Differential Distinguishing Attack of Shannon Stream Cipher

Yaser Esmaeili Salehani, Elham Shakour, S. Mehdi M. Hassanzadeh
Zaeim Electronic Ind., R&D Department
Bergen University, Informatic Department, Selmer Center
Email: yesmaeili@zaeim.co.ir

ABSTRACT

In this paper, some weak points in design of Shannon algorithm are probed and by exploiting them, a differential distinguisher is proposed for this algorithm. It is proved that Shannon is vulnerable against differential attack with complexity less than exhaustive search. We show that a distinguishing attack is examined that can recognize the output key stream (running keys) from a truly random stream with computational complexity about $O(2^{8.92})$ with error probability equal to 0.001 by $2^{8.92}$ random differential initial states.

Keywords: Shannon stream cipher, differential, distinguisher.

INTRODUCTION

The Shannon stream cipher was proposed by Philip Hawkes et al. as a software-efficient algorithm, up to 256 bits key length [HMPRV07]. Shannon is an entirely new design, influenced by members of the SOBER family of stream ciphers [HR99]. It consists of a single 32-bit wide, 16-element nonlinear feedback shift register and an extra word, which is supplemented for message authentication with 32 parallel CRC-16 registers.

In addition to keystream generation, Shannon also offers message authentication functionality that is directly incorporated into its structure. In our analysis of Shannon, we consider only the keystream generator part, however.

Although no serious weaknesses have been reported so far but we demonstrate weaknesses which lead to differential weaknesses on initial state to key stream outputs [BD06] and a distinguishing attack as well. The paper is organized as follows: In section 2, a brief description of Shannon algorithm is presented. The detail of our attack is presented in section 3. Finally, we summarize our results, conclude this paper and propose future works.
DESCRIPTION OF SHANNON

Shannon is a synchronous stream cipher designed by Hawkes, McDonald, Paddon, Rose, and de Vries [HMPRV07] of Qualcomm Australia. The keystream generator of Shannon produces a keystream of 32-bit words based on a 256-bit secret key. It is based on a single NLFSR and an NLF.

The vector of words \( \sigma_t = (r_t[0], \ldots, r_t[15]) \) is known as the state of the register at time \( t \), and the state \( \sigma_0 = (r_0[0], \ldots, r_0[15]) \) is called the initial state. The key state is initialized from the secret key by the key loading. The key state can be used directly as the initial state, or can be further perturbed by the nonce loading process to form the initial state. An additional state word called Konst (for historical reasons) is derived from the key loading or nonce loading, and is used in the nonlinear feedback [HMPRV07].

The MAC accumulation in Shannon is a combination of nonlinear accumulation in the main shift register (as in Helix) and linear accumulation in 32 parallel CRC registers, which we implement as a word-wide 16-word register [HMPRV07].

![Figure 1: Description of Shannon](image-url)
The state transition function transforms state $\sigma_i$ into state $\sigma_{i+1}$, and derives an output keystream word $v_i$, in the following manner:

1. $r_{i+1}[i] \leftarrow r_i[i+1]$, for $i = 1...14$.
2. $r_{i+1}[15] \leftarrow f_1(r_i[12] \oplus r_i[13] \oplus Konst) \oplus (r_i[0] << 1)$.
3. Let $temp \leftarrow f_2(r_{i+1}[2] \oplus r_{i+1}[15])$.
4. $r_{i+1}[0] \leftarrow r_i[1] \oplus temp$ (that is, “feed forward” to the new lowest element).
5. $v_i \leftarrow temp \oplus r_{i+1}[8] \oplus r_{i+1}[12]$.

The nonlinear functions $f_1$ and $f_2$ are defined in below:

There are two related nonlinear functions used in the state updating (and output) function of Shannon, mapping an input word to an output word. Both functions have the same form given an input word $w$:

1. $t \leftarrow w \oplus ((w << A) | (w << B))$
2. Return $t \leftarrow t \oplus ((t << C) | (t << D))$

The only difference between the functions is the ordering of the constants $\{A, B, C, D\}$. For $f_1$ they are $\{5, 7, 19, 22\}$ respectively, while for $f_2$ they are $\{7, 22, 5, 19\}$ respectively.

For further details of Shannon, such as the message authentication we refer to the specification [HMPRV07].

**DIFFERENTIAL WEAKNESSES OF SHANNON**

A differential of a stream cipher is a prediction that a given input difference (be it the key, the IV, or the internal state) produce some output difference (be it the key stream or the internal state) [BD06]. As in block ciphers, we are not interested in what exactly happens in the cipher when the differential is satisfied, but only with the probability of the differential. We note that just like in block ciphers, it is hard to compute the exact probability of a given differential, but we can of course use the probability of a given
characteristic with the same input and output differences as a lower bound [BD06].

Now, we present differential weaknesses on Shannon that is based on some serious weaknesses on this algorithm. First, we can say that the Shannon algorithm consists of two parts. They are composed of key loading that produces the initial state by secret key and nonce and key generation that produces the output key stream by initial state. Key loading part and key generation part can be considered separately.

Next, we have shown that chosen difference on the initial state represents a differential distinguisher on the key stream output. This weakness comes from nonlinear function $f_2$ used in nonlinear part of the algorithm. Also, it causes to our scenario attack.

Moreover, it can be found the differential weakness of $f_1$ function that can be used for another differential weakness for future works.

To introduce the attack, we first define two notations $\delta_{i_1,i_2,...,i_n}$, $\Delta_{i_1,i_2,...,i_n}$; $i_1,i_2,...,i_n \in \{0,1,...,31\}$, which indicate a 32-bit known word with 1’s in positions $i_1,i_2,...,i_n$ and 0’s in other positions.

To introduce the attack, we first define two notations $\delta_{i_1,i_2,...,i_n}$, $\Delta_{i_1,i_2,...,i_n}$; $i_1,i_2,...,i_n \in \{0,1,...,31\}$, which indicate a 32-bit known word with 1’s in positions $i_1,i_2,...,i_n$ and 0’s in other positions.

To turn this remark into an attack, it is necessary to consider the following scenario:

**Attack Scenario:**

1. Adversary produces 512 bits random initial state named IS.
2. Adversary passes IS and IS' to the black box. IS' differs only at 11th word from IS (i.e. IS[i]=IS'[i] for any i=0,1,...,15 but 11, IS[11] = IS[11]⊕δ).
3. The black box can choose randomly between a random oracle or Shannon algorithm. If the Shannon is chosen, its register is updated by IS and IS’ respectively.
4. Output values ($v_t$ with IS and $v'_t$ with IS’) are computed at $n$ clocks.
5. This scenario is repeated N times and Adversary calculates the output differences ($v_t \oplus v'_t = \Delta_t$) in each repetition.
Differential Distinguishing Attack Of Shannon Stream Cipher

6. Adversary distinguishes the output of Shannon from random oracle by using of distinguisher that presented in this paper and the output differences (v_i⊕v_i=Δ_i).

Before describing the differential weaknesses on Shannon, we first exhibit differential weaknesses of \( f_2 \) function.

\( f_2 \) Differential Properties

For given value \( \delta \), we can trace the differential output of \( f_2 \) (a=1,2). Below we describe this differential trails for \( f_2 \) function. Differential properties of \( f_i \) function while the input difference is \( \delta \) can be calculated in the same way, too.

Let \( \delta_i \) is a 32-bit differential value which differs only \( i^{th} \) bit (i.e. \( \delta_i = 1 \) for \( i \) and 0 for the other numbers).

We suppose that 31\(^{st} \) bit of input is activated. Then the output differential of \( f_2 \) function is determined bit by bit. We conclude that nine bits of output from \( f_2 \) function will be impressed by \( \delta_i \). Eight bits of them are determined with some probabilities as one bit of them is set with probability 1. Table 1 depicts these bits differentials and their probabilities for \( \delta_{31} \). For the other \( i \) of \( \delta_i \), \( i \in \{1,...,30\} \), we can give trails which just differ from bit position of output but similar to calculated probabilities for \( \delta_{31} \).

<table>
<thead>
<tr>
<th>( W )</th>
<th>( W' )</th>
<th>( \langle\langle 7 \rangle\rangle )</th>
<th>( \langle\langle 7 \rangle\rangle )</th>
<th>( \langle\langle 22 \rangle\rangle )</th>
<th>( \langle\langle 22 \rangle\rangle )</th>
<th>( T )</th>
<th>( \langle\langle 5 \rangle\rangle )</th>
<th>( \langle\langle 5 \rangle\rangle )</th>
<th>( \langle\langle 19 \rangle\rangle )</th>
<th>( \langle\langle 19 \rangle\rangle )</th>
<th>( M )</th>
<th>( M' )</th>
<th>( \Delta M )</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>( W_{11} )</td>
<td>( W'_{11} )</td>
<td>( W_{24} )</td>
<td>( W'_{24} )</td>
<td>( W_{5} )</td>
<td>( W'_{5} )</td>
<td>( T_{11} )</td>
<td>( T'_{11} )</td>
<td>( T_{12} )</td>
<td>( T'_{12} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 1 )</td>
<td>Always</td>
<td></td>
</tr>
<tr>
<td>( W_{23} )</td>
<td>( W'_{23} )</td>
<td>( W_{21} )</td>
<td>( W'_{21} )</td>
<td>( W_{8} )</td>
<td>( W'_{8} )</td>
<td>( T_{21} )</td>
<td>( T'_{21} )</td>
<td>( T_{22} )</td>
<td>( T'_{22} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>Always</td>
<td></td>
</tr>
<tr>
<td>( W_{21} )</td>
<td>( W'_{21} )</td>
<td>( W_{22} )</td>
<td>( W'_{22} )</td>
<td>( W_{7} )</td>
<td>( W'_{7} )</td>
<td>( T_{23} )</td>
<td>( T'_{23} )</td>
<td>( T_{24} )</td>
<td>( T'_{24} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>Always</td>
<td></td>
</tr>
<tr>
<td>( W_{24} )</td>
<td>( W'_{24} )</td>
<td>( W_{21} )</td>
<td>( W'_{21} )</td>
<td>( W_{6} )</td>
<td>( W'_{6} )</td>
<td>( T_{25} )</td>
<td>( T'_{25} )</td>
<td>( T_{26} )</td>
<td>( T'_{26} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>Always</td>
<td></td>
</tr>
<tr>
<td>( W_{22} )</td>
<td>( W'_{22} )</td>
<td>( W_{30} )</td>
<td>( W'_{30} )</td>
<td>( W_{5} )</td>
<td>( W'_{5} )</td>
<td>( T_{27} )</td>
<td>( T'_{27} )</td>
<td>( T_{28} )</td>
<td>( T'_{28} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>Always</td>
<td></td>
</tr>
<tr>
<td>( W_{25} )</td>
<td>( W'_{25} )</td>
<td>( W_{30} )</td>
<td>( W'_{30} )</td>
<td>( W_{4} )</td>
<td>( W'_{4} )</td>
<td>( T_{30} )</td>
<td>( T'_{30} )</td>
<td>( T_{31} )</td>
<td>( T'_{31} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>( 3/4 )</td>
<td></td>
</tr>
<tr>
<td>( W_{21} )</td>
<td>( W'_{21} )</td>
<td>( W_{18} )</td>
<td>( W'_{18} )</td>
<td>( W_{3} )</td>
<td>( W'_{3} )</td>
<td>( T_{32} )</td>
<td>( T'_{32} )</td>
<td>( T_{34} )</td>
<td>( T'_{34} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>( 3/4 )</td>
<td></td>
</tr>
<tr>
<td>( W_{24} )</td>
<td>( W'_{24} )</td>
<td>( W_{11} )</td>
<td>( W'_{11} )</td>
<td>( W_{2} )</td>
<td>( W'_{2} )</td>
<td>( T_{40} )</td>
<td>( T'_{40} )</td>
<td>( T_{41} )</td>
<td>( T'_{41} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>Always</td>
<td></td>
</tr>
<tr>
<td>( W_{22} )</td>
<td>( W'_{22} )</td>
<td>( W_{15} )</td>
<td>( W'_{15} )</td>
<td>( W_{6} )</td>
<td>( W'_{6} )</td>
<td>( T_{50} )</td>
<td>( T'_{50} )</td>
<td>( T_{51} )</td>
<td>( T'_{51} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>Always</td>
<td></td>
</tr>
<tr>
<td>( W_{23} )</td>
<td>( W'_{23} )</td>
<td>( W_{14} )</td>
<td>( W'_{14} )</td>
<td>( W_{7} )</td>
<td>( W'_{7} )</td>
<td>( T_{33} )</td>
<td>( T'_{33} )</td>
<td>( T_{34} )</td>
<td>( T'_{34} )</td>
<td>( M_{9} )</td>
<td>( M'_{9} )</td>
<td>( 0 )</td>
<td>Always</td>
<td></td>
</tr>
</tbody>
</table>

International Journal of Cryptology Research
Therefore we can say that the differential output bits of $f_2$ function respectively. 
\[X_j \text{ bit (X=W',T',M')} \]  
Probabilistically equal to $Y_j$ bit (Y=W,T,M). Two last columns of Table I depict the bit differentials output ($\Delta M$) and their probabilities respectively. Theoretically, Shannon stream cipher is a random number generator (RNG) for generation of key stream.

So, we can suppose that the output bits of $f_2$ function are independent. Therefore we can say that the differential output bits of $f_2$ function are 32-bit word $\Delta M$ (i.e. 0x80000000 from Table I) with the probability of 
\[ \prod_{i=0}^{31} p_i = 1 \times \left( \frac{3}{4} \right)^4 \left( \frac{1}{2} \right)^4 \approx 2^{-5.66}. \]
We achieved a little more than this probability by implementation.
Our Differential Distinguishing Attack

In this section a differential distinguishing attack is presented by using the weakness of \( f_2 \) function. According to the previous section, if the initial differential is \( \delta_{11} \) of 11\(^{th}\) word, the output differential of Shannon is equal to 0x80000000 with probability of \( 2^{-5.66} \) after 9 times clock.

For applying our attack by using of this bias, we take \( N \) differential outputs of black box by repeating the scenario \( N \) times. Each time, after 9 clocks, we have a 32-bit differential output word. Consequently, a sequence consisting of \( N \) 32-bit differential words is constructed. We refer to this sequence by \( O_9 \).

We have two hypotheses for \( O_9 \): \( H_0: O_9 \) is the output of Shannon. \( H_1: O_9 \) is the output of true random generator. The probability of \( H_0 \) and \( H_1 \) is equal to:

\[
H_0 = \begin{cases} 
\Pr\{O_{9,i} = 0x80000000\} = 2^{-5.66} \\
\Pr\{O_{9,i} \neq 0x80000000\} = 1 - 2^{-5.66}
\end{cases}
\]

\[
H_1 = \begin{cases} 
\Pr\{O_{9,i} = 0x80000000\} = 2^{-32} \\
\Pr\{O_{9,i} \neq 0x80000000\} = 1 - 2^{-32}
\end{cases}
\]

Let \( O_{9,i} \) denote the \( i^{th} \) word of sequence \( O_9 \). By using of frequency test, we can distinguish this output with error probability 0.001. We need \( N=2^{8.92} \) words in sequence \( O_9 \). So, we need to run Shannon stream cipher \( N=2^{8.92} \) times. Then, the computational complexity is equal to \( O(2^{8.92}) \).

Table 2 depicts these differences and probabilities. We choose two initial vectors which their difference is only in 31\(^{st}\) bit (\( \delta_{31} \)) of 11\(^{th}\) word. According to results of Table 3, differences of key stream with adequate probability are obtained.
TABLE 2: Differences of initial state and key stream

<table>
<thead>
<tr>
<th>t</th>
<th>Différence values ((r_t \oplus r'_t))</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Always</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Always</td>
</tr>
<tr>
<td>2</td>
<td>(\delta_{31})</td>
<td>Always</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>Always</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>Always</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>Always</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>Always</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>Always</td>
</tr>
<tr>
<td>8</td>
<td>0x80000000</td>
<td>(P = 2^{-3.56})</td>
</tr>
</tbody>
</table>

Also we can apply this style of differential distinguishing while the position of input differential is changed. It means that we can cycle the register \(n\) times with two initial states \(r_0\) and \(r'_0\) (they consist of 16 words which differ only \(j^{th}\) word, \(j=10,9,\ldots,3\) and \(n=j-1\)) until the probability of word-oriented key stream output difference is greater than \(2^{-32}\).

**CONCLUSIONS AND FUTURE WORK**

In this paper, we proposed differential distinguishing attack on Shannon algorithm. Differential distinguishers are described base on the weaknesses of \(f_2\) function. Our proposed distinguishing attack can recognize the output cipher from a truly random sequence by the computational complexity is equal to \(O(2^{8.92})\) and error probability equal to 0.001.

Also, there is a lot of interesting research directions which can be exploited further, and we mention one of them. We can distinguish differential initial state which come from two alike random keys but differs just a known position key words as the same as our proposed scenario.
REFERENCE

