Automatic Search of Impossible Differentials and Zero-Correlation Linear Hulls*

Shengbao Wu and Mingsheng Wang

Chinese Academy of Sciences

ASK 2013, Aug. 29th@Weihai,China

*Partial work of this talk has been published in Indocrypt 2012.

Outline

- Introduction
- Automatic Search of Impossible Differentials
- Automatic Search of Zero-Correlation Linear Hulls
- Discussions and Conclusions

Outline

- Introduction
  - Impossible differential cryptanalysis (IDC) and zero-correlation linear cryptanalysis (ZCLC)
  - Previous tools for finding impossible differentials (IDs) and their disadvantages
  - Our contributions
- Automatic Search of Impossible Differentials
- Automatic Search of Zero-Correlation Linear Hulls
- Conclusions

Introduction

——Impossible differential cryptanalysis and zero-correlation linear cryptanalysis

- Impossible Differential Cryptanalysis (IDC)
  - One of the most popular cryptanalytic tools for block ciphers
  - Proposed by Knudsen (1998) to attack DEAL and extended by Biham et al. to analyze IDEA and Skipjack
  - Exploits differentials with probability zero to recover keys
- Zero-Correlation Linear Cryptanalysis (ZCLC)
  - The counterpart of IDC in the domain of linear cryptanalysis
  - Exploits zero-correlation linear hulls (ZCLH) to recover keys
  - Has attracted much attention in the last two years (DCC 2012, FSE'12, Asiacrypt'12, SAC'13), and has been proved to be more powerful than IDC in some cases

Introduction

——Impossible differential cryptanalysis and zero-correlation linear cryptanalysis

- Perform a successful IDC and ZCLC
  - Two important factors
    - Length of an ID/a ZCLH, number of IDs/ZCLHs
  - Length of an ID/a ZCLH
    - The longer the ID/ZCLH is, the better the attack will be.
  - Find more IDs/ZCLHs
    - More IDs/ZCLHs we find, more probabilities we can perform a successful attack or improve known attacks.

Introduction

——Previous tools and their disadvantages

- $\mathcal{U}$-method and UID-method
  - In Indocrypt 2003, Kim et al. proposed the $\mathcal{U}$-method to find impossible differentials for block cipher structures with bijective round functions.
  - Extended by Luo et al. (2009), and named as the UID-method
  - Based on the miss-in-the-middle approach

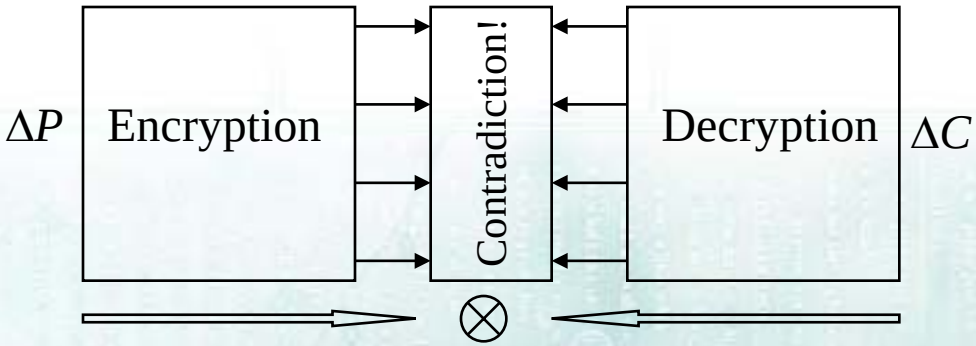


Fig 1. Basic idea of the miss-in-the-middle approach

  - Employed by the designers of LBlock (ACNS'11), Piccolo (CHES'11) and TWINE (SAC'12)

Introduction

——Previous tools and their disadvantages

- Disadvantages of previous tools
  - Miss-in-the-middle approach limits their power

Fig 2. Miss-in-the-middle approach (1-(a)) and IDs with information feedback (1-(b))

  - Failed to find the longest known IDs of many block ciphers, such as Camellia, MIBS and E2
  - There is a gap between previous tools and ad hoc approaches

Introduction

——Our contributions

- Proposed a new tool to search truncated IDs for word-oriented block ciphers with bijective Sboxes (Indocrypt'12)
  - The $\mathcal{U}$-method and the UID-method are specific cases of our tool, and our tool is more powerful than them.
  - Although our tool does not improve the lengths of IDs for existing word-oriented block ciphers, it helps in reducing the gap between previous tools and ad hoc approaches
  - Not only rediscovers the longest truncated IDs of many word-oriented block ciphers known so far, but also finds new results

Introduction

——Our contributions

Table 1. Summary of new truncated IDs for word-oriented block ciphers.

Block cipher	Word unit	Previous results			In this paper		
		Round	No. of IDs	Method	Round	No. of IDs	**New IDs**
CLEFIA	Byte	9	72	ad hoc	9	72	**0**
AES	Byte	4	269,554	ad hoc	4	3,608,100	**3,338,546**
ARIA	Byte	4	156	ad hoc	4	94,416	**94,260**
Camellia*	Byte	8	3	ad hoc	8	4	**1**
E2	Byte	6	1	ad hoc	6	56	**55**
MIBS	Nibble	8	2	ad hoc	8	8	**6**
LBlock	Nibble	14	64	U-method	14	80	**16**
Piccolo	Nibble	7	1	U-method	7	450	**449**

Introduction

——Our contributions

- Extend our tool to search IDs for bit-based block ciphers (new)
  - Use more properties of Sboxes, besides that they are bijective
- Extend our tool to search ZCLHs for word-oriented block ciphers and bit-based block ciphers (new)

Introduction

——Our contributions

Table 2. Compare the results of IDs and ZCLHs (one 2.66GHz core, Magma)

Block cipher	**Results of IDs**		**Time**	**Results of ZCLHs**		**Time**
	round	No		round	No	
ARIA	4	94260	2 weeks	4	58128	days
E2	6	56	Hours (Enumerate half side)	6	41	Hours (Enumerate half side)
MIBS	8	6		8	2	
Camellia	8	4		8	8	
LBlock	14	80		14	80	
PRESENT-48	6	146	days (from 1 Sbox to 1 Sbox)	5	4200	days (from 1 Sbox to 1 Sbox)
PRESENT-64	6	3776		6	960	
PRESENT-96	6	40621		6	12275	

Note: PRESENT-like ciphers with block size 48/64/96 are in fact block ciphers EPCBC-48, PRESENT, EPCBC-96 (CANS'11), respectively.

Outline

- Introduction
- Automatic Search of Impossible Differentials
  - Sketch of our tool
  - Difference propagation system
  - Some details of our tool
    - Word-oriented block cipher
    - Bit-based block cipher
  - Complexity analysis
- Automatic Search of Zero-Correlation Linear Hulls
- Discussions and Conclusions

Automatic Search of Impossible Differentials

——Sketch of our tool

- Treat a block cipher as an entirety, describe the propagation of differences in a block cipher as a system of equations —— difference propagation system
- Predict information by solving a difference propagation system iteratively
  - Abort condition: a contradiction is detected or we cannot get new information any longer
- A contradiction is detected => the difference propagation system has no solution => an impossible differential is found

Automatic Search of Impossible Differentials

——Algorithm for searching IDs

1. Build the difference propagation system of a block cipher;
2. **for** *each pair of* $(\Delta P, \Delta C)$ *we choose* **do**;
3. *index*:=true;
4. **while** *index* **do**;
5. Predict information by solving the difference propagation system;
6. **if** *a contradiction is found* **then**;
7. *index*:=false; **return** true;
8. **elseif** *cannot get any new information* **then**;
9. *index*:=false;
10. **end if**;
11. **end while**;
12. **end for**;

Algorithm 1. Algorithm for searching IDs

Automatic Search of Impossible Differentials

——Algorithm for searching IDs

1. Build the difference propagation system of a block cipher;
2. **for** *each pair of* $(\Delta P, \Delta C)$ *we choose* **do**;
3. *index*:=true;
4. **while** *index* **do**;
5. Predict information by solving the difference propagation system;
6. **if** *a contradiction is found* **then**;
7. *index*:=false; **return** true;
8. **elseif** *cannot get any new information* **then**;
9. *index*:=false;
10. **end if**;
11. **end while**;
12. **end for**;

Algorithm 1. Algorithm for searching IDs

Automatic Search of Impossible Differentials

——Difference propagation system

- Build equations for basic primitives
  - consider the XOR difference and suppose subkeys are XORed to the state, then the key addition layers can be omitted.
  - Four basic primitives are widely used in a block cipher

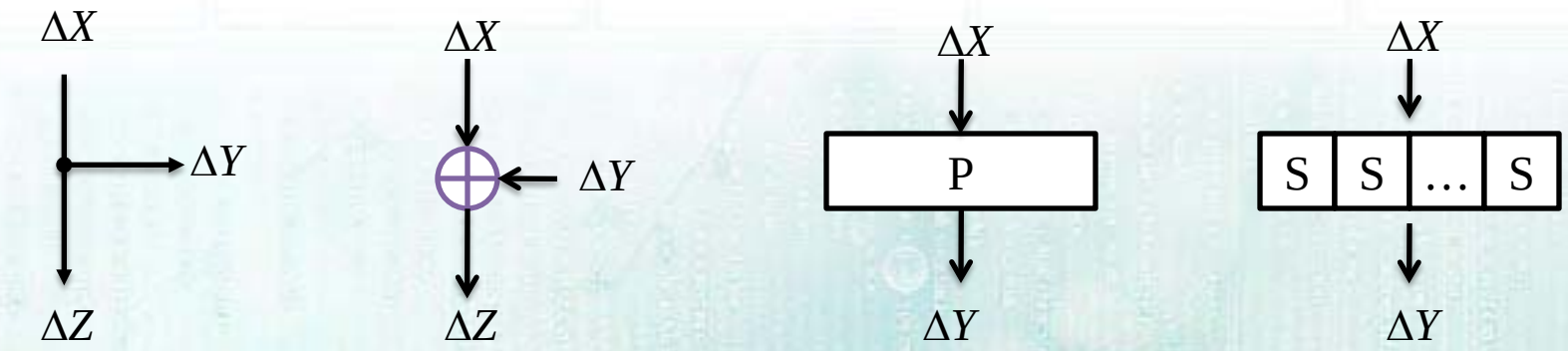
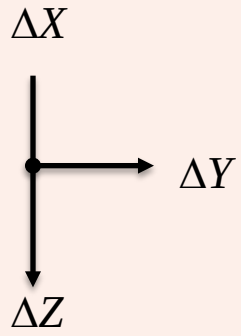
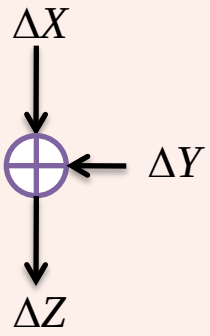
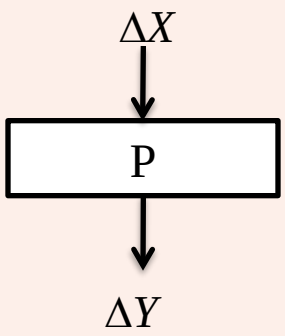
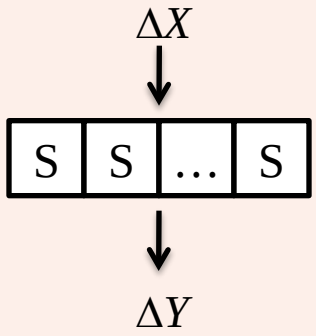


Fig 3. Basic primitives of a block cipher

Automatic Search of Impossible Differentials

——Difference propagation system

- Suppose $\Delta X = (\Delta x_1, \Delta x_2, \ldots, \Delta x_n), \Delta Y = (\Delta y_1, \Delta y_2, \ldots, \Delta y_n)$ and $\Delta Z = (\Delta z_1, \Delta z_2, \ldots, \Delta z_n)$

Basic primitives				
Equations	$\Delta x_i \oplus \Delta y_i = 0,$ $\Delta x_i \oplus \Delta z_i = 0$	$\Delta x_i \oplus \Delta y_i \oplus \Delta z_i = 0$	$\Delta y_i \oplus \sum_{j=1}^{n} P_{i,j} \cdot \Delta x_j = 0$	$\overline{S}(\Delta x_i, \Delta y_i) = 0$
Note	include *2n* linear equations	include *n* linear equations	include *n* linear equations	include *n* non-linear equations from *n* Sboxes

Automatic Search of Impossible Differentials

——Non-linear equation derived from an Sbox

- Build the difference distribution table (DDT) of an Sbox $F_2^s \to F_2^t$ ($2^s \times 2^t$ entries)
  - DDT[i][j]>0 means that the input difference i may propagate to the output difference j, with some probabilities
- Replace each nonzero entry of a DDT with 1
  - In algebraic view, the replaced DDT is equivalent to a truth table with 2^{s+t} entries
- Recover the boolean function f based on the truth table
  - $s+t$ variables represent the input and output difference
  - $f+1=0$ totally catches the difference propagation behavior of this Sbox

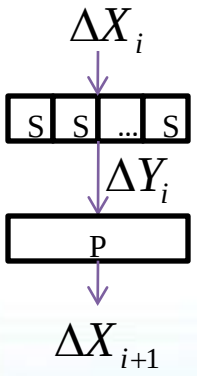
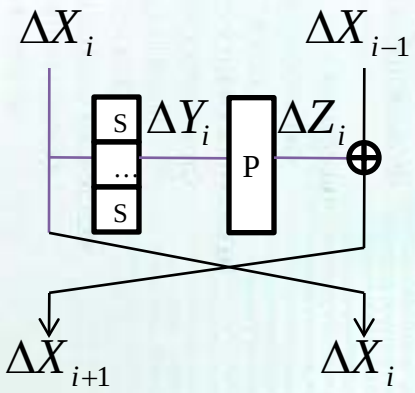
Automatic Search of Impossible Differentials

——Difference propagation system

- Difference propagation system of a block cipher
  - Combine the equations derived from basic primitives together to build a difference propagation system
  - Choose *r*-round SPN ciphers and Feistel ciphers with SPN round functions as examples

Automatic Search of Impossible Differentials

——Difference propagation system

Block ciphers	Difference propagation system	Note
	$\begin{cases} \overline{S}(\Delta X_{i,j}, \Delta Y_{i,j}) = 0, \\ \Delta X_{i+1}^T \oplus P \cdot \Delta Y_i^T = 0 \end{cases}$ $(1 \le i \le r, 1 \le j \le n)$	$N=(2r+1)n$ variables rn linear equations rn non-linear equations from Sbox layers
	$\begin{cases} \overline{S}(\Delta X_{i,j}, \Delta Y_{i,j}) = 0, \\ \Delta Z_i^T \oplus P \cdot \Delta Y_i^T = 0, \\ \Delta X_{i-1} \oplus \Delta Z_i \oplus \Delta X_{i+1} = 0 \end{cases}$ $(1 \le i \le r, 1 \le j \le n)$	$N=(3r+2)n$ variables $2rn$ linear equations rn non-linear equations from Sbox layers

Automatic Search of Impossible Differentials

——Difference propagation system and impossible differentials

Theorem. $\Delta P \to \Delta C$ is an impossible differential, if the difference propagation system initialized by $\Delta P, \Delta C$ has no solution.

Proof. A block cipher is built by a list of basic primitives and the difference propagation behavior of each primitive is totally described by the corresponding equations. A differential trail in fact defines a solution of the difference propagation system. Thus, the difference propagation system initialized by $\Delta P, \Delta C$ has no solution implies that there is no differential trails starting from ΔP and ending at ΔC, that is, $\Delta P \to \Delta C$ is an impossible differential.

Automatic Search of Impossible Differentials

——Solve a difference propagation system

- **An impossible differential is obtained if we confirm that the corresponding difference propagation system has no solution!**
- Difficulty and our strategy
  - Solve a nonlinear system is difficult!
  - Our strategy: predict information by partially solving the system. An impossible differential is confirmed if we find some contradictions during information prediction.

Automatic Search of Impossible Differentials

——Algorithm for searching IDs

1. Build the difference propagation system of a block cipher;
2. **for** *each pair of* $(\Delta P, \Delta C)$ *we choose* **do**;
3. *index*:=true;
4. **while** *index* **do**;
5. Predict information by soling the difference propagation system;
6. **if** *a contradiction is found* **then**;
7. *index*:=false; **return** true;
8. **elseif** *cannot get any new information* **then**;
9. *index*:=false;
10. **end if**;
11. **end while**;
12. **end for**;

Algorithm 1. Algorithm for searching IDs

Automatic Search of Impossible Differentials

——Predict information and detect contradictions

- A difference propagation system can be divided into two subsystems — $\mathcal{L}$ and $\mathcal{NL}$
  - $\mathcal{L}$ includes all linear equations
  - $\mathcal{NL}$ includes all non-linear equations from the Sbox layers

Example 1. For an r-round SPN cipher, we have

$$\begin{cases} \overline{S}(\Delta X_{i,j}, \Delta Y_{i,j}) = 0 \ (1 \le i \le r, 1 \le j \le n), \longrightarrow \text{System } \mathcal{NL} \\ \Delta X_{i+1}^{T} \oplus P \cdot \Delta Y_{i}^{T} = 0 \ (1 \le i \le r, 1 \le j \le n) \longrightarrow \text{System } \mathcal{L} \end{cases}$$

Automatic Search of Impossible Differentials

——Predict information and detect contradictions

- Predict information from $\mathcal{NL}$

Lemma 1. Suppose S is a bijective Sbox, x is its input difference and y is its output difference, then

$$x=0\ (\neq 0) \text{ if and only if } y=0\ (\neq 0).$$

- Predict information from $\mathcal{L}$
  - If linear system $\mathcal{L}$ has solutions, we can solve it by Gauss-Elimination algorithm and recover information from the remaining system

Lemma 2. After solving a linear system by Gauss-Elimination algorithm, then
1) If an affine equation with only a variable, i.e., $x \oplus c=0$ (c is a constant), is found in the system, then $x=0$ if $c=0$ and $x\neq 0$ if $c\neq 0$;
2) If a linear equation with two variables, i.e., $x \oplus y=0$, is found in the system, then $x\neq 0$ if and only if $y\neq 0$.

Automatic Search of Impossible Differentials

——Predict information and detect contradictions

Example 2. Suppose we have known that $u=0, x\neq 0$, and the difference propagation system is

$$\begin{cases} \overline{S}(u,v)=0, \\ x\oplus y=0, \\ x\oplus y\oplus z=0. \end{cases}$$

After one step of information prediction, we have

$$\begin{cases} \overline{S}(u,v)=0, \\ x\oplus y=0, \\ x\oplus y\oplus z=0. \end{cases} \xrightarrow{\text{Lemma 1}} v=0$$

$$\begin{cases} \overline{S}(u,v)=0, \\ x\oplus y=0, \\ x\oplus y\oplus z=0. \end{cases} \xrightarrow{\text{Gauss-Elimination}} \begin{cases} x\oplus y=0, \\ z=0. \end{cases} \xrightarrow{\text{Lemma 2}} \begin{array}{l} y\neq 0, \\ z=0 \end{array}$$

Automatic Search of Impossible Differentials

——Predict information and detect contradictions

- Detect contradictions

Proposition 1. For given plaintext difference ΔP and ciphertext difference ΔC, $\Delta P \to \Delta C$ is impossible if one of the following two situations happens:

1) The linear system $\mathcal{L}$ doesn't have any solution. That is, the rank of its coefficient matrix is not equal to the rank of its augmented matrix;

2) There exists a variable with both zero and nonzero values.

Automatic Search of Impossible Differentials

——Predict information and detect contradictions

- A tiny example of the second type of contradiction

Example 3. Suppose we have known that $x=0, z\neq 0$, and the following equations are included in a difference propagation system

$$\begin{cases} \overline{S}(x,y)=0, \\ y\oplus z=0. \end{cases}$$

After a step of information prediction, we have

$$\begin{cases} \overline{S}(x,y)=0, \xrightarrow{\text{Lemma 1}} y=0 \\ y\oplus z=0. \xrightarrow{\text{Lemma 2}} y\neq 0 \end{cases}$$

Contradiction!

Automatic Search of Impossible Differentials

——Drawbacks of Lemma 1

- Lemma 1 is not able to exploit any properties of the Sboxes beyond the fact that they are bijective.
  - May miss some longer IDs if other properties of an Sbox is required in detecting them.
- Suitable for searching truncated IDs of word-oriented block ciphers, where the input/output difference of an Sbox propagates through the block cipher as a unit.
  - Results listed in Tab.1 are obtained by this technique.

Automatic Search of Impossible Differentials

——Situations in bit-based block ciphers

- Situation is changed in bit-based block ciphers
  - Difference propagation system is not word-oriented but bit-oriented, each variable represents a bit.
  - The input/output difference of an Sbox is broken into bits and no longer influences other variables as a unit.
    - May encounter the situation that partial bits of an Sbox are known while other bits are unknown.
    - Known bits of an Sbox may help us to retrieve new information of other bits

Automatic Search of Impossible Differentials

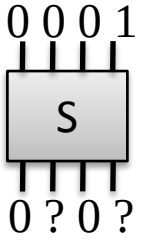
——Predict information from an Sbox

- Retrieve more information from an Sbox
  - A bit of an Sbox can be 0, 1 or ? (3^{t+s} cases)
  - For each case
    - Substitute the variables in the non-linear equation with known bits, and then solve it to get new information of the unknown bits.
  - Precompute and store new information in a table
    - In the procedure of information prediction, add new information to the difference propagation system
  - Useful new information
    - The value of a bit
    - A linear relation between unknown bits
    - An inconsistent case

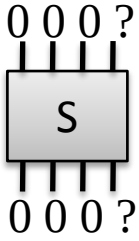
Automatic Search of Impossible Differentials

——Situations in bit-based block ciphers

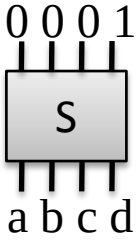
Example 4. Consider the following three cases of PRESENT's Sbox



inconsistent case



both ? are zero



d=1,b=0,a+c+1=0

The precomputed table is as follows

Input and output difference	New information
[0 0 0 1; 0 ? 0 ?]	inconsistent case
[0 0 0 a; 0 0 0 b]	a=b=0
[0 0 0 1; a b c d]	d=1,b=0,a+c+1=0
……	……

Automatic Search of Impossible Differentials

——Predict information from an Sboxes

Example 5. Suppose we consider a small version of PRESENT with 16 bits. The plaintext difference is

(0000, 0000, 0000, ****)

while the ciphertext difference is

(0000, 0000, 0000, ****).

Then, we derive that six bits of each Sbox in the second round are zero while other bits are known.

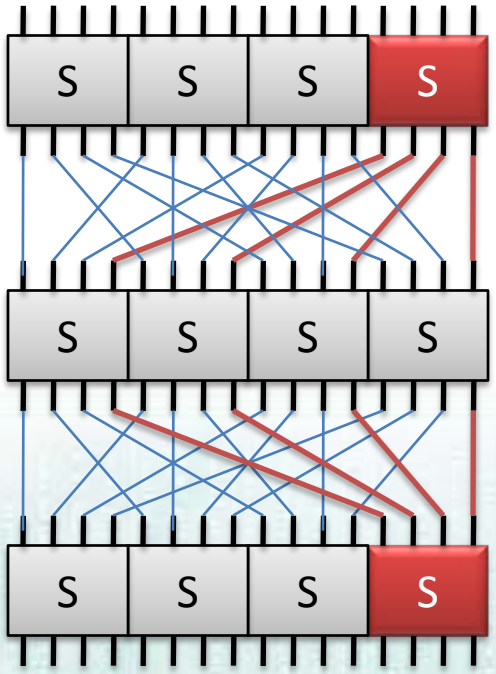
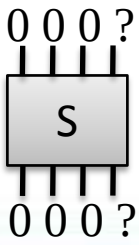


Fig.4. A small version of PRESENT

Automatic Search of Impossible Differentials

——Situations in bit-based block ciphers

Example 5. The second case of example 4, that is



both ? are zero

implies that differential listed in right side is impossible! This impossible differential can not be detected if we only use the bijective property of PRESENT's Sbox.

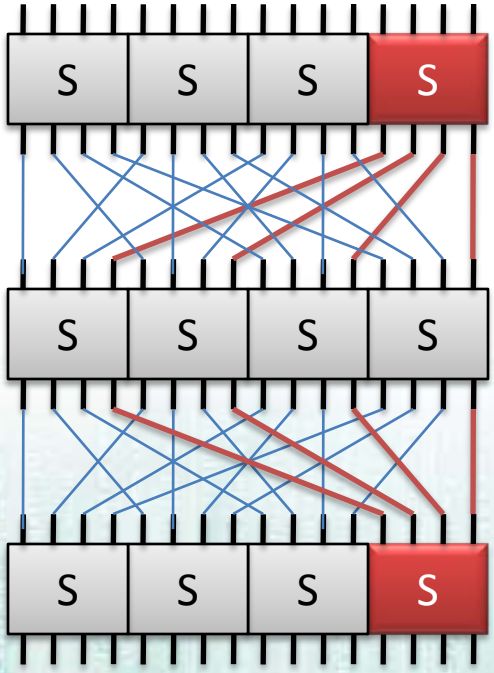


Fig. 5. A 3-round impossible differential

Automatic Search of Impossible Differentials

——Detect contradictions for bit-based block ciphers and experimental results

- Detect contradictions for a bit-based block cipher

Proposition 1'. Besides the condition 1) and 2) given in Proposition 1, $\Delta P \rightarrow \Delta C$ is also impossible if

3) An inconsistent case is found during predicting information from an Sbox.

- Experimental results for PRESENT-like ciphers
  - After exploiting more properties of the Sboxes beyond that they are bijective, the length of impossible differentials for PRESENT-48/64/96 increase from 2/2/3 round to 6/6/6 round

Automatic Search of Impossible Differentials

——Algorithm complexity

- Complexity

1. Build the difference propagation system of a block cipher;
2. **for** *each pair of* $(\Delta P, \Delta C)$ *we choose* **do**;
3. *index*:=true;
4. **while** *index* **do**;
5. Predict information by solving the difference propagation system;
6. **if** *a contradiction is found* **then**;
7. **return** true; **break**;
8. **elseif** *cannot get any new information* **then**;
9. *index*:=false;
10. **end if**;
11. **end while**;
12. **end for**;

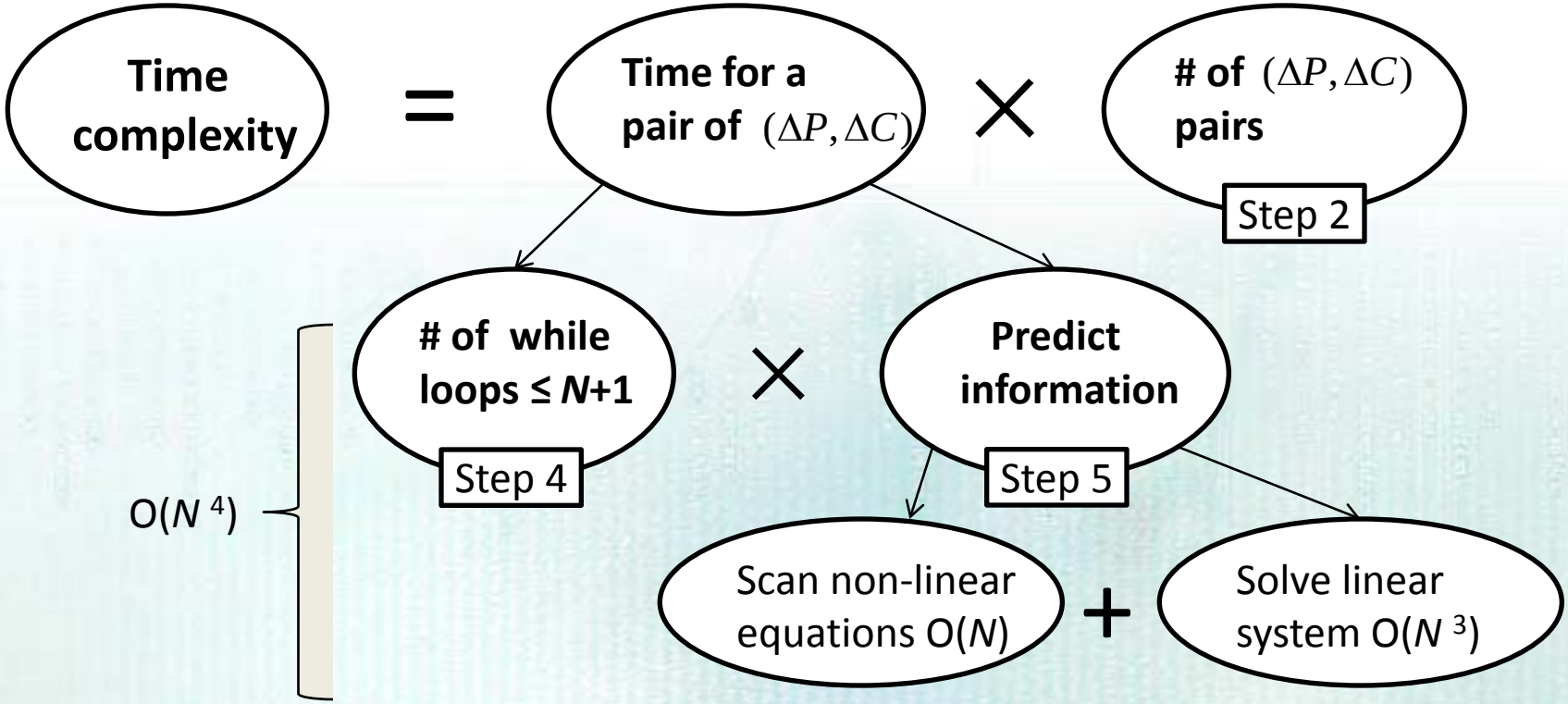
Memory complexity: store the difference propagation system and pre-computed table.

Automatic Search of Impossible Differentials

——Algorithm complexity

2. **for** *each pair of* $(\Delta P, \Delta C)$ *we choose* **do**;
4. **while** *index* **do**;
5. Predict information by solving the difference propagation system;

Suppose N is the number of variables involved in a difference propagation system.



Outline

- Introduction
- Automatic Search of Impossible Differentials
- Automatic Search of Zero-Correlation Linear Hulls
  - Sketch of the tool
  - Linear mask propagation system
- Discussions and Conclusions

Automatic Search of Zero-Correlation Linear Hulls

——Zero-correlation linear hulls

- The theorem of correlation matrix (Daemen, FSE'94)

$$C(\Gamma c \cdot E_k(x), \Gamma p \cdot x) = \sum_U C_U = \sum_{\Gamma p = \alpha_0, \alpha_1, \cdots \alpha_r = \Gamma c} \prod_{i=1}^{r} C(\alpha_i^t \cdot f_i(x), \alpha_{i-1}^t \cdot x)$$

  - $\Gamma p \to \Gamma c$ is a ZCLH if
    - The sum of correlations of all linear trails is zero.
    - All linear trails are unbiased.（√）
- In key-alternating block ciphers

$$C(\Gamma c \cdot E_k(x), \Gamma p \cdot x) = \sum_U (-1)^{d_k} \mid C_U \mid$$

  - The key addition layer can be omitted.

Automatic Search of Zero-Correlation Linear Hulls

——Sketch of searching ZCLHs

- Sketch of searching ZCLHs
  - Build a linear mask propagation system to catch all linear trails with bias
  - A ZCLH is obtained if the corresponding linear mask propagation system has no solution.
  - The techniques of predicting information and detecting contradictions are similar to those of finding impossible differentials.

Automatic Search of Zero-Correlation Linear Hulls

——Algorithm for searching ZCLHs

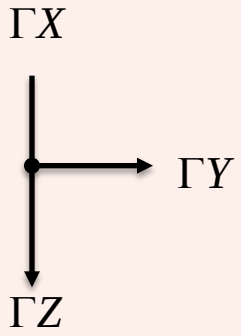
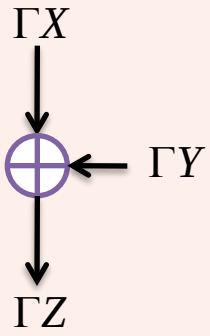
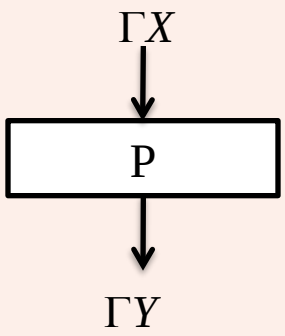
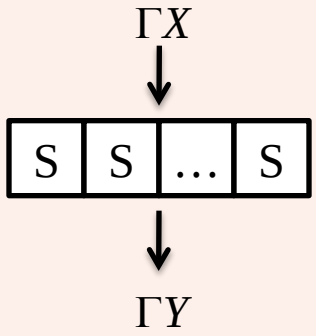
1. Build the linear mask propagation system of a block cipher;
2. **for** *each pair of* $(\Gamma P, \Gamma C)$ *we choose* **do**;
3. *index*:=true;
4. **while** *index* **do**;
5. Predict information by soling the linear mask propagation system;
6. **if** *a contradiction is found* **then**;
7. *index*:=false; **return** true;
8. **elseif** *cannot get any new information* **then**;
9. *index*:=false;
10. **end if**;
11. **end while**;
12. **end for**;

Algorithm 1. Algorithm for searching ZCLHs

Automatic Search of Zero-Correlation Linear Hulls

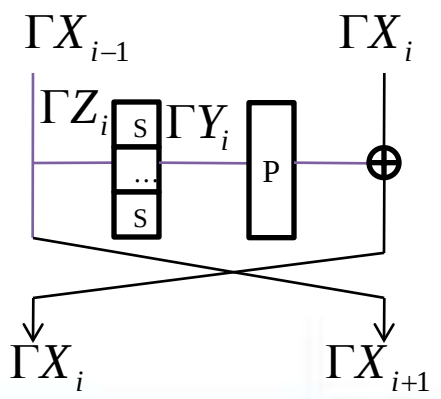
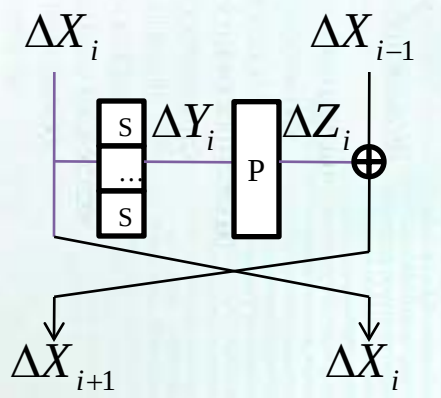
——Linear mask propagation system

- Suppose $\Gamma X = (\Gamma x_1, \Gamma x_2, \ldots, \Gamma x_n), \Gamma Y = (\Gamma y_1, \Gamma y_2, \ldots, \Gamma y_n)$ and $\Gamma Z = (\Gamma z_1, \Gamma z_2, \ldots, \Gamma z_n)$

Basic primitives				
Equations	$\Gamma x_i \oplus \Gamma y_i \oplus \Gamma z_i = 0$	$\Gamma x_i \oplus \Gamma y_i = 0,$ $\Gamma x_i \oplus \Gamma z_i = 0$	$\Gamma Y^t = (P^t)^{-1} \cdot \Gamma X^t$	$\overline{S}(\Delta x_i, \Delta y_i) = 0$
Note	dual property of the difference propagation and the linear mask propagation		$\Delta Y^t = P \cdot \Delta X^t$	linear distribution table (LDT)

Automatic Search of Zero-Correlation Linear Hulls

——Linear mask propagation system

Block ciphers	Linear mask propagation system	Note
	$\begin{cases} \overline{S}(\Gamma Z_{i,j}, \Gamma Y_{i,j}) = 0, \\ \Gamma Y_i^T \oplus P^t \cdot \Gamma X_i^T = 0, \\ \Gamma X_{i-1} \oplus \Gamma Z_i \oplus \Gamma X_{i+1} = 0 \end{cases}$ $(1 \le i \le r, 1 \le j \le n)$	$N=(3r+2)n$ variables $2rn$ linear equations rn non-linear equations from Sbox layers
	Difference propagation system	
	$\begin{cases} \overline{S}(\Delta X_{i,j}, \Delta Y_{i,j}) = 0, \\ \Delta Z_i^T \oplus P \cdot \Delta Y_i^T = 0, \\ \Delta X_{i-1} \oplus \Delta Z_i \oplus \Delta X_{i+1} = 0 \end{cases}$ $(1 \le i \le r, 1 \le j \le n)$	$N=(3r+2)n$ variables $2rn$ linear equations rn non-linear equations from Sbox layers

Outline

- Introduction
- Automatic Search of Impossible Differentials
- Automatic Search of Zero-Correlation Linear Hulls
- Discussions and Conclusions

Discussions and Conclusions

——Discussions

- Correctness and effectiveness
  - IDs and ZCLHs found by our tool must be correct
  - Finds the longest IDs and ZCLHs of many block ciphers
  - Besides Sbox based designs, the idea can be extended to other designs, e.g., ARX block ciphers.
- Although our tool is more powerful than previous tools, it may also miss some longer IDs and ZCLHs
  - Only partially solve the difference/linear mask propagation system
  - Enumerating all possible pairs of $(\Delta P, \Delta C)$ and$(\Gamma p, \Gamma c)$ is beyond our computational ability
  - May miss some other conditions for detecting contradictions

Discussions and Conclusions

——Conclusions

- Conclusions
  - A new tool for searching IDs and ZCLHs is introduced.
  - It helps in closing the gap between previous tools and ad hoc approaches
  - For designers : evaluate the security of their block ciphers against impossible differential cryptanalysis and zero-correlation linear cryptanalysis
  - For attackers: find longer or new IDs and ZCLHs, which may improve known attacks.
- Future work
  - Use more information of nonlinear equations or find more general filtering conditions to detect contradictions

Thank you for your attention!