Impossible Differential Cryptanalysis of Surge

Zhiqiang Qu*, Xuan Shen, Jun He

College of Information and Communication, National University of Defense Technology. Wuhan, Hubei, China

34399992@qq.com

Abstract. In 2018, Li Lang et al. proposed a new lightweight block cipher which is called Surge. Its block size is 64-bit, and the length of key size is 64/80/128-bit, respectively. They also proved that the cipher has good performance on security and effectively resists on differential attack, linear attack and algebraic attack. In this paper, some 4-round impossible differentials are constructed with the miss-in-the-middle method. Furthermore, 6-round impossible differential attacks are proposed on Surge-64/ Surge-80/ Surge-128 algorithms based on the 4-round impossible differentials. The data complexity of the attack is $2^{38}/2^{39}/2^{39.5}$ chosen plaintexts, the time complexity is $2^{51.4}/2^{51.7}/2^{60.9}$ 6-round encryptions, and the storage complexity is $2^{40}/2^{40.1}/2^{49}$ storage units. This is the first time to make the impossible differential cryptanalysis for Surge.

1. Introduction

In 2018, Li Lang et al. [1] proposed a new lightweight block cipher Surge which is low resource, high effectiveness and high security. Designers show that the algorithm can effectively resist against differential attack, linear attack and algebraic attack. As far as we know, impossible differential cryptanalysis is an effective analysis method for AES-like block ciphers [2-4]. This paper attempts to make an impossible differential analysis of Surge to further evaluate its security.

Unlike differential analysis, impossible differential analysis [5-6] uses differentials with probability zero to get rid of the wrong keys. If the guessed key makes the plaintext pairs satisfy the input and output of the impossible differentials, then the guessed key must be the wrong key. After many times of screening out the wrong keys, the remaining key is the correct key. Impossible differential analysis can be divided into two stages: the first stage is to construct the impossible differential as long as possible, and the second stage is to get rid of the wrong key using the constructed impossible differentials, so as to get the correct key. Construction of the impossible differentials is the main one for this attack.

In this paper, some 4-round impossible differentials are constructed for Surge, and then 6-round impossible differential attacks on Surge-64/ Surge-80/ Surge-128 algorithms are implemented for the first time with the constructed distinguishers. The data complexity of the attack is $2^{38}/2^{39}/2^{39.5}$, the time complexity is $2^{51.4}/2^{51.7}/2^{60.9}$, and the storage complexity is $2^{40}/2^{40.1}/2^{49}$ respectively, which is shown in TABLE I.
Table 1: the Result of 6-Round Impossible Differential Attack on Surge

| Algorithm | Data complexity | Time complexity | Storage complexity |
|-----------|-----------------|-----------------|-------------------|
| Surge-64  | $2^{38}$        | $2^{11.4}$      | $2^{46}$          |
| Surge-80  | $2^{39}$        | $2^{11.7}$      | $2^{40.1}$        |
| Surge-128 | $2^{38.5}$      | $2^{10.9}$      | $2^{49.2}$        |

The organization of this paper is as follows: In section II, we introduce the encryption and decryption process of Surge. Then, we construct some 4-round impossible differentials of Surge in section III. Furthermore, we propose 6-round impossible differential attacks with the constructed 4-round impossible differentials in section IV. Finally, we summarize this paper in section V.

2. Description of Surge
Surge is an AES-like block cipher with SPN structure. Its block size is 64 bit, and its master key length is 64/80/128 bit called Surge-64/Surge-80/Surge-128 respectively. Take Surge-64 for example, each 64-bit block is divided into 16 units as $X = (x_0, x_1, L, x_{15})$, and the round key is also divided into 16 units as $K = (k_0, k_1, L, k_{15})$. The Surge-64 algorithm has 32 rounds. The round function includes 5 operations: Addition Constant (AC), Addition Round Key (AK), S-box transformation (BS), Shift Row (SR) and Mixing Column (MC). It is shown in Fig. 1.

**Figure 1**: One round encryption processes of Surge.

**Addition Constant**: The round constants are expressed in hexadecimal digits with {0, 1, 2, 3} at high byte and {0, 1, 2, ..., f} at low byte. The constant addition transformation method is the $x_i$ XORing with the high byte of the $r$-th round constant, the $x_i$ XORing with the low byte of the $r$-th round constant. The round constant addition operation is denoted as $AC$, and the corresponding inverse operation in the decryption process is denoted as $AC^{-1}$.

**Addition Round Key**: It is a XOR operation between plaintext or the intermediate ciphertext and the round key of the $r$-th round. The key addition operation is denoted as $AK$, and the corresponding inverse operation in the decryption process is denoted as $AK^{-1}$. The round key of Surge-64 is the original key, as shown in Formula (1):

![Diagram](image_url)
For Surge-80, the original key can be expressed as $K = (k_0, k_1, L, k_{19})$ and round keys generated according to Formula (2):

\[
K' = \begin{bmatrix}
  k_0 & k_1 & k_2 & k_3 \\
  k_4 & k_5 & k_6 & k_7 \\
  k_8 & k_9 & k_{10} & k_{11} \\
  k_{12} & k_{13} & k_{14} & k_{15}
\end{bmatrix}, \quad 1 \leq r \leq 32
\]  

(1)

For Surge-128, the original key can be expressed as $K = (k_0, k_1, L, k_{31})$ and round keys generated according to Formula (3):

\[
K' = \begin{bmatrix}
  k_0 & k_1 & k_2 & k_3 \\
  k_4 & k_5 & k_6 & k_7 \\
  k_8 & k_9 & k_{10} & k_{11} \\
  k_{12} & k_{13} & k_{14} & k_{15} \\
  k_{16} & k_{17} & k_{18} & k_{19} \\
  k_{20} & k_{21} & k_{22} & k_{23} \\
  k_{24} & k_{25} & k_{26} & k_{27} \\
  k_{28} & k_{29} & k_{30} & k_{31}
\end{bmatrix}, \quad 1 \leq r \leq 32
\]  

(2)

\[
K' = \begin{bmatrix}
  k_0 & k_1 & k_2 & k_3 \\
  k_4 & k_5 & k_6 & k_7 \\
  k_8 & k_9 & k_{10} & k_{11} \\
  k_{12} & k_{13} & k_{14} & k_{15} \\
  k_{16} & k_{17} & k_{18} & k_{19} \\
  k_{20} & k_{21} & k_{22} & k_{23} \\
  k_{24} & k_{25} & k_{26} & k_{27} \\
  k_{28} & k_{29} & k_{30} & k_{31}
\end{bmatrix}, \quad 1 \leq r \leq 32
\]  

(3)

S-box transformation: It is the only non-linear component of Surge. The S-box transformation operation is denoted as $BS^r$, and the corresponding inverse operation in the decryption process is denoted as $BS^{-r}$.

Shift Row: For a $4 \times 4$ matrix consisting of 16 elements, each row of the matrix shift 3, 2, 1, 0 units to the left respectively shown in Fig. 1. The row shift transformation operation is denoted as $SR^r$, and the corresponding inverse operation in the decryption process is denoted as $SR^{-r}$.

Mixing Column: According to the finite field operation $GF(2^8)$, transform matrix $M$ is constructed shown in Formula (4). Here $M$ is a MDS matrix. The data in the matrix is expressed in hexadecimal. The mixing column operation is denoted as $MC^r$, and the corresponding inverse operation in the decryption process is denoted as $MC^{-r}$.
Generally speaking, plaintext $P$ is encrypted into $C$ after $n$ rounds encryption, and the encryption process is shown in Fig. 2. Note that there is no $MC$ operation in the last round of the algorithm.

3. Round Impossible Differentials of Surge

In this section, we will construct a 4-round impossible differential using the miss-in-the-middle method. We constructed a 4-round impossible differential of Surge shown in Fig. 3. It should be noted that round constant addition and key addition are XOR operations, which do not affect differential propagation. Therefore, this paper considers round constant addition and key addition as an operation. When the input difference of the first round is non-zero at $x_i$ and the other positions are all zero, after 2 rounds from the encryption direction, the output difference of the second round in the middle is non-zero at each position. Assuming that the output difference of the fourth round at $(x_1, x_2, x_3, x_4)$ is zero and the rest is arbitrary, the input difference of the third round at $(x_1, x_2, x_3, x_4)$ is zero after 2 rounds from the decryption. This is contradictory to the output difference of the second round, so a 4-round impossible differential can be obtained.

For convenience of expression, this paper defines $\varepsilon\{x_1, x_2, L, x_4\}$ denotes the values of $x_1, x_2, L, x_4$ are non-zero and others are zero, and $\eta\{x_1, x_2, L, x_4\}$ denotes the values of $x_1, x_2, L, x_4$ are zero and others are arbitrary. Then $\varepsilon\{x_i\}$ is a 4-round impossible differential shown in Fig 3.

![Figure 3](image-url)
4. Round Impossible Differential Attack on Surge

In this section, we presented 6-round impossible differential attack on Surge shown in Fig. 4 based on the 4-round impossible differential described in previous section. The 6-round impossible differential attack on Surge is constructed with additional 1 round at the beginning and the end of the 4-round impossible differential respectively. If plaintext pair whose difference satisfies the input and ciphertext pair whose difference satisfies the output, they constitutes an impossible differential. According to the impossibility of difference, all round key guessed satisfying the above conditions are incorrect.

4.1. Round Impossible Differential Attack on Surge-64

From Fig. 3, it can be seen that there is no $MC$ in the last round of 4-round impossible differential, and in 6-round of attack, 4-round impossible differential is in the middle, so $MC$ should be added after 4-round impossible differential. That is, the last round decryption operation is $MC^{-1} \circ AC^{-1} \circ AK^{-1} \circ BS^{-1} \circ SR^{-1}(C)$. Then a 6-round attack can be constructed as shown in Fig. 4. The procedure of this attack is as follows:

**Step 1:** Choose structures of $2^n$ plaintexts which differ only at $(x_1, x_2, x_3, x_4, x_5, x_9, x_{10}, x_{14}, x_{15})$, having all possible values in these positions. One structure proposes $2^6 \times 2^6 = 2^{12}$ pairs of plaintexts, so $2^{11+n}$ plaintexts pairs can be obtained.

**Step 2:** Choose pairs whose ciphertext pairs have zero difference at $(x_0, x_1, x_4, x_5, x_9, x_{10}, x_{14}, x_{15})$. The expected number of such pairs is about $2^{11+n} \times 2^{-3} = 2^{n-1}$.

**Step 3:** For remaining ciphertext pairs, guess the value of $(k^{12}, k^{14}, k^{15}), k^{11}$, and each middle ciphertext is computed by $MC^{-1} \circ AC^{-1} \circ AK^{-1} \circ BS^{-1} \circ SR^{-1}(C)$ separately, and their difference must be zero in the last two columns. If the difference at $(x_1, x_{12}) / (x_1, x_9) / (x_4, x_9) / (x_5, x_{12})$ is zero (The probability is $2^{-8} \times 4 = 2^{-6}$). Correspondingly, the difference at $(x_1, x_6, x_{11}, x_{12}) / (x_2, x_3, x_9, x_{13}) / (x_3, x_4, x_9, x_{14}) / (x_4, x_5, x_{10}, x_{15})$ must be zero, the expected number of the remaining pairs is about $2^{8-1} \times 2^{n} = 2^{7}$.

**Step 4:** Guess the value of $(k^1, k^1, k^1, k^1)$, the plaintext pair in the previous step is encrypted 1 round. If the output difference of ciphertext pairs satisfies $\epsilon(x_0) / \epsilon(x_4) / \epsilon(x_5) / \epsilon(x_{12})$ (The probability is $2^{-13} \times 4 = 2^{-10}$). Since such a difference is impossible, every round key that satisfies the difference in the third or fourth step is incorrect.

**Step 5:** Repeat step 3 and step 4 until find the right value of $(K^1, K^n)$.

![Figure 4](image-url) 6-round impossible differential attack on Surge.
Complexity analysis: According to formula (1), the round key of Surge-64 is the seed key, that is $k_{6}^{6} = k_{12}^{6}$.

After step 5, the number of wrong $(K^i,K^o)$ is approximately $71.04 \times 2^{32} \approx 2^{38}$ chosen plaintexts. The time complexity of the attack is dominated by step 3 and step 4. In this attack there are about $2 \times 2^{32} \times 2^{n-1} = 2^{33} \approx 2^{34}$ partial decryptions required in step 3, Equivalent to $2^{33} \times 8/16 = 2^{34}$ rounds of decryption. Therefore, the storage complexity of the attack is $2^{33} \times 8/16 = 2^{34}$ partial decryption required in step 4, Equivalent to $2^{33} \times 4/16 = 2^{34}$ rounds of decryption. In the process of attack, we need to store $2^{n-1}$ ciphertext pairs in step 3, $2^{n-7}$ ciphertext pairs in step 4, and $2^{40}$ guessed keys.

After 6-rounds impossible differential attack, we can recover 40 bits of the seed key. The remaining 24 bits can be obtained by exhaustive method. So, the total time complexity is $2^{51.4} \cdot 2^{24} \approx 2^{51.4}$.

### 4.2. Round Impossible Differential Attack on Surge-80

The same method was used to attack Surge-80 algorithm, we need guess 32-bit value of $k^{6}$ at $(k_{5}^{6}, k_{9}^{6}, k_{10}^{6}, k_{11}^{6}, k_{12}^{6}, k_{15}^{6}, k_{16}^{6}, k_{17}^{6})$ in step 3, and 16-bit value of $K^i$ at $(k_{3}^{1}, k_{6}^{1}, k_{9}^{1}, k_{12}^{1})$ in step 4. According to formula (2), we know that $k_{6}^{6} = k_{12}^{6}$. So, we need guess 8-bit value of $K^i$ in step 4.

Complexity analysis: The data complexity of the attack is $2^{n+16} = 2^{34+22} = 2^{38}$ chosen plaintexts. The time complexity of the attack is about $(2^{34} \cdot 2^{22}) \times 1/6 \approx 2^{34} \cdot 2^{18} = 2^{52}$-round encryptions of Surge. In the process of attack, we need to store $2^{n-1}$ ciphertext pairs in step 3, $2^{n-7}$ ciphertext pairs in step 4, and $2^{40}$ guessed keys.

After 6-rounds impossible differential attack, we can recover 40 bits of the seed key. The remaining 24 bits can be obtained by exhaustive method. So, the total time complexity is $2^{51.4} \cdot 2^{24} \approx 2^{51.4}$.

### Table 2 Key Recovery and Attack Complexity of Surge-80

| Impossible differential | Key recovery | Data complexity | Time complexity | Storage complexity |
|-------------------------|-------------|-----------------|----------------|-------------------|
| $\varepsilon[x_1] \rightarrow \eta[x_1,x_2]$ | 40bit | $2^{38}$ | $2^{51.4}$ | $2^{40}$ |
| $\varepsilon[x_1] \rightarrow \eta[x_1,x_2]$ | 36bit | $2^{38}$ | $2^{51.4}$ | $2^{16}$ |
| In total | 76bit | $2^{39}$ | $2^{51.7}$ | $2^{51.1}$ |

We only recover 40 bits of the key, consider recovering the remaining key information, a new 6-rounds impossible differential attack is carry out. The key recovery and attack complexity of the two attacks are shown in TABLE II.

Thus, we can recover 76 bits of the seed key. The remaining 4 bits can be obtained by exhaustive method. So, the total time complexity is $2^{51.7} \cdot 2^{4} \approx 2^{51.7}$.

### 4.3. Round Impossible Differential Attack on Surge-128

The same method in section B was used to attack Surge-128 algorithm, the key recovery and attack complexity of the two attacks are shown in TABLE III.
Table. 3 Key Recovery and Attack Complexity of Surge-128

| Impossible differential | Key recovery | Data complexity | Time complexity | Storage complexity |
|-------------------------|-------------|----------------|----------------|-------------------|
| $\epsilon[x_1] \rightarrow$ | 48bit       | $2^{38.5}$     | $2^{59.9}$     | $2^{48}$          |
| $\eta[x_1, x_2, x_3, x_4]$ |             |                |                |                   |
| $\epsilon[x_1] \rightarrow$ | 48bit       | $2^{38.5}$     | $2^{59.9}$     | $2^{48}$          |
| $\eta[x_1, x_2, x_3, x_4]$ |             |                |                |                   |
| In total               | 96bit       | $2^{39.5}$     | $2^{61.9}$     | $2^{48}$          |

Thus, we can recover 96 bits of the seed key. The remaining 32 bits can be obtained by exhaustive method. So, the total time complexity is $2^{60.9} + 2^{32} \approx 2^{60.9}$.

5. Conclusion
In this paper, some 4-round impossible differentials of Surge are constructed. With the 4-round impossible differentials, we give 6-round impossible differential attacks on Surge-64/ Surge-80/ Surge-128 algorithm respectively. As far as we know, this paper is the first time to make the impossible differential cryptanalysis for Surge. The analysis results show that Surge algorithm is immune to the impossible differential attack. At the same time, the analysis method in this paper can be further applied to the analysis of other AES-like block ciphers.

References
[1] L. LI, B. LIU, “Surge : A new low-resource and efficient lightweight block cipher,” Computer science, vol.45, No.2, pp. 236-240, 2018.
[2] B. Sun, M. Liu, J. Guo, V. Rijmen, and R. Li, “Provable security evaluation of structures against impossible differential and zero correlation linear cryptanalysis,” EUROCRYPT 2016, M. Fischlin and JS. Coron, eds., LNCS 9665, pp.196–213, Springer, Heidelberg, 2016.
[3] W. Zhang, W. Wu, and D. Feng, “New results on impossible differential cryptanalysis of reduced AES,” ICISC 2007, LNCS 4817, pp.239–250, Springer, Heidelberg, 2007.
[4] X. SHEN, G. LIU, C. LI, L. QU, “Impossible Differential Cryptanalysis of Fantomas and Robin,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E101.A.5, pp.863-866, 2018.
[5] L.R. Knudsen, DEAL—A 128-bit block cipher, Department of Informatics, University of Bergen, Norway, Technical Report, 1998.
[6] E. Biham, A. Biryukov, and A. Shamir, “Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials,” Advances in Cryptology- EUROCRYPT 1999, J. Stern, eds., LNCS 1592, pp.12–23, Springer, Heidelberg, 1999.