Algebraic Techniques in Linear Cryptanalysis

Hongru Wei[1,2] Yafei Zheng[1]
[1]School of Mathematics and Physics, University of Science and Technology Beijing, Beijing，100083
[2]State Key Laboratory of Information Security，Beijing，100083
e-mail: weihr168@yahoo.com.cn

***Abstract*—Linear cryptanalysis is a statistical analysis method. Linear cryptanalysis constructs probabilistic patterns first and then distinguishes the cipher from a random permutation using lots of plaintext-ciphertext pairs. Linear cryptanalysis has a big data complexity. Algebraic attack attempts to exploit the algebraic structure of the cipher by expressing the encryption transformation as a set of polynomial equations and then attempts to solve the system to recover the encryption key. Algebraic attacks do not need too much data. This paper combines these two methods by using algebraic techniques in linear cryptanalysis, and proposes a new cryptanalysis method called Algebraic Techniques in Linear Cryptanalysis. This new method is used in the existing linear cryptanalysis of PRESENT. To recover 8-bit key information of 21-round PRESENT, the data complexity is 2^6 and the time complexity is 2^6 .Compared with the result of linear cryptanalysis, the data complexity is obviously decreased.**

Keywords- linear cryptanalysis; algebraic attack; data complexity

I. INTRODUCTION

A new cryptanalytic method against block cipher “Algebraic Techniques in Linear Cryptanalysis” is proposed by combining both algebraic attack and linear cryptanalysis. The new method is applied to the lightweight block cipher PRESENT. Similar to Algebraic Techniques in Differential Cryptanalysis, the new method constructs probabilistic patterns based on the algebraic structure of the algorithm and the linear trails, and then tries to recover the key by solving the system. In this way, the data complexity will be decreased obviously. This paper is structured as follows. First, the block cipher PRESENT is briefly described in section 2. Then linear cryptanalysis and algebraic attack are described in section 3.Section 4 gives the details of the new cryptanalytic method. Section 5 describes the application of the new attack against reduced round PRESENT. And a brief summary is presented in section 6.

II. BLOCK CIPHER PRESENT

The block cipher PRESENT was proposed by Bogdanov et al. at CHES 2007 as an ultra-lightweight block cipher[1]. PRESENT is an SP-network with a blocksize of 64 bits and consists of 31 rounds. Two key sizes of 80 and 128 bits are supported. Each round of the cipher has three layers: KeyAddLayer, SboxLayer and Player. In every round, a single 4-bit S-box is applied 16 times in parallel.

AddRoundKey. Given round key $K = k_{63}^i \cdots k_0^i$, $1 \le i \le 32$ and the current state $b_{63} \ldots b_0$,AddRoundKey consists of the operation for $0 \le j \le 63$, $b_j \to b_j \oplus k_j^i$.

SboxLayer. The S-box of PRESENT is a 4-bit to 4-bit S-box : $F_2^4 \to F_2^4$.

Player. The bit permutation is given by that bit i is moved to bit position $P(i)$.

KeySchedule. PRESENT can take keys of 80 or 128 bits. Here the key schedule PRESENT-80 is focused. The user-supplied key is stored in a key register K and represented as $k_{79}k_{78}\cdots k_0$. At round i the 64-bit round key $K_i = k_{63}k_{62}\cdots k_0$ consists of the 64 leftmost bits of the current contents of register K. After extracting the round key K_i , the key register $K = k_{79}k_{78}\ldots k_0$ is updated as follows.

a. $[k_{79}k_{78}\cdots k_0] = [k_{18}k_{17}\cdots k_{20}k_{19}]$

b. $[k_{79}k_{78}k_{77}k_{76}] = S[k_{79}k_{78}k_{77}k_{76}]$

c. $[k_{19}k_{18}k_{17}k_{16}k_{15}] = [k_{19}k_{18}k_{17}k_{16}k_{15}] \oplus round\_counter$

III. BRIEF INTRODUCTIONS OF LINEAR CRYPTANALYSIS AND ALGEBRAIC ATTACK

A. Linear cryptanalysis

Linear cryptanalysis[2][3] is a known plaintext attack. The basic idea is recovering some key bits by using the unbalanced linear approximations between plaintext, ciphertext and key of the cipher. Firstly, linear cryptanalysis needs to find an “effective” linear expression of a given cipher as follows: $P_{[i_1,i_2\ldots i_a]} \oplus C_{[j_1,j_2\ldots j_b]} = K_{[k_1,k_2\ldots k_c]}$, here $i_1, i_2 \ldots i_a, j_1, j_2 \ldots j_b, k_1, k_2 \ldots k_c$ are fixed bits. For randomly given plaintext P and ciphertext C , the probability of the equation is $p \ne 1/2$ .and bias $\varepsilon = |p - 1/2|$ gives the validity of the equation and is called the advantage of the approximation. To find an effective linear approximation, the statistical method is used to give some linear approximations of the input and the output of the main part of the round function. After connecting the approximations of each round and eliminating the intermediate variables, approximations called linear trails which only involve the plaintext、the ciphertext and the key will be received. If an effective linear expression is already known, then a key bit can be tested as follows:

(1) Let T be the number of the plaintexts which satisfy that the left of the equation equals to 0.

(2)If $T > N/2$ (N is the number of plaintexts), then when $p > 1/2$, $K_{[k_1 k_2 \ldots k_c]} = 1$; when $p < 1/2$, $K_{[k_1 k_2 \ldots k_c]} = 0$.

A linear approximation $\alpha \to \beta$ of an iterated block cipher is called a linear hull. A linear hull includes all the linear approximations which has input α and output β. When there is only one linear trail between a given pair, linear trail is the same with linear hull. A linear hull is a cluster of linear trails and its approximation advantage will be larger than any single linear trail. Given input mask a and output mask b and cipher $Y = Y(X, K)$, the approximation advantage of a linear hull is $ALH(a,b) = \sum_c (p(a \cdot X \oplus b \cdot Y \oplus c \cdot K = 0) - 1/2)^2 = \varepsilon^2$. And c is the key mask. To recover the key, $N = t / ALH(a,b) = t/\varepsilon^2$ known plaintexts are needed, t is a constant.

B. Algebraic attack

The basic idea of algebraic attack [3][4] can be described as follows:(1)express the cipher as simple equations of several variables and these variables can be some fixed bits(or bytes) of plaintext、ciphertext and the key, or some bits of the intermediate state and the round subkeys;(2) substitute the equations in (1) with the collected plaintext-ciphertext pairs and try to solve the equations to recover the key.

IV. ALGEBRAIC TECHNIQUES IN LINEAR CRYPTANALYSIS

When algebraic attack is applied to a cipher, firstly the cipher will be expressed into a system of equations, then by solving the system some related key bits can be recovered. In order to make the system easy to solve, the problem how to increase the number of linear equations is considered [5].

Assume a equation system F' can be constructed from a single plaintext-ciphertext pair (P', C'), and then under the same encryption key, F'' can be constructed from another pair (P'', C''). Combining F' with F'' : $F = F' \cup F''$. The number of linear equations is increased in this way. But even the initial key and the key strategy of F' and F'' are the same, the intermediate variables are not absolutely the same. That is to say, while with the number of the equations is increased, the number of unknown variables in the system is also increased [5]. So another method is considered by applying the characteristic of linear approximations in Method-1.

A. Method-1

Assume the target cipher is an example of SP-network, and a linear trail $\Delta = (\delta_0, \delta_2, \ldots \delta_r)$ is obtained. $\delta_{i-1} \to \delta_i$ means one round approximation with probability $p_i = 1/2 + \varepsilon_i$. According to the Piling-Up Lemma of Matsui [2], approximations of each round can be considered as mutually independent, so the probability of Δ can be $P = 1/2 + 2^{r-1} \prod_{i=1}^{r} \varepsilon_i$.

Let X_{ij}^m Y_{ij}^m be the input and output of the m th active S-box in the i th round, then system $\overline{F}$ is constructed. For each active S-box related to the known linear trail:

$$\sum_{j=0}^{s-1} X_{ij}^m \oplus \sum_{j=0}^{s-1} Y_{ij}^m = 0 \ , \quad i = 1,2 \ldots r \ , \quad j = 0,1 \ldots s-1 \ , \quad m = 1,2 \cdots m_i$$

m_i is the number of active S-boxes in the i th round. The probability of the above equation is $p_{im} = 1/2 + \varepsilon_{im}$

Then from n active S-boxes in the whole linear trail, $n = \sum_{i=0}^{r} m_i$ new equations can be achieved, and these n equations constitute $\overline{F}$.

Set $F = F' \cup \overline{F}$, and $\overline{F}$ includes all the equations achieved from the related active S-boxes. The probability of F (equal to the probability of $\overline{F}$) is:

$$P = (1/2 + \varepsilon_{11}) \ldots (1/2 + \varepsilon_{1m_1})(1/2 + \varepsilon_{21}) \ldots (1/2 + \varepsilon_{1m_2}) \ldots (1/2 + \varepsilon_{r1}) \ldots (1/2 + \varepsilon_{1m_r})$$

$$= \prod_{i=1}^{r} \prod_{j=0}^{m_j} (1/2 + \varepsilon_{ij}) \approx 2^{-\sum_{i=0}^{r} m_i} = 2^{-n}$$

n is the number of all active S-boxes that the linear trail involves.

Then $1/P \approx 2^{-n}$ pairs of plaintext and ciphertext can be used to solve the above system F, and a non-empty solution can be expected. From the algebraic point of view, F is easier to be solved than F'. Since unlike F'', $\overline{F}$ increases the number of linear equations without adding new variables to F'. However, to solve such a big system $1/P$ times directly is also very difficult, so the following method-2 considers another way to recover the key to avoid this problem.

B. Method-2

Still assume the cipher is an example of SP-network. A linear trail $\Delta = (\delta_0, \delta_2, \ldots \delta_r)$ is known with same probability P. A plaintext-ciphertext pair (P, C) is called a right pair if it satisfies the linear trail Δ. For the sake of simplicity, the following analysis is based in the assumption that there is only one active S-box in the first round. And its input is X_{1j} and output is Y_{1j}. Then

$$X_{1j} = P_j \oplus K_{0j} \ , Y_{1j} = S(X_{1j}) = S(P_j \oplus K_{0j}) \ , \quad j = 0,1 \ldots s-1 \ .$$

If (P, C) is a right pair, there must be:

$$\sum_{j=0}^{s-1} X_{1j} \oplus \sum_{j=0}^{s-1} Y_{1j} = \sum_{j=0}^{s-1} (P_j \oplus K_{0j}) \oplus \sum_{j=0}^{s-1} S(P_j \oplus K_{0j}) = 0$$

And $X_{1j} Y_{1j}$ are not zero at the same time.

The above equation only has unknown variables K_{0j}. So on the premise of a known right pair, a small system can be set up for each active S-box in the first round to recover some information of the first round key. And similarly, this method can be used to recover some information of the last round key. Let X_{rj} be the input and Y_{rj} be the output of the last round, then there is:

$$\sum X_{rj} \oplus \sum Y_{rj} = 0$$

Set up the small system, and some information of the last round key can be recovered by solving the system.

The premise of Method-2 is a known right pair of the linear trail, so the next task is to consider how such a right pair can be obtained.

Here system $F = F' \cup \overline{F}$ in Method-1 can be a filter. t_1 is denoted as the time to judge a right pair(determined by experiment) , and the time to judge a wrong pair is no more than t_1 . A pair (P,C) is substituted into the system F . If in time t_1 , there is no conflict, (P,C) will be a right pair of the linear trail. By this way, after a right pair is obtained, the small system in Method-2 can be constructed to recover the key information.

Observe the process of the key recovery in Method-2, it only related to the approximations of the active S-boxes in the first and last round, but has no relation to the intermediate states of the linear trail Δ . This characteristic is similar to linear hull. So the linear trails will be replaced by linear hull in constructing the filtration system.

Assume an r-round linear hull $\delta_0 \to \delta_r$ and a corresponding right pair is already obtained. Then similarly the small system in Method-2 can be used to recover the key.

Let $X_{1j}Y_{1j}$ be the input and output of the S-boxes in the i th round and then there is:

$$\begin{cases} \sum_j X_{1j} \oplus \sum_j Y_{1j} = 0 \\ \sum_j X_{rj} \oplus \sum_j Y_{rj} = 0 \end{cases} \xrightarrow[Y_{rj}=C_j \oplus K_{rj}]{X_{1j}=P_j \oplus K_{0j}} \begin{cases} \sum_j (P_j \oplus K_{0j}) \oplus \sum_j S(P_j \oplus K_{0j}) = 0 \\ \sum_j S^{-1}(C_j \oplus K_{rj}) \oplus \sum_j (C_j \oplus K_{rj}) = 0 \end{cases}$$

The above system is denoted as $\widetilde{F}$. The probability of $\widetilde{F}$ is about $(1/2)^{n_1+n_2}$, n_1 , n_2 are the numbers of active S-boxes in the first an last round respectively, and $n_1 + n_2 <= n$. Construct system $F = F' \cup \widetilde{F}$, and F can be a filtration system. In F , $\widetilde{F}$ is much more simple than $\overline{F}$ in Method-1. Precisely, $\widetilde{F}$ is achieved from $\overline{F}$ by removing the equations corresponding to the active S-boxes in the middle round. And the equations corresponding to the last and first round active S-boxes constitutes $\widetilde{F}$ .The probability of $\widetilde{F}$ is about $(1/2)^{n_1+n_2}$, larger than $(1/2)^n$ in Method-1. That is to say, after $(1/2)^{n_1+n_2}$ times filtering operations, rather than $(1/2)^n$ times in linear trail situation, a right pair can be expected. The decrease of filtering is the advantage of using linear hull instead of linear trails. At the same time, the number of equations in the filtration system is also decreased, then the time to judge a right pair (denoted as t_2)will be shorter than t_1 .

V. APPLY THE NEW METHOD TO PRESENT

A. A single linear trail

There is a 21-round linear trail of PRESENT-80 [4] :

$$0000000000A00000 \xrightarrow{1r} 0000000000200000 \xrightarrow{1r} 0000000000200000 \cdots \xrightarrow{19r} 0000000000200000$$

According to Matsui's Piling-up Lemma, the bias of this trail is $\varepsilon = 2^{-42}$. 22 active S-boxes are involved. System F in Method-1 can be constructed and its probability is about $P \approx 2^{-22}$. So after 2^{22} times filtering of plaintext and ciphertext pairs, a right pair (P,C) satisfying this linear trail can be obtained.

Construct the small system in Method-2 for active S-box S_{10} in the first round:

$$\begin{cases} X_0 = P_0 \oplus K_0 \\ X_1 = P_1 \oplus K_1 \\ X_2 = P_2 \oplus K_2 \\ X_3 = P_3 \oplus K_3 \\ Y_0 = X_0X_1X_3 + X_0X_2X_3 + X_0 + X_1X_2X_3 + X_1X_2 + X_2 + X_3 + 1 \\ Y_1 = X_0X_1X_3 + X_0X_2X_3 + X_0X_2 + X_0X_3 + X_0 + X_1 + X_2X_3 + 1 \\ Y_2 = X_0X_1X_3 + X_0X_1 + X_0X_2X_3 + X_0X_2 + X_0 + X_1X_2X_3 + X_2 \\ Y_3 = X_0 + X_1X_2 + X_1 + X_3 \\ X_0 + X_1 + X_2 + X_3 + Y_0 + Y_1 + Y_2 + Y_3 = 0 \text{ (added by the linear trail)} \end{cases}$$

$X_0X_1X_2X_3$ are 4 bits input and $Y_0Y_1Y_2Y_3$ are the 4 bits output of S_{10} . The above system can be simply denoted as

$$\sum X + \sum Y = f(k_0, k_1, k_2, k_3) = 0 .$$

There are $2^4 = 16$ possible key candidates, set a counter for every candidate. If the key satisfies

$$\sum X + \sum Y = f(k_0, k_1, k_2, k_3) = 0$$

its corresponding counter plus 1.

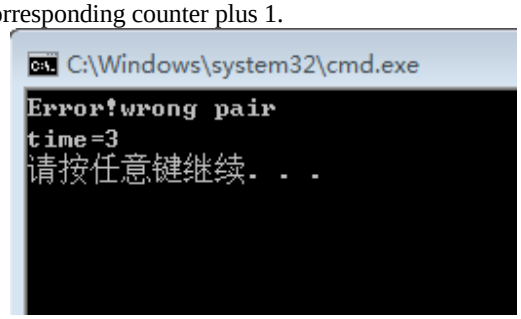

Fig 1. Right pair detection in linear trails

In a personal computer under the programming environment of visual studio 2012, the time to judge a right pair by using the above filtration system can be $t_1 <= 10s$.

When a right pair $(0000000000A00000, 0000000000200000)$ is obtained, the small system can be constructed for S_{10} in the first round, and simplify it: $\sum X \oplus \sum Y = f(k_0, k_1, k_2, k_3)$. Substitute each key candidate into this equation and its counter plus 1 when it matches the conditions. Repeat this process m times, and take the key relevant to the peak counter as the right key. For example, let $m = 8$, then the counter relevant to the right key should be 8, and the probability of taking the wrong candidates is $2^{-8} < 0.01$. Under the same programming environment, the time to judge a right candidate is $t_3 <= 1s$. Then $8s$ are needed to detecting the key.

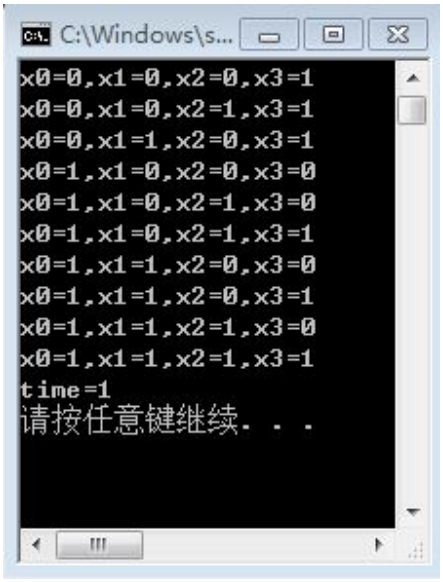

Fig 2. Key detection

The 4 bits information of the first round key K_0 can be recovered by using this attack. The data complexity is $8\times 2^{22}=2^{25}$, and the time complexity is

$$m\times t_1\times 2^{22}+m\times t_3\times 2^4 <= 2^{25}\times 10+2^7\approx 2^{28.32}.$$

If m is bigger, the probability of taking the wrong key candidates will be much small, and the attack will be more accurate.

On the basis of the above attack, 4 bits information of the last round key can also be recovered. Then in all, 8 bits of the key can be recovered. The whole data complexity is still 2^{25}, and the time complexity is

$$T=m\times t_1\times 2^{22}+m\times t_3\times 2^4+m\times t_4\times 2^4 <= 2^{28.32}$$

t_4 is the time needed for a single key detection in the last round. t_4 is negligible compared to t_1. The remaining 72 bits can be obtained by using other linear trails, or be obtained by exhaustive search.

B. Linear hull

Taking the linear hull $(0000000000200000, 0000000000200000)$ instead of the linear trail [4]:

$$0000000000A00000 \xrightarrow{1r} 0000000000200000 \xrightarrow{20r} 0000000000200000$$

The system F constructed under the above hull has the probability about $(1/2)^3$. So after 2^3 times detections of plaintext and ciphertext pairs, a right pair satisfying the linear hull is expected. The next step will be constructing the small system for S_{10} in the first round and S_{10} in the last round and finally recover some key information. This process is actually the same with the case of a single linear trail. However, the time for detection of right pairs denoted as t_2 will be smaller. According to experiment, $t_2 <= 1$. When $m=8$, the data complexity is $m\times 2^3=2^6$, and the time complexity is $T=2^3\times m\times t_2+2^4\times m\times t_3+2^4\times m\times t_4\approx 2^8$.

The above results also hold with PRESENT-128.

VI. SUMMARY OF THIS PAPER

Based on the study of linear cryptanalysis and algebraic attack, and algebraic techniques in differential cryptanalysis, a new attack "Algebraic Techniques in Linear Cryptanalysis" is proposed. And the new method is applied to 21-round PRESENT-80. Under linear trail and linear hull, to recover 8 bits of the key, the data complexities are 2^{25}, 2^6 respectively; and the time complexities are $2^{28.32}$, 2^8 respectively. Compared to the results under traditional cryptanalysis, both the data complexity and the time complexity are obviously decreased (TABLE Ⅰ).

TABLE I. THE CRYPTANALYSIS RESULTS

Key size	round	Key bits	Data complexity	Time complexity	cryptanalysis
80/128	21	8	2^{78}	$2^{78.6}$	LC (trail)
80/128	21	4	$2^{69.3}$	$2^{64.9}$	LC (hull)
80/128	21	8	2^{25}	$2^{28.32}$	This paper (trail)
80/128	21	8	2^6	2^8	This paper (hull)

ACKNOWLEDGMENT

This work was supported by the Oriented Award Foundation for Science and Technological Innovation, Inner Mongolia Autonomous Region, China(2012)and Foundation of State Key Laboratory of Information Security (Institute of Software, Chinese Academy of Sciences) (02-04-3).

REFERENCES

[1] A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar1, A. Poschmann, M.J.B. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher.CHES 2007, LNCS 4727：450~466.

[2] Howard M.Heys. A Tutorial on Linear and Differential Cryptanalysis. CORR 2001-17.

[3] Wu wenling, Feng Dengguo, Zhang Wwentao. Design and analysis of block cipher. Beijing: Tsinghua University press.2009 (in Chinese)

[4] Jorge Nakabara Jr, Pouyan Sepehrdad, Bingsheng Zhang, Meiqin Wang.Linear (Hull) and Algebraic Cryptanalysis of the Block Cipher PRESENT. Cryptology and Network Security, 8th International Conference, CANS 2009:1~18.

[5] Martin Albrecht, Carlos Cid. Algebraic Techniques in Differential Cryptanalysis. FSE 2009.