Abstract

Nowadays there are different types of attacks in block and stream ciphers. In this work we will present some of the most used attacks on stream ciphers. We will present the newest techniques with an example of usage in a cipher, explain and comment. Previous we will

1 INTRODUCTION

Over the years, many ciphers was developed. There are block and stream ciphers for all kind of applications. However, we need to guarantee the security of that ciphers. Then, we developed different attacks to test the resistance of our ciphers.

In this work, we will present the constitution of block and stream ciphers. We will show the difference between them.

We will discuss about the most importants attacks for stream ciphers. We will present the most importante works in the area, explain the attack and give an example of application. We selected nine attacks, but there are many others. The attacks that we select are: Exhaustive Search, Algebraic, Correlation, Fault, Distinguishing, Chosen-IV, Slide, Cube, Time-Memory Trade-off and Guess and Determine. To made this selection, we choose historical importance, efficiency of the attack and newest attacks.

1.1 Papers Organization

In the Section 2 we will discuss about block and stream ciphers, we will give the difference between them and examples of ciphers. After the concept of ciphers, in the Section 3 we will discuss about the attacks in stream ciphers. We will explain how the attack works, the most relevants work in the area and example of the application. In the Section 4 we will give a brief discussion about all the work and the importance of the attacks.

2 TYPES OF CIPHERS

In terms of ciphers, there are two types of ciphers: Block ciphers and Stream Ciphers. In this section we will present a concept about this two ciphers.
2.1 Block Ciphers

A concept of block ciphers was determined by Menezes et al [61]:

“A block cipher is a function which maps \( n \)-bit plaintext blocks to \( n \)-bit ciphertext blocks; \( n \) is called the blocklength. It may be viewed as a simple substitution cipher with large character size. The function is parameterized by a \( k \)-bit key \( K \), taking values from a subset \( K \) (the key space) of the set of all \( k \)-bit vectors \( V_k \). It is generally assumed that the key is chosen at random. Use of plaintext and ciphertext blocks of equal size avoids data expansion.” [61]

In mathematical terms, we can define block ciphers like as:

**Definition 1.** An \( n \)-bit block cipher is a function \( E : V_n \times K \rightarrow V_n \), such that for each key \( K \in K \), \( E(P,K) \) is an invertible mapping (the encryption function for \( K \)) from \( V_n \) to \( V_n \), written \( E_K(P) \). The inverse mapping is the decryption function, denoted \( D_K(C) \). \( C = E_K(P) \) denotes that ciphertext \( C \) results from encrypting plaintext \( P \) under \( K \). [61]

2.1.1 Operation Modes of Block Ciphers

Talking about block ciphers, we have four most common modes of operation: ECB(Electric codebook), CBC(Chosen-block Chaining), CFB(Cipher feedback) and OFB(Output feedback). We will do an explanation about this four modes of operation.

**ECB Mode:**

This mode produces identical ciphertext, because the blocks are enciphered independently of other blocks. In the algorithm of ECB mode from Menezes et al [61], we can verify this property of this mode. To help to understanding we have the Figure 1 and the Algorithm 1.

![Figure 1: ECB Operation Mode of block ciphers](image)

**Algorithm 1:** Algorithm of ECB mode.

**Data:** \( k \)-bit key \( K \); \( n \)-bit plaintext blocks \( x_1, \ldots, x_n \)

**Result:** produce ciphertext blocks \( c_1, \ldots, c_n \); decrypt to recover plaintext.

1. Encryption: for \( 1 \leq j \leq n \), \( c_j \leftarrow E_K(x_j) \).
2. Decryption: for \( 1 \leq j \leq n \), \( x_j \leftarrow E_K^{-1}(c_j) \).

**CBC Mode:**

In this operation we have a dependency in each block, because every ciphered block has a xor operation with the previous block. In the Figure 2 we can understand better this idea. The problem of this mode is the error propagation, if a single
bit error in ciphertext block \( c_j \), then affects all the other blocks after \( c_j \). In the Algorithm 2 we can see this dependency.

**Algorithm 2:** Algorithm of CBC mode.

**Data:** \( k \)-bit key \( K \); \( n \)-bit \( IV \); \( n \)-bit plaintext blocks \( x_1, \ldots, x_n \)

**Result:** produce ciphertext blocks \( c_1, \ldots, c_n \); decrypt to recover plaintext.

1. Encryption:
   \( c_0 \leftarrow IV \)
   for \( 1 \leq j \leq n, c_j \leftarrow E_K(c_{j-1} \oplus x_j) \).
2. Decryption:
   \( c_0 \leftarrow IV \)
   for \( 1 \leq j \leq n, x_j \leftarrow c_{j-1} \oplus E_K^{-1}(c_j) \).

**CFB Mode:**

In this operation the plain text is ciphered in \( r \)-bits plaintext units. This operation is needed, because some applications need \( r \)-bits ciphered and transmitted without delay. This \( r \) is fixed, \( r < n \) (often \( r = 1 \) or \( r = 8 \)). In the Algorithm 3 we can understand better this operation.

**Algorithm 3:** Algorithm of CFB mode.

**Data:** \( k \)-bit key \( K \); \( n \)-bit \( IV \); \( r \)-bit plaintext blocks \( x_1, \ldots, x_n \) (\( 1 < r < n \))

**Result:** produce \( r \)-bit ciphertext blocks \( c_1, \ldots, c_n \); decrypt to recover plaintext.

1. Encryption: \( I_1 \leftarrow IV \). (\( I_j \) is the input value in a shift register.)
   For \( i \leq j \leq n \):
   \( O_j \leftarrow E_K(I_j) \). (Compute the block cipher output)
   \( t_j \leftarrow \) the \( r \) leftmost bits of \( O_j \).
   \( c_j \leftarrow x_j \oplus t_j \).
   \( I_{j+1} \leftarrow 2^r \cdot I_j + c_j \bmod 2^n \).
2. Decryption: \( I_1 \leftarrow IV \) for \( 1 \leq j \leq n \), upon receiving \( c_j \):
   \( x_j \leftarrow c_j \oplus t_j \), where \( t_j, O_j \) and \( I_j \) are computed as above.

In the Figure 3 we can understanding better the Algorithm 3.

**OFB Mode:**

In the last most common operation, we have a mode of operation that is used for applications in which all error propagation must be avoided. This operation is similar to CFB, the difference is that the output of the encryption block function \( E \) serves as the feedback. Exists two versions of OFB, we will present the version ISO 10116 (Algorithm 4), but exists the FIPS version.
Algorithm 4: Algorithm of OFB (ISO 10116) mode.

**Data:** \( k \)-bit key \( K \); \( n \)-bit IV; \( r \)-bit plaintext blocks \( x_1, \ldots, x_n \) (\( 1 < r < n \))

**Result:** produce \( r \)-bit ciphertext blocks \( c_1, \ldots, c_n \); decrypt to recover plaintext.

1. **Encryption:** \( I_1 \leftarrow IV \) (If \( I_j \) is the input value in a shift register.)
   For \( i \leq j \leq n \), given plaintext block \( x_j \):
   - \( O_j \leftarrow E_K (I_j) \). (Compute the block cipher output)
   - \( t_j \leftarrow \text{the } r \text{ leftmost bits of } O_j \).
   - \( c_j \leftarrow x_j \oplus t_j \).
   - \( I_{j+1} \leftarrow O_j \).

2. **Decryption:** \( I_1 \leftarrow IV \) for \( 1 \leq j \leq n \), upon receiving \( c_j \):
   \( x_j \leftarrow c_j \oplus t_j \), where \( t_j \), \( O_j \) and \( I_j \) are computed as above.

In the Figure 4 we can understand better the Algorithm 4.

We present the most common operations mode of the block cipher, exists others operations like: Propagating cipher-block chaining (PCBC) and Counter (CTR).

[61]

2.1.2 Examples of Block Ciphers

Nowadays in terms of ciphers exists a lot of block ciphers, in this subsection we will present the block ciphers used by National Institute of Standards and Technology (NIST). According to NIST, they use this block ciphers: AES [37] [38], Triple DES [39], and Skipjack [57] [65].

However, exist other important block ciphers like: Blowfish [71], LED Block Cipher [50] and others [47].
2.2 Stream Ciphers

In the subsection 2.1 we talked about block ciphers, now we will talk about another class of ciphers the stream ciphers. The principal difference between these two types of ciphers is in block ciphers we cipher a block of data per time and in stream ciphers we cipher a stream of data.

"Block ciphers tend to simultaneously encrypt groups of characters of a plaintext message using a fixed encryption transformation." [61]

In the block ciphers we had the operation modes, in stream ciphers we have something like this. We have stream ciphers based on linear feedback shift registers (LFSRs) and stream ciphers that does not use LFSRs. In this work we will discuss attacks on stream ciphers based on LFSRs, because the attacks consists at the LFSRs.

2.2.1 Examples of Stream Ciphers

There exists a lot of streams cipher, we will present stream ciphers used by European Network of Excellence in Cryptology (ECRYPT). The EUROCRYPT has a project called eSTREAM, this project is promoting the design of efficient and compact stream ciphers suitable for widespread adoption [46] [49].

The stream ciphers recommended by ECRYPT are: HC-128 [77], Grain v1 [52], Rabbit [20], MICKEY 2.0 [9], Salsa20/12 [13], Trivium [27] and SOSEMANUK [12].

3 ATTACKS’ TYPES

In this section we will present the attacks’ types, we also explain and discuss what is the use of the attack.

It is important to understand that all the attacks have one purpose. The purpose is to discover the key used in the process of ciphering and deciphering. Each attack has a method to try to discover the key that was used, we will give some examples of the application of the attack.

3.1 Exhaustive Search Attack

The exhaustive search attack is also called brute force. The method of this attack is to search through all possible states, checking for a match between the resulting and the observed keystream.

Fortunately, Babbage [8] in 1995 improved the exhaustive search attack in stream ciphers. He defined two attacks in this area.

In the first attack, the attacker first produces a list of \( n \)-bit subsequences, sorted in lexicographic (or numeric) order. Then the attacker selects a random candidate state in this list and checks if the selected state produces the output of cipher, then the attacker found the initial state else he continues to try to find the initial state [8].

The second attack was defined by Babbage [8] as:

“Let \( V \) be a vector space of dimension \( n \) over \( GF(2) \), with each possible KG(Keystream Generator) state an element of \( V \). The initial state, which we wish to determine, is \( s_0 \), and the state transition function is linear, and so can be represented by an \( n \times n \) matrix \( A \), so that \( s_i = s_0 A^i \).

The output function \( h : V \rightarrow GF(2) \), so that the \( i \)th keystream bit \( k_i \) is equal to \( h(s_i) \).” [8]
3.2 Algebraic Attack

The algebraic attack is used in stream ciphers based on LFSRs. This attack tries to find the initial state given some keystream bits.

The algebraic attacks have two steps. In the first step, the attack tries to find a system of equations in the bits of the secret key $K$ and the output bits $Z_t$ [6]. If it has enough low degree equations and known key bits stream, then the secret key $K$ can be recovered by solving this system of equations in a second step. This system could be solved using Groebner bases [19] [26], XL, XSL and others [70] [33].

For Courtois [35] the algebraic attack can be defined in a synchronous stream cipher, which has a state $s \in GF(2)^n$. At each clock $t$ the state $s$ is updated by a “connection function” $s \rightarrow L(s)$ that is assumed to be linear over $GF(2)$. Then a combine $f$ is applied to $s$, to produce the output bit $b_t = f(s)$. The goal for the attack is to find the initial state of $s$ [35] [34].

Flori et al [48] approach how to avoid the algebraic attacks using a good binary strings distribution. Unfortunately, they just had a conjecture and do not have a theorem. However, Wang and Johansson [75] proved that is capable to have a Boolean function [28] [24] [30] with fast algebraic immunity and higher order nonlinearity. To determine the computation of immunity against algebraic and fast algebraic attack you can consult the Armknecht et al work [7].

Using this attack Orumiehchiha et al [67] recovered both initial state and secret key, from WG-7 cipher [58], with the time complexity $2^{27}$.

3.3 Correlation Attack

The correlation attack was proposed by Siegenthaler in 1985 [73]. An important work in this area was elaborated by Meier and Staffelbach [60]. After them, Mihaljevi and Goli [63] was one of the promising work. Other important work is from Anderson [5], he started the search for the optimum correlation attack. They opened the world of cryptanalysis to correlation attack.

The correlation attack is defined as:

“The correlation attack exploits the existence of a statistical dependence between the keystream and the output of a single constituent LFSR.” [29]

In the Figure 5, we can see how works a stream cipher based on LFSR. The random noise in the Figure 5 is the keystream (LFSRs), the function $h(x)$ is to expand the secret key and the output $K_i$ is the secret key.

![Figure 5: The idea of a stream cipher with LFSR](image)

In the Figure 6 we can see the idea of the correlation attack. Using this attack Mihaljevi et al [62] recovered the internal state of LILI-128 [31] in a complexity time of the order $2^{35}$.

In the work of Wei et al [76], they presented a new correlation attack on nonlinear combining generators. In the moment, we have a good review about correlation attacks in Meier work [59] and in the work of Canteaut [29].

3.4 Fault Attack

The fault attack is a powerful cryptanalytic tool. It is widely applied in cryptosystems which are not vulnerable to direct attack. It is easy to find examples of fault
attacks in block ciphers, but the first application of this attack in stream cipher was developed by Hoch and Shamir \[54\].

In this attack, the attacker can apply some bit flipping faults to either the RAM or the internal register of the cryptographic device. However, he had only a partial control over their number, location and timing. This model tries to reflect a situation in which the attacker has the possession of the physical device, and the faults are transient rather than permanent \[54\].

A good work in this area was developed by Barenghi et al \[11\], they talk about this technique and where it can be applied. In their work has examples using stream ciphers and block ciphers.

In the work of Banika et al \[10\], they used in the Grain family \[52\] \[51\] \[2\] to recover the initial state of the LFSRs.

### 3.5 Distinguishing Attack

The distinguishing attacks in stream ciphers was introduced by Coppersmith et al \[32\]. In the Figure 7, they illustrated a style of cipher that can be used in this attack.

The technique defined by Coppersmith et al \[32\] is:

“An attack is specified by a linear function $l$, and by a decision rule for the following hypothesis-testing problem: The two distributions that we want to distinguish are:

**Cipher.** The Cipher distribution is $D_c = \langle l(x_j + y_j, NF(x_j) + z_j) \rangle_{j=1,2,...}$, where the $y_j, z_j$s are chosen at random from the appropriate linear subspace (defined by the linear process of the cipher), and the $x_j$s are random and independent.

**Random.** Using the same notations, the “random process” distribution is $D_r = \langle l(x_j, x'_j) \rangle_{j=1,2,...}$, where the $x_j$s and $x'_j$s are random and independent. We call the function $l$, the distinguishing characteristic used by attack.” \[32\]

Other relevant work in the area of distinguishing attack is the one from Englund et al \[44\]. They explained how the attack is used in block cipher. Moreover, they
explained how they create a new scenario for this attack. For an example, they used this new scenario in the LEX cipher [14] of the eSTREAM project [46].

An example of cryptoanalysis using this attack, is the work of Noferesti et al [66]. They reduced the complexity of the attack from $O(2^{32})$ to $O(2^{30.79})$ in the Bivium cipher [22], a simplified version of Trivium [27].

### 3.6 Chosen-IV Attack

In the Chosen-IV attack one of the relevant work in this area is from Joux and Muller [55]. To understand more about this attack we should bring the definition from Joux and Muller work:

“In general, a stream cipher produces a pseudo random sequence $PRNG(K, IV)$ from a secret key $K$ and an initialization vector $IV$. Then, the ciphertext $C$ is computed from the plaintext $P$ by:

$$C = PRNG(K, IV) \oplus P$$

The main idea behind the use of initialization vectors is to generate different pseudorandom sequences without necessarily changing the secret key, since it is totally insecure to use twice the same sequence.” [55]

Then, this attack exploits the weaknesses in the key scheduling algorithm of the stream cipher. The attack tried to extract from the memory, the initial state of the LFSR. Like the algebraic attack, in the subsection 3.2, the chosen-IV attack created a system of equations. This system of equations is created using the parts from the key recovered in the memory, more specifically in the vector $IV$.

In the work of Englund et al [45], they explained how this attack works. Also, they proposed different algorithms to improve the search and gave a practical demonstration of this algorithm.

Using this attack, Joux and Muller [55] recovered the key in a complex time of $2^{72}$ and used $2^{36}$ bytes of memory. For other examples, we can cite the work of Ding and Guan [40], who explored the weakness of the Grain-128 stream [51]. Biryukov et al [15] also used the attack in SNOW 3G to reduce the complexity to recover the key, they recovered the key in practical complexities $2^{57}$ time and $2^{33}$ keystream.

### 3.7 Slide Attack

The first time that the slide attack appeared in the literature was with Biryukov and Wagner [17]. They used the attack in TREYFER, WAKE-ROFB and others block ciphers. In 2000 they improved the slide attack and used in other block ciphers [18]. More recently slide attacks have been applied to other stream ciphers, such as Trivium with Priemuth-Schmid and Biryukov [69].

The main idea of the attack is defined by Biryukov and Wagner like:

“The idea is to slide one copy of the encryption process against another copy of the encryption process, so that the two processes are one round out of phase.” [17]

![Figure 8: A typical slide attack](image)
In the Figure 8 shows the typical slide attack. We let $X_0$ and $X'_0$ denote two plaintexts, with $X_j = F_j(X_{j-1})$ and $X'_j = F'_j(X'_{j-1})$. With this notation, we line up $X_1$ next to $X'_0$, and $X_{j+1}$ next to $X'_j$. Now, we suppose that $F_j = F_{j+1}$ for all $i \geq 1$; this is the assumption required to make the slide attack work. The observation is that if we have a match $X_1 = X'_0$, then we will also have $X_r = X'_r$. Therefore, we call a pair $(P, C), (P', C')$ of known plaintexts (with corresponding ciphertexts) a slid pair if $F(P) = P'$ and $F(C) = C'$ [17].

Using this technique Alhamdan et al [4], they demonstrated a slid property of the loaded state of the Sfinks cipher [23]. They demonstrated how to recover the key in the state update of the cipher as well.

3.8 Cube Attack

The cube attack is relative new. It has been introduced by Dinur and Shamir [41] in 2009.

“The attack exploits the existence of low degree polynomial representation of a single output bit (as a function of the key and plaintext bits) in order to recover the secret key. In order to derive the secret key, the attacker sums this bit over all possible values of a subset of the plaintext bits. The summations are used in order to derive linear equations in the key bits which can be efficiently solved.” [42]

According with Dinur and Shamir [42], this attack can be applied in almost any cryptosystem. In this paper we talked about stream cipher and fortunately the lats work of Dinur and Shamir is specific about cube attacks in stream ciphers [42].

There is many works that use the cube attacks, we have the work of Mroczkowski and Szmidt [64] using cube attack on Trivium [27]. Other important work is the work from Abdul-Latip et al[1], they extended the cube attack and combine with other techniques. Also, Zhao et al [78] used the same techniques in the PRESENT cipher [21] and the key search space can be reduced to $2^{8}$ for PRESENT-80 with $2^{8.95}$ chosen plaintexts and to $2^{9}$ for PRESENT-128 with $2^{9.78}$ chosen plaintexts.

3.9 Time-Memory Trade-off Attack

Cryptanalytic Time/Memory Tradeoff started with Hellman [53] in 1980. In his work, Hellman introduced this attack in block ciphers with $N$ possible keys in time $T$ and memory $M$ related by the tradeoff curve $TM^2 = N^3$ for $1 \leq T \leq N$.

However, Biryukov and Shamir [16] extended this attack for stream ciphers. The Time/Memory/Data Tradeoff Attack has two phases:

“During the preprocessing phase (which can take a very long time) the attacker explores the general structure of the cryptosystem, and summarizes his findings in large tables (which are not tied to particular keys). During the realtime phase, the attacker is given actual data produced from a particular unknown key, and his goal is to use the precomputed tables in order to find the key as quickly as possible.” [16]

In any time-memory tradeoff attack there are five key parameters:

- $N$: represents the size of the search space.
- $P$: represents the time required by the preprocessing phase of the attack.
- $M$: represents the amount of random access memory (in the form of hard disks or DVDs) available to the attacker.
- $T$: represents the time required by the realtime phase of the attack.
• $D$: represents the amount of realtime data available to the attacker.

In the work of Broek and Poll [25] has a comparison of time-memory trade-off attacks on stream ciphers. Other relevant work in this area is the Khoo and Tan [56], they used the time-memory-data trade-off attack on different block ciphers.

Using this attack, Verdult et al [74] recovered the key from Hitag2 stream cipher in 360 seconds. The importance of the Hitag2 is primarily used in RFID transponder systems manufactured by Philips/NXP, and used by many car manufacturers for unlocking car doors remotely [36].

3.10 Guess and Determine Attack

According with Ahmadi and Eghlidos [3] the Guess and Determine Attack is defined as:

"In GD attacks, the attacker first guesses (the values of) a set of state elements of the cryptosystem, called a basis; hence, the name. The basis can correspond to different elements of different states (multiple times). Next, she determines the remaining state elements and running key sequence, and compares the resulting key sequence with the observed key sequence. If these two sequences are equal, then the guessed values are true and the cryptosystem has been broken, otherwise the attacker should repeat the above scenario with other guessed values. " [3]

Moreover, Ahmadi and Eghlidos [3] improved the guess and determine(GD) attack using a heuristic. Using this new technique, they examined the resistance of the SOSEMANUK [12]. If they used the GD attack, then they have a result of $O(2^{222})$ complexity. Using the new algorithm they have a result of $O(2^{102})$ complexity.

Other application of this attack was proposed by Sha and Mahalanobis [72]. They used the GD attack on the A5/1 Stream cipher. Using the GD attack they recovered the key in a time complexity of $2^{58.5}$, which is much less than the brute-force attack with a complexity of $2^{64}$.

In the moment, Dunkelman and Keller [43] made a cryptanalysis of the stream cipher LEX [14] and in this cryptanalysis they used the GD attack.

An example of first work with GD attack was produced by Pasalic [68]. He started the GD attacks on LFSRs for stream ciphers.

4 CONCLUSION

In this work, we review the idea of block and stream ciphers. Explained the methods of operation of the block cipher. Also, we review the attacks in stream ciphers in the literature. We presented attacks and techniques derived from this attack. We explained the main idea of the attack and the application of the cipher. We saw there is ciphers, recommend by NIST and ECRYPT, susceptible of these attacks.

Unfortunately, we can not explain all the attacks in stream ciphers. The attacks in ciphers will grow up as the development of new ciphers are made. This will happen because we will develop ciphers based in other type of mathematical problem.

References

[1] Shekh Faisal Abdul-Latip, Mohammad Reza Reyhanitabar, Willy Susilo, and Jennifer Seberry. Extended cubes: enhancing the cube attack by extracting


[27] Christophe Cannire. Trivium: A stream cipher construction inspired by block cipher design principles. In SokratisK. Katsikas, Javier Lpez, Michael


