Analysis of chosen plaintext attacks on the WAKE Stream Cipher

Marina Pudovkina

maripa@online.ru

Moscow Engineering Physics Institute (Technical University)
Department of Cryptology and Discrete Mathematics

Abstract. Stream ciphers are an important class of encryption algorithms, which are widely used in practice. In this paper the security of the WAKE stream cipher is investigated. We present two chosen plaintext attacks on this cipher. The complexities of these attacks can be estimated as $10^{19.2}$ and $10^{14.4}$.

Keywords. WAKE. Stream Cipher. Cryptanalysis.

1 Introduction

Symmetric cryptosystems can be subdivided into block and stream ciphers. Block ciphers operate with a fixed transformation on large blocks of plaintext data; stream ciphers operate with a time-varying transformation on individual plaintext digits. Typically, a stream cipher consists of a keystream generator whose pseudo-random output sequence is added modulo 2 to the plaintext bits. A major goal in stream cipher design is to efficiently produce random-looking sequences. But the keystream can be generated efficiently; there certainly exists such a simple description.

WAKE is the Word Auto Key Encryption algorithm, invented by David Wheeler [1]. It has a very simple description and produces a stream of 4n-bit words, which can be XORed with a plaintext stream to produce ciphertext, or XORed with a ciphertext stream to produce plaintext. It is fast on most modern computers, and relies on repeated table use and having a large state space. WAKE works in CFB mode; the previous ciphertext word is used to generate the next key word. It is being used in the current version of Dr. Solomon's Anti-Virus program [5]. It is known that WAKE is insecure against a chosen plaintext or chosen ciphertext attack but descriptions and complexities of these attacks are unknown.

The aim of this paper is to describe two chosen plaintext attacks on WAKE that are intrinsic to the design principles of WAKE and are independent of the key scheduling. The complexities of these attacks can be estimated as $2^{4n}+2^{8n}$ for first and $2^{4n}+2^{6n}$ for second. Our algorithms become infeasible for n>8.

The paper is organized as follows. In section 2 we give a description of WAKE. In section 3 we discuss some properties of WAKE. Section 4 describes attacks on WAKE. We conclude in section 5.

2 Description of WAKE

In fact the WAKE stream cipher is a family of algorithms indexed by a positive integer n (in practice n=8). It works in CFB mode; the previous ciphertext word is used to generate the next key word. It

uses a table T={T[0],…,T[2^n-1] } of 2^n 4n-bit words. This table T has a special property: the high-order n-bit of all the entries is a permutation of all possible n-bit words, and the low-order 3n-bit words are random.

The internal state of WAKE at time t consists of four 4n-bit words a_t, b_t, c_t, d_t. The table T and initial a_0, b_0, c_0, d_0 are generated from the key.

Let P=p_1, p_2,…, p_L be a plaintext and Y=y_1, y_2,…, y_L be a ciphertext

The next-state and output functions of WAKE for every t are defined by

The next-state function F

$a_i = M(a_{i-1}, y_{i-1}, T)$

$b_i = M(b_{i-1}, a_i, T)$

$c_i = M(c_{i-1}, b_i, T)$

$d_i = M(d_{i-1}, c_i, T)$

The output-function

$z_i = d_i$

Encryption

$y_i = z_i \oplus p_i$

Dencryption

$p_i = z_i \oplus c_i$

Thus, we have $y_i=F(y_{i-1}, a_{i-1}, b_{i-1}, c_{i-1}, d_{i-1})$.

The transformation M(x, y, T) is an invertible transformation defined by:

$$M(x, y, T) = (x + y) >> n \oplus T[(x + y)(\mathrm{mod}\ 2^n)].$$

This is shown in Figure 1. The operation >> is a right shift, not a rotation.

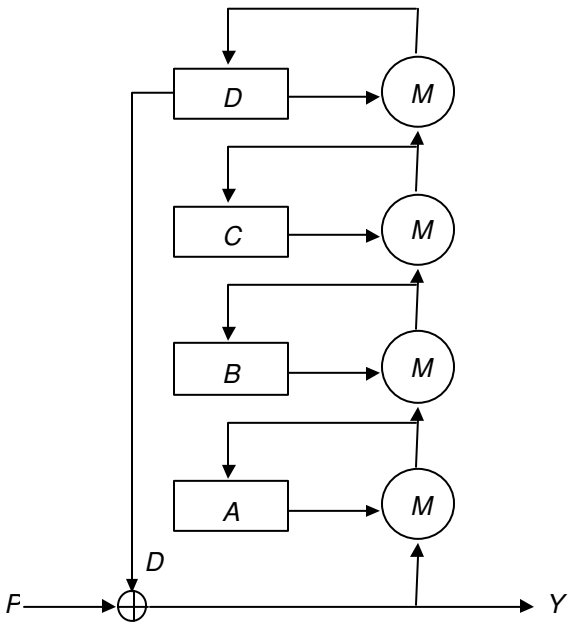

Fig. 1

3 Properties of WAKE

In this section we describe some properties of WAKE, which are important in the description of chosen plaintext attacks on this cipher.

Proposition 1
The transformation $F(k, a_{i-1}, b_{i-1}, c_{i-1}, d_{i-1}, T)$ where $k \in \{0,\ldots,2^{4n}-1\}$ is a permutation on $\{0,\ldots,2^{4n}-1\}$.
Proof.
Note that $M(x, y, T)$, where $y \in \{0,\ldots,2^{4n}-1\}$, is a permutation on $\{0,\ldots,2^{4n}-1\}$ because if

$$(x + y_1) >> n \oplus T[(x + y_1)(\mathrm{mod}\ 2^n)] = (x + y_2) >> n \oplus T[(x + y_2)(\mathrm{mod}\ 2^n)],$$

where $y_1, y_2 \in \{0,\ldots,2^{4n}-1\}$, $y_1 \neq y_2$, then

$$T[(x + y_1)(\mathrm{mod}\ 2^n)] = T[(x + y_2)(\mathrm{mod}\ 2^n)],$$

and

$$(x + y_1) >> n = (x + y_2) >> n.$$

It follows that $y_1 = y_2$.
Note that $F(k, a_{i-1}, b_{i-1}, c_{i-1}, d_{i-1}, T)$ is a composition of permutation and can be represented as follows:

$$F(k, a_{i-1}, b_{i-1}, c_{i-1}, d_{i-1}, T) = M(c_{i-1}, M(d_{i-1}, M(b_{i-1}, M(a_{i-1}, k)))).$$

Therefore, F is a permutation on $\{0,\ldots,2^{4n}-1\}$.
The proposition is proved.

Let $a_i(p_{i-k}, p_{i-k+1},\ldots, p_{i-1})$, $b_i(p_{i-k}, p_{i-k+1},\ldots, p_{i-1})$, $c_i(p_{i-k}, p_{i-k+1},\ldots, p_{i-1})$, $d_i(p_{i-k},\ldots, p_{i-1})$ be meanings of A, B, C, D registers at time i providing that $p_{i-k}, p_{i-k+1},\ldots, p_{i-1}$ have been ciphered. Let $y_i(p_{i-k},\ldots, p_{i-1}, p_i)$ be a i^{th} word of a ciphertext at time i which is obtained after being ciphered $p_{i-k}, p_{i-k+1},\ldots, p_i$.

Proposition 2
Let $\alpha \in Z_{2^n}$.

1. The number of elements j such that $d_i[j] >> 3n = T[\alpha]$ is 2^{3n}.
2. The number of elements j such that $c_i[j] >> 3n = d_i[j] >> 3n = T[\alpha]$ is 2^{2n}.
3. The number of elements j such that $b_i[j] >> 3n = c_i[j] >> 3n = d_i[j] >> 3n = T[\alpha]$ is 2^n.
4. The number of elements j such that $a_i[j] >> 3n = b_i[j] >> 3n = c_i[j] >> 3n = d_i[j] >> 3n = T[\alpha]$ is 1.

Proof.
Let us remark that by proposition 1 $\begin{pmatrix} 0 & \ldots & j & \ldots & 2^{4n}-1 \\ d_i[0] & \ldots & d_i[j] & \ldots & d_i[2^{4n}-1] \end{pmatrix}$ is a permutation of all possible 4n-bit and the high-order n-bit of all the entries of T is a permutation of all possible n-bit
Thus, the number of elements j such that $d_i[j] >> 3n = T[\alpha]$ is $2^{3n} = 2^{4n}/2^n$.
If $c_i[j] >> 3n = d_i[j] >> 3n = T[\alpha]$, then

$$d_i[j] = (c_i[j] + d_{i-1}) >> n \oplus T[\alpha],$$
$$\alpha = (c_i[j] + d_{i-1})(\mathrm{mod}\ 2^n),$$

therefore, the second-order n-bit and the third-order n-bit of $c_i[j]$ are arbitrary. Thus, the number of elements j such that $c_i[j] >> 3n = d_i[j] >> 3n = T[\alpha]$ is 2^{2n}.
If $b_i[j] >> 3n = c_i[j] >> 3n = d_i[j] >> 3n = T[\alpha]$, then

$$c_i[j] = (b_i[j] + c_{i-1}) >> n \oplus T[\alpha],$$
$$\alpha = (b_i[j] + c_{i-1})(\mathrm{mod}\ 2^n),$$

and the third-order n-bit of $c_i[j]$ is arbitrary, hence third-order n-bit of $b_i[j]$ is also arbitrary. It follows that the number of elements j such that $b_i[j]>>3n=c_i[j]>>3n=d_i[j]>>3n= T[\alpha]$ is 2^n.
It is obvious that if $a_i[j]>>3n=b_i[j]>>3n =c_i[j]>>3n=d_i[j]>>3n=T[\alpha]$, then there is the only $b_i[j]$.
The proposition is proved.

Proposition 3
The high-order n-bit of $d_i(k+2^{3n}),\ldots,d_i(k+r\cdot 2^{3n}),\ldots,d_i(k+(2^n-1)\cdot 2^{3n})$ is a permutation on $\{0,\ldots,2^n-1\}$.
Proof.
Note that
$a_i(k+r\cdot 2^{3n})=(k+r\cdot 2^{3n}+a_{i-1})>>n \oplus T[(k+r\cdot 2^{3n}+a_{i-1})(\text{mod } 2^n)]=(k+r\cdot 2^{3n}+a_{i-1})>>n \oplus T[(k+a_{i-1})(\text{mod } 2^n)]$.
Thus, $a_i(k+r\cdot 2^{3n}) (\text{mod } 2^{2n})= a_i(k+2^{3n}) (\text{mod } 2^{2n})$, $a_i(k+r\cdot 2^{3n})>>3n= a_i(k+2^{3n}) >>3n$, $r=0\ldots 2^n-1$, and the third-order n-bit of $a_i(k+2^{3n}),\ldots,a_i(k+r\cdot 2^{3n}),\ldots,a_i(k+(2^n-1)\cdot 2^{3n})$ is a permutation on $\{0,\ldots,2^n-1\}$.
By

$$b_i(k+r\cdot 2^{3n})=(a_i(k+r\cdot 2^{3n})+b_{i-1})>>n \oplus T[(a_i(k+r\cdot 2^{3n})+b_{i-1})(\text{mod } 2^n)],$$

it follows that $b_i(k+r\cdot 2^{3n}) (\text{mod } 2^n)= b_i(k+2^{3n}) (\text{mod } 2^n)$, $b_i(k+r\cdot 2^{3n})>>3n= b_i(k+2^{3n}) >>3n$, $r=0\ldots 2^n-1$.
Using

$$c_i(k+r\cdot 2^{3n})=(b_i(k+r\cdot 2^{3n})+c_{i-1})>>n \oplus T[(b_i(k+r\cdot 2^{3n})+c_{i-1})(\text{mod } 2^n)],$$

we get $c_i(k+r\cdot 2^{3n})>>3n= c_i(k+2^{3n}) >>3n$, $r=0\ldots 2^n-1$.

Let us remark that $(c_i(k+2^{3n})+d_{i-1})(\text{mod } 2^n),\ldots,(c_i(k+r\cdot 2^{3n})+d_{i-1})(\text{mod } 2^n),\ldots,(c_i(k+(2^n-1)\cdot 2^{3n})+d_{i-1})(\text{mod } 2^n)$ are all various, therefore the high-order n-bit of $d_i(k+2^{3n}),\ldots,d_i(k+r\cdot 2^{3n}),\ldots, d_i(k+(2^n-1)\cdot 2^{3n})$ are different.
The proposition is proved.

Proposition 4
The high-order n-bit of $c_i(k+2^{2n}),\ldots,c_i(k+r\cdot 2^{2n}),\ldots,c_i(k+(2^n-1)\cdot 2^{2n})$ is a permutation on $\{0,\ldots,2^n-1\}$.
Proof.
By

$$a_i(k+r\cdot 2^{2n})=(k+r\cdot 2^{2n}+a_{i-1})>>n \oplus T[(k+r\cdot 2^{2n}+a_{i-1})(\text{mod } 2^n)],$$
$$b_i(k+r\cdot 2^{2n})=(a_i(k+r\cdot 2^{2n})+b_{i-1})>>n \oplus T[(a_i(k+r\cdot 2^{2n})+b_{i-1})(\text{mod } 2^n)],$$
$$c_i(k+r\cdot 2^{2n})=(b_i(k+r\cdot 2^{2n})+c_{i-1})>>n \oplus T[(b_i(k+r\cdot 2^{2n})+c_{i-1})(\text{mod } 2^n)],$$

where $r\in Z_{2^n}$, it follows that

$$a_i(k+r\cdot 2^{2n}) (\text{mod } 2^n)= a_i(k+2^{2n}) (\text{mod } 2^n),$$
$$a_i(k+r\cdot 2^{2n})>>3n= a_i(k+2^{2n}) >>3n,$$
$$b_i(k+r\cdot 2^{2n})>>3n= b_i(k+2^{2n}) >>3n,$$

and $(b_i(k+2^{2n})+c_{i-1})(\text{mod } 2^n),\ldots,(b_i(k+r\cdot 2^{2n})+c_{i-1})(\text{mod } 2^n),\ldots,(b_i(k+(2^n-1)\cdot 2^{2n})+c_{i-1})(\text{mod } 2^n)$ are all various, therefore the high-order n-bit of $c_i(k+2^{2n}),\ldots,c_i(k+r\cdot 2^{2n}),\ldots, c_i(k+(2^n-1)\cdot 2^{2n})$ are different.
The proposition is proved.

Remark
The high-order n-bit of $c_i(k+2^{3n}),\ldots,c_i(k+r\cdot 2^{3n}),\ldots,c_i(k+(2^n-1)\cdot 2^{3n})$ are equal.
It follows by proposition 3.

4 Attacks on WAKE

In this section we describe two chosen plaintext attacks on WAKE stream cipher.

First let us carry out an estimation of the unicity distance D_{WAKE} of WAKE. Recall that the unicity distance is the number of keystream symbols that need to be observed in a known plaintext attack before the key can be uniquely determined. Note that the number of various states of WAKE is equal to $2^{16n} \cdot (2^{4n})^{2^n}$. Then we get that $2^{16n} \cdot (2^{4n})^{2^n} \approx (2^{4n})^{D_{WAKE}}$. Therefore, $D_{WAKE} \approx 4+2^n$.

Let $p_j(k)$ be a j^{th} word of a plaintext which is equal to k, i.e. $p_j(k)=k$. Let us denote with mark "*" elements of an output sequence $\{z_i^*\}$ produced on a guessed state and the guessed table Ò.
The method consists of three steps.

The first attack

Step 1

Choose k_1, k_2. (For example, $k_1=0$, $k_2=0$)

1. Encrypt $p_1(k_1)$ to obtain $y_1(k_1)$.
2. Determine $d_0= p_1(k_1) \oplus y_1(k_1)$.
3. Encrypt $p_2(0)$, $p_2(1),\ldots,p_2(t),\ldots,p_2(2^{4n}-1)$ to obtain $y_2(k_1, 0)$, $y_2(k_1, 1),\ldots.$, $y_2(k_1, t),\ldots$, $y_2(k_1, 2^{4n}-1)$.
4. Compute $d_1(k_1)=y_2(k_1, 0) \oplus p_2(0)$.
5. Encrypt $p_3(0)$ to obtain $y_3(k_1, 0, 0)$, $y_3(k_1,1, 0),\ldots.$, $y_3(k_1, j, 0),\ldots$, $y_3(k_1, 2^{4n}-1, 0)$.
6. Compute $d_2(k_1, j)= y_3(k_1, j, 0) \oplus p_3(0)$, for $j=1\ldots2^n$.

Step 2

1. Guess $c_1(k_1)$.
2. Compute $\alpha(k, c)= (d_0+ c_1(k_1)) \bmod 2^n$.
3. Compute $T[\alpha(k, c)]= d_1(k_1) \oplus (d_0+ c_1(k_1))>>n$.
4. Find j_t, $t=1\ldots2^{3n}$, such that $d_2(k_1, j_t)>>3n=T[\alpha(k, c)]$.
5. Use $d_2(k_1, j_t)=(c_2(k_1, j_t)+d_1(k_1))>>n \oplus T[\alpha(k, c)]$ to determine $c_2(k_1, j_t)$, where $t=1\ldots2^{3n}$.
6. Find r_t, $t=1\ldots2^{2n}$, such that $c_2(k_1, r_t)>>3n=T[\alpha(k, c)]$, $r_t \in \{ j_1,\ldots, j_{2^{3n}} \}$.
7. Use $c_2(k_1, r_t)=(b_2(k_1, r_t)+c_1(k_1))>>n \oplus T[\alpha(k, c)]$ to determine $b_2(k_1, r_t)$.
8. Find β_t, $t=1\ldots2^n$, such that $b_2(k_1, \beta_t)>>3n=T[\alpha(k, c)]$, $\beta_t \in \{ r_1,\ldots,r_{2^{2n}} \}$.
9. Guess $a_1(k_1)$.
10. Find β such that $\alpha(k, c)=(\beta+a_1(k_1)) \pmod{2^n}$.
11. Compute $a_2(k_1, \beta)=(a_1(k_1)+\beta)>>n \oplus T[\alpha(k, c)]$.

Step 3 (Restoration of T)

1. Use $a_3(k_1, \beta_t)=(\beta_t + a_2(k_1, k_2+ j_{t_k} \cdot 2^{2n}+ r_{t_k} \cdot 2^{3n}))>>n \oplus T[\beta 3_i + a_2(k_1, k_2+ j_{t_k} \cdot 2^{2n}+ r_{t_k} \cdot 2^{3n}) \pmod{2^n}]$, $i=1\ldots2^n$, to determine $T[0],\ldots,T[2^n-1]$.
2. Compute the first $L = D_{WAKE}$ of elements of the output sequence z_1^*, $z_2^*,\ldots$ z_L^*. If $z_1^* = z_1$, $z_2^* = z_2,\ldots z_L^* = z_L$, then we have found the correct initial state of the cryptosystem, otherwise return to step 2.

Let us estimate the complexity of this attack.

First note that in step 1 we compute 2^{4n} plaintexts and in step 2 the average of guessed elements is estimated $2^{4n} \cdot 2^{4n}=2^{8n}$. Therefore, the complexity of the method is estimated $T_m \approx 2^{4n}+2^{8n}$.

We stress that the complexity of the brute force attack is equal to $T_{br}=2^{16n} \cdot (2^{4n})^{2^n}$.
For n=8 used in practice the complexities are $T_m \approx 10^{19.2}$ and $T_{br} \approx 3 \cdot 10^{2504}$.

Now we describe another attack on WAKE. The complexity of this attack is smaller than the complexity of the previous attack but the number of used plaintexts is larger.
The method also consists of three steps.

The second attack

Step 1

Choose k_1, k_2. (For example, $k_1=0$, $k_2=0$)

1. Encrypt $p_1(k_1)$ to obtain $y_1(k_1)$.
2. Determine $d_0= p_1(k_1) \oplus y_1(k_1)$.
3. Encrypt $p_2(k_2)$, $p_2(k_2+2^{3n}),\ldots,p_2(k_2+r\cdot 2^{3n}),\ldots,p_2(k_2+(2^n-1)\cdot 2^{3n})$, $p_2(k_2+2^{2n})$, $p_2(k_2+2^{2n}+2^{3n}),\ldots$, $p_2(k_2+2^{2n}+r\cdot 2^{3n}),\ldots,p_2(k_2+2^{2n}+(2^n-1)\cdot 2^{3n}),\ldots,p_2(k_2+j\cdot 2^{2n}+2^{3n}),\ldots,p_2(k_2+j\cdot 2^{2n}+r\cdot 2^{3n}),\ldots,p_2(k_2+j\cdot 2^{2n}+(2^n-1)\cdot 2^{3n}),\ldots,p_2(k_2+(2^n-1)\cdot 2^{2n}+2^{3n}),\ldots,p_2(k_2+(2^n-1)\cdot 2^{2n}+r\cdot 2^{3n}),\ldots,p_2(k_2+(2^n-1)\cdot 2^{2n}+(2^n-1)\cdot 2^{3n})$ to obtain $y_2(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n})$ for all $j, r \in Z_{2^n}$. (We have encrypted 2^{2n} of words.)
4. Compute $d_1(k_1)=y_2(k_1, k_2) \oplus p_2(k_2)$.
5. Encrypt $p_3(0)$, $p_3(1),\ldots,p_3(t),\ldots,p_3(2^{4n}-1)$ to obtain $y_3(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, 0)$, $y_3(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, 1),\ldots.$, $y_3(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, t),\ldots$, $y_3(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, 2^{4n}-1)$ for all $j, r \in Z_{2^n}$.
6. Compute $d_2(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n})= y_3(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, 0) \oplus p_3(0)$ for all $j, r \in Z_{2^n}$.
7. Encrypt $p_4(0)$ to obtain $y_4(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, 0, 0)$, $y_4(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n},1, 0),\ldots.$, $y_4(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, j, 0),\ldots$, $y_4(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, 2^{4n}-1, 0)$ for all $j, r \in Z_{2^n}$.
8. Compute $d_3(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, t)= y_4(k_1, k_2+j\cdot 2^{2n}+r\cdot 2^{3n}, t, 0) \oplus p_4(0)$, for all $j, r, t \in Z_{2^n}$.

Step 2

1. Guess $c_1(k_1)$.
2. Compute $\alpha(k, c)= (d_0+ c_1(k_1)) \bmod 2^n$.
3. Compute $T[\alpha(k, c)]= d_1(k_1) \oplus (d_0+ c_1(k_1))>>n$.
4. Find j_t, r_t, $t=1\ldots 2^n$, such that $d_2(k_1, k_2+j_t\cdot 2^{2n}+r_t\cdot 2^{3n})>>3n=T[\alpha(k, c)]$.
5. Use $d_2(k_1, k_2+j_t\cdot 2^{2n}+r_t\cdot 2^{3n})=(c_2(k_1, k_2+j_t\cdot 2^{2n}+r_t\cdot 2^{3n})+d_1(k_1))>>n \oplus T[\alpha(k, c)]$ to determine $c_2(k_1, k_2+j_t\cdot 2^{2n}+r_t\cdot 2^{3n})$, where $t=1\ldots 2^n$.
6. Find j_{t_k}, r_{t_k} such that $c_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n})>>3n=T[\alpha(k, c)]$, $j_{t_k} \in \{ j_1,\ldots, j_{2^n} \}$, $r_{t_k} \in \{ r_1,\ldots, r_{2^n} \}$. (By proposition 3 and proposition 4 j_{t_k}, r_{t_k} exist.)
7. Use $c_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n})=(b_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n})+c_1(k_1))>>n \oplus T[\alpha(k, c)]$ to determine $b_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n})$.
8. Find $\beta 1_s$, $s=1\ldots 2^{3n}$ such that $d_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n}, \beta 1_s)>>3n=T[\alpha(k, c)]$.

9. Use $d_3(k_1, k_2+j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 1_s)=(c_3(k_1, k_2+j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 1_s)+d_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n}))>>n \oplus T[\alpha(k, c)]$ to determine $c_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 1_s)$, where $s=1\ldots 2^{3n}$.
10. Find $\beta 2_i$, $i=1\ldots 2^{2n}$, such that $c_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 2_i)>>3n=T[\alpha(k, c)]$, $\beta 2_i \in \{ \beta 1_1,\ldots, \beta 1_{2^{3n}} \}$.
11. Use $c_3(k_1, k_2+j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n},\beta 2_i)=(b_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 2_i)+ c_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n}))>>n \oplus T[\alpha(k, c)]$ to determine $b_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 2_i)$, where $i=1\ldots 2^{2n}$.
12. Find $\beta 3_i$, $i=1\ldots 2^n$, such that $b_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 3_i)>>3n=T[\alpha(k, c)]$, $\beta 3_i \in \{ \beta 2_1,\ldots, \beta 2_{2^{2n}} \}$.
13. Use $b_3(k_1, k_2+j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n},\beta 3_i)=(a_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 3_i)+ b_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n}))>>n \oplus T[\alpha(k, c)]$ to determine $a_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 3_i)$, where $i=1\ldots 2^n$.
14. Find $\beta 4$ such that $a_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n}, \beta 4)>>3n=T[\alpha(k, c)]$, $\beta 4 \in \{ \beta 3_1,\ldots, \beta 3_{2^n} \}$.
15. Use $a_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n},\beta 4)=(\beta 4+ a_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n}))>>n \oplus T[\alpha(k, c)]$ to determine $a_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n})$.

Step 3 (Restoration of T)

1. Use $a_3(k_1, k_2+ j_{t_k}\cdot 2^{2n}+r_{t_k}\cdot 2^{3n},\beta 3_i)=(\beta 3_i + a_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n}))>>n \oplus T[\beta 3_i + a_2(k_1, k_2+ j_{t_k}\cdot 2^{2n}+ r_{t_k}\cdot 2^{3n}) \pmod{2^n}]$, $i=1\ldots 2^n$, to determine $T[0],\ldots,T[2^n-1]$.
2. Compute the first $L = D_{WAKE}$ of elements of the output sequence $z_1^*, z_2^*,\ldots z_L^*$. If $z_1^* = z_1$, $z_2^* = z_2,\ldots z_L^* = z_L$, then we have found the correct initial state of the cryptosystem, otherwise return to step 2.

Let us estimate the complexity of the method. Note that in step 1 we compute 2^{6n} plaintexts and in step 2 the average of guessed elements is estimated 2^{4n}. Therefore, the complexity of the method is estimated $T_m \approx 2^{4n}+2^{6n}$. For n=8 used in WAKE the complexities are $T_m \approx 10^{14.4}$ and $T_{br} \approx 3\cdot 10^{2504}$.

5 Conclusion

We have presented two chosen plaintext attacks on the WAKE stream cipher. The complexity of fist attack is $10^{19.2}$ and the second attack is $10^{14.4}$ but the complexity of the brute force attack is $3\cdot 10^{2504}$.

Our results are intrinsic to the design principles of WAKE and are independent of the key scheduling. We believe improvements to these attack are possible. Although the attacks are by far not practical, it gives new intrinsic insight into the algorithm.

References

[1] D.J. Wheeler, "A Bulk Data Encryption Algorithm", Fast Software Encryption (Ed. R. Anderson), LNCS, No. 809, Springer-Verlag, 1994, pp. 127-134.

[2] Clapp C., " Optimizing a Fast Stream Cipher for VLIW, SIMD, and Superscalar Processors", Fast Software Encryption (Ed. S. Vaudenay), LNCS 1372, Springer-Verlag, 1998.
[3] Pudovkina M. "Cryptanalysis of the Vesta-2M Stream Cipher ", presented at the Rump Session, Eurocrypt'2001, Innsbruck, Tyrol, Austria, May 6-10, 2001
[4] Varfolomeev A.A., Zhukov A.E., Pudovkina M., "Analysis of Stream Ciphers ", Moscow, 2000.
[5] Schneier B. Applied Cryptography, John Wiley&Sons,1996.
[6] Varfolomeev, A.A., Pudovkina M. "A Cycle Structure of the Solitaire Keystream Generator". 3nd International Workshop on Computer Science and Information Technologies CSIT'2001, YFA, 2001
[7] Pudovkina M. "A Cycle Structure of the Alleged RC4 Keystream Generator". Journal of "Security of information technologies", Moscow, 4, 2000.