The cube attack on stream cipher Trivium and quadraticity tests

Piotr Mroczkowski Janusz Szmidt
Military Communication Institute
Poland

10 czerwca 2010

Outline

1. Papers and Preprints
2. Cube Attacks
   - Mathematical background
   - The structure of the attack
3. Trivium - specification
4. Cube Attack on Trivium
5. Conclusion

Cube Attack - Papers and Preprints

- Itai Dinur and Adi Shamir. **"Cube Attacks on Tweakable Black Box Polynomials"**, Eurocrypt, 2009
- Michael Vielhaber. **"Breaking One. Fivium by AIDA an Algebraic IV Differential Attack"**, IACR Cryptology ePrint Archive, 2007.
- J-P. Aumasson, W. Meier, I. Dinur, A. Shamir. **"Cube testers and key recovery attacks on reduced round MD6 and Trivium"**, Fast Software Encryption, 2009.
- I. Dinur, A. Shamir. **"Side channel cube attacks on block ciphers"**, IACR Cryptology ePrint Archive, 2009/127.
- P. Mroczkowski, J. Szmidt. **The Cube Attack on Courtois Toy Cipher**, IACR Cryptology ePrinf Archive, 2009/497.

- $p(x_1, \ldots, x_n)$ - the polynimial of n variables over $GF(2)$;
- $I = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$ - a subset of indexes of the variables of p;
- $t_I = x_{i_1} \ldots x_{i_k}$ - a monomial of k variables, such that $i_1, \ldots, i_k \in I$.

Then we have a decomposition:

$$p(x_1, \ldots, x_n) = t_I \cdot p_{S(I)} + q(x_1, \ldots, x_n)$$

- $p_{S(I)}$ - the superpoly of I in p which does not depend on the variables $x_{i_1}, \ldots, x_{i_k}$;
- $q(x_1, \ldots, x_n)$ - the remainder of I in p.

The maxterm of the polynomial p we call the monomial t_I, such that:

$$deg(p_{S(I)}) = 2$$

or

$$deg(p_{S(I)}) = 1$$

It means that the polynomial $p_{S(I)}$ corresponding to the subset of indexes I is a quadratic or linear one, which is not a constant.

Let:

- $I = \{i_1, \ldots, i_k\} \subset \{1, \ldots, n\}$ - a fixed subset of k indexes;
- $C_I = \{0,1\}^k$ - the k-dimensional Boolean cube, where on the place of each of the indexes we put 0 or 1;
- $v \in C_I$ - a given vector which defines the derived polynomoal p_v depending on $n-k$ variables, where in the basic polynomoal p we put the values corresponding to the vector v.

Summing over a cube - summing all vectors in the cube C_I we obtain the polynomial:

$$p_I = \sum_{v \in C_I} p_v$$

For any polynomial p and set of indexes I we have: **Theorem**

$$p_I = p_{S(I)} \mod 2$$

- Let us consider cryptosystem described by the function:

$$p(v, x)$$

- $v = (v_1, \dots, v_m)$ - m public variables (the initial value or plaintext);
- $x = (x_1, \dots, x_n)$ - n secret variables (the key);
- $p(v_1, \dots, v_m, x_1, \dots, x_n)$ - the polynomial describing the cryptosystem:
  - p is not explicitelly known (*a black box*);
  - the value of p represents the ciphertext bit.
- We will consider the known plaintext attack, where at the preprocessing stage the attacker has also an access to secret variables.

1. **The preprocessing stage**
   - The attacker can change the values of public and secret variables.
   - The task is to obtain a system of quadratic and linear equations on secret variables.
2. **The stage *on line* of the attack** - the key is secret now.
   - The attacker can change the values of public variables.
   - The task is to obtain the right hand sides of equations.
   - The system of equation can be solved giving some bits of the key.

The aim - **to create the system of quadratic and linear equations on secret variables**.

- The first task of this stage of attack is:
  - to fix a dimension of the cube;
  - to fix the public variables over which we will sum (*the tweakable variables*);
  - to put zero to the other public variables.
- We do the summation over a fixed cube for several values of secret variables and collect the obtained values.
- We do the quadraticity and linear tests for the obtained function of secret variables and store it when it is quadratic or linear one.

Quadraticity and linear tests

- **Quadraticity test for boolean function:**

$$f(x \oplus x' \oplus x'') = f(x \oplus x') \oplus f(x \oplus x'') \oplus f(x' \oplus x'')$$

$$\oplus f(x) \oplus f(x') \oplus f(x'') \oplus f(0)$$

for arbitrary vectors x, x', x'' of n boolean variables.

- **Linearity test for boolean function:**

$$f(x \oplus x') = f(x) \oplus f(x') \oplus f(0)$$

for arbitrary vectors x, x' of n boolean variables.

- The next task is to calculate the exact form (the coefficients) of the obtained quadratic or linear function of secret variables.
  - The free term of the linear function we obtain putting its all argument equal zero.
  - The coefficient of the variable x_i is equal 1 if and only if the change of this variable implies the change of values of the function.
  - The coefficient of the variable x_i is equal 0 if and only if the change of this variable does not imply the change of values of the function.

- This system of quadratic and linear equations will be used in the *on line* stage of attack.
- The preprocessing stage is done only once in cryptanalysis of the algorithm.

Cube Attack - The stage *on line* of attack

The aim - **to find some bits of secret key** with complexity lower than the exhuastive search in the brute force attack.

- In this stage an attacker has:
  - the access only to public variables (the plaintext for block ciphers, the initial values for stream ciphers);
  - the system of quadratic and linear equations, obtained in the preproccesing stage.
- An attacker
  - changing public variables calculates the corresponding bits of the ciphertext under the unknown value of secret variables;
  - using the derived system of equations for secret variables, where the right hand sides of these equations are the values of bits of ciphertext obtained after summation over the same cubes as in the preprocessing stage, calculates the secret variables (the unknown bits of the key).

- Algorithm Trivium (the authors: C. de Canniere and B. Preneel) is one of the finalists of eSTREAM competitions.
- The basic parameters are the 80-bit key and the 80-bit initial value.
- The inner state of Trivium are 288 bits loaded to three nonlinear registers of different lengths.
- In each round of the algorithm the registers are shifted on one bit.
- The feedback in each register is given by a nonlinear function.

27. The specification of Trivium

$$(s_1, s_2, \ldots, s_{93}) \leftarrow (k_1, k_2, \ldots, k_{80}, 0, \ldots, 0)$$

$$(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (IV_1, IV_2, \ldots, IV_{80}, 0, \ldots, 0)$$

$$(s_{178}, s_{179}, \ldots, s_{288}) \leftarrow (0, 0, \ldots, 0, 1, 1, 1)$$

for i = 1 to 1152

$$t_1 \leftarrow s_{66} + s_{93}$$

$$t_2 \leftarrow s_{162} + s_{177}$$

$$t_3 \leftarrow s_{243} + s_{288}$$

$$t_1 \leftarrow t_1 + s_{91} \cdot s_{92} + s_{171}$$

$$t_2 \leftarrow t_2 + s_{175} \cdot s_{176} + s_{264}$$

$$t_3 \leftarrow t_3 + s_{286} \cdot s_{287} + s_{69}$$

28. The specification of Trivium, cont.

$$(s_1, s_2, \ldots, s_{93}) \leftarrow (t_3, s_1, \ldots, s_{92})$$

$$(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (t_1, s_{94}, \ldots, s_{176})$$

$$(s_{178}, s_{179}, \ldots, s_{288}) \leftarrow (t_2, s_{178}, \ldots, s_{287})$$

end for

29. The specification of Trivium, cont.

- The generation of the ouput bitstring (z_i) of the maximal length up to $N = 2^{64}$ bits, can be represented as:
  for i=1 to N

$$t_1 \leftarrow s_{66} + s_{93}$$

$$t_2 \leftarrow s_{162} + s_{177}$$

$$t_3 \leftarrow s_{243} + s_{288}$$

$$z_i \leftarrow t_1 + t_2 + t_3$$

$$t_1 \leftarrow t_1 + s_{91} \cdot s_{92} + s_{171}$$

$$t_2 \leftarrow t_2 + s_{175} \cdot s_{176} + s_{264}$$

$$t_3 \leftarrow t_3 + s_{286} \cdot s_{287} + s_{69}$$

30. The specification of Trivium, cont.

$$(s_1, s_2, \dots, s_{93}) \leftarrow (t_3, s_1, \dots, s_{92})$$
$$(s_{94}, s_{95}, \dots, s_{177}) \leftarrow (t_1, s_{94}, \dots, s_{176})$$
$$(s_{178}, s_{179}, \dots, s_{288}) \leftarrow (t_2, s_{178}, \dots, s_{287})$$

end for

31. The cube attack on Trivium

- Dinur and Shamir investigated the reduced versions of Trivium which consit 672 and 735 (instead of 1152) initialization rounds.
- During the preprocessing stage they obtained 63 linearly independent maxterms corresponding to 12-dimensional cubes and output bits of the indices from 672 to 685.

32. The cube attack on Trivium, cont.

- In the *on line* stage the attacker must find the values of the maxterms summing over 63 12-dimensional cubes.
- After solving the system of linear equations the attacker obtains 63 bits of the key and the remaining 17 bits of the key are found by brute force search.
- The complexity of the attack (in the on line stage) is ca. 2^{19} evaluatioins of the investigated, reduced algorithm. It is smaller than the complexity 2^{55} in the previous attacks on this version of Trivium.

Quadratic equations

Examples of quadratic expressions obtained for reduced version of Trivium after 650 initialization rounds:

$$x40x41 + x15 + x42, c3 = 1$$

$$x77x78 + x22 + x52 + x79 + 1, c0 = 0$$

$$x44x65 + 1, c1 = 1 \implies x44 = 0 \vee x65 = 0$$

$$x47x48 + x22 + x49 + 1, c2 = 1$$

$$x53x54 + x28 + x55, c2 = 0$$

$$x22x23 + x24 + x66 + 1, c0 = 0$$

$$x11x67 + x11, c0 = 0 \implies x11 = 0 \vee x67 = 1$$

ci is the index of output bit after 650 initialization rounds.

- Developing a new information technology:
- Very fast parallel implementation of Trivium - 128 independent key streams (Paul Crowley)
- Programming in assembler using language Python

Thank you