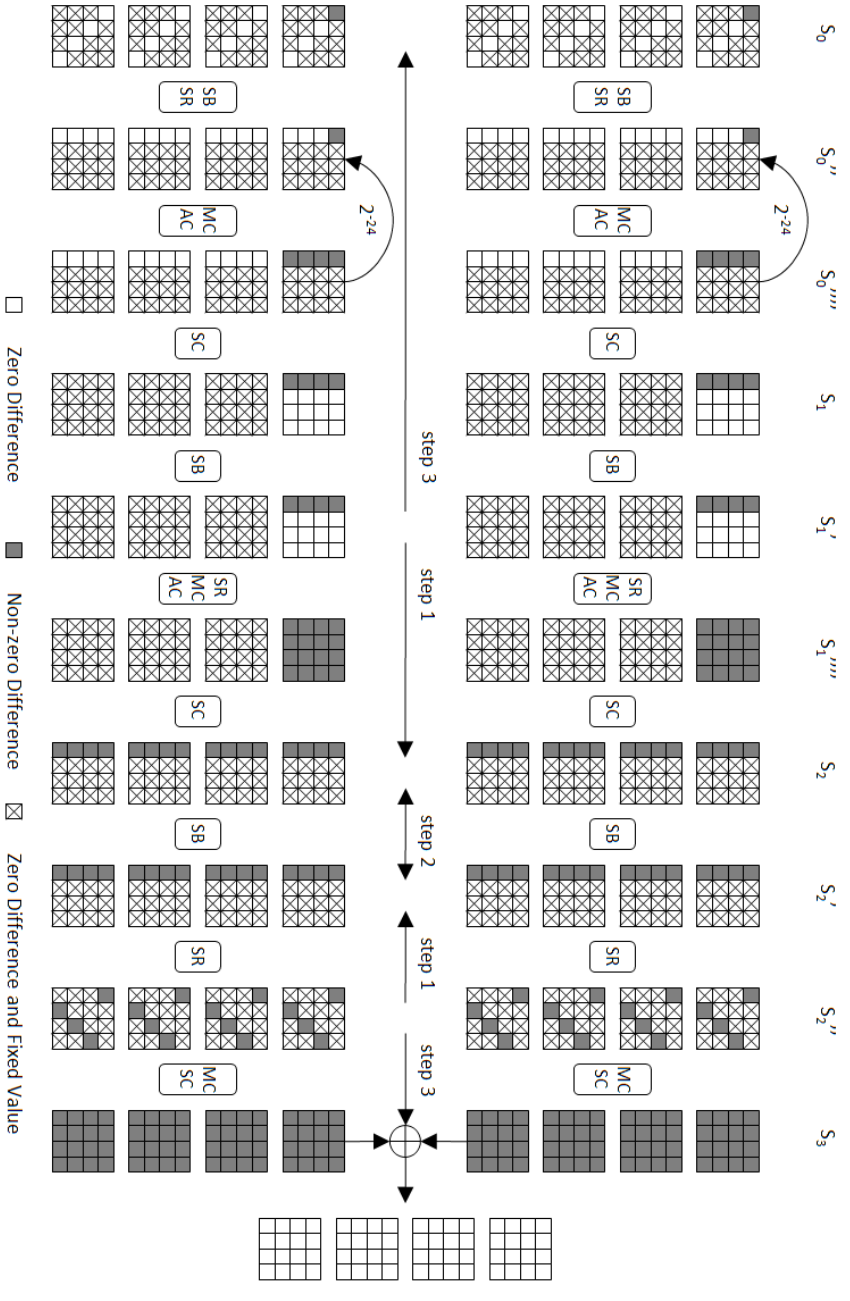

Fig. 9. Rebound differential path for LANE-512-(3,4)

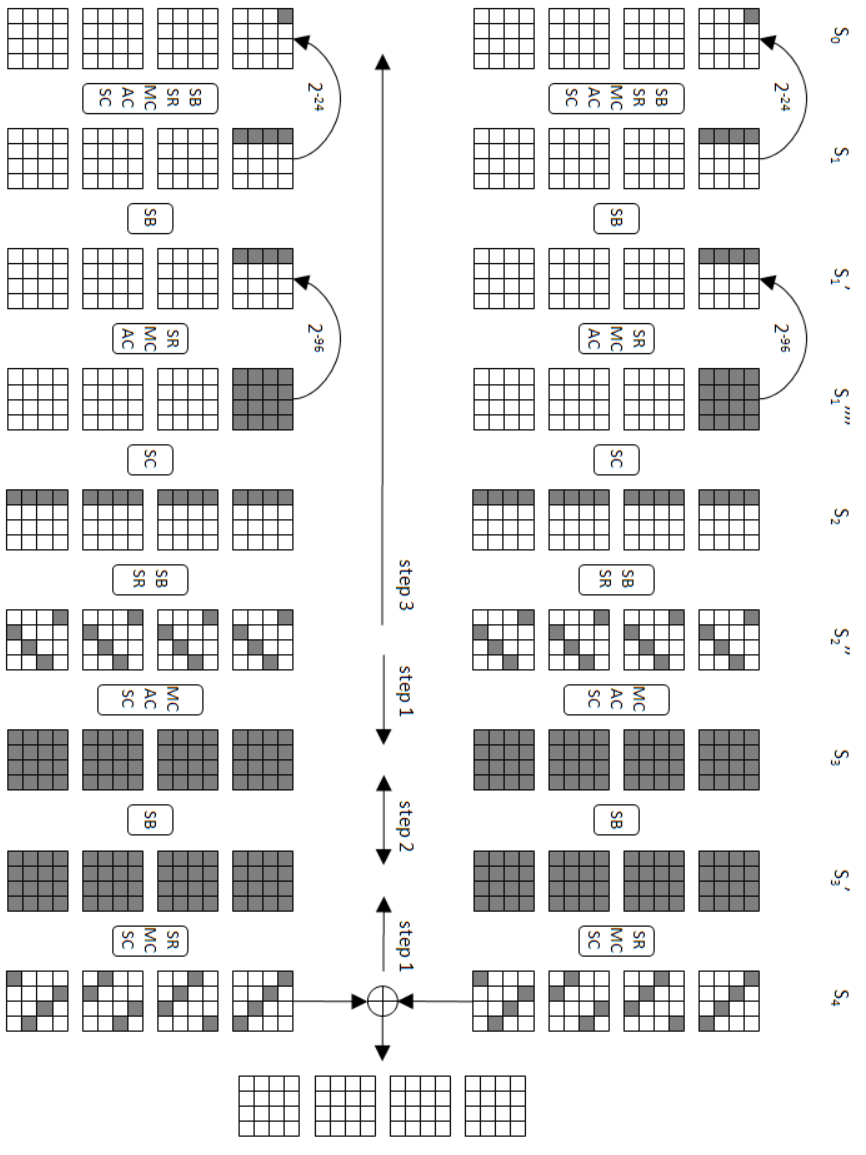

Fig. 10. Rebound differential path for LANE-512-(4,4)