Not Today: Notoriously Difficult and Ambitious Cryptography

I Putu Gede Wirasuta[1], **Christzen Leonardy**[2].

[1,2] Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika (STEI), Institut Teknologi Bandung (ITB), Jalan Ganesha 10, Bandung 40132
E-mail: 13517015@std.stei.itb.ac.id, 13517125@std.stei.itb.ac.id

Abstract. Not Today algorithm is a block cipher algorithm that has a block size of 196 bit. This algorithm is implemented using a feistel network that has 15 iterations. The operations done in the round function are bit rotations, shifting, exclusive-or, substitution, and transposition. The encryption result is safe enough according to analysis.

Keywords: block, cipher, feistel.

1. Introduction

During this pandemic, we are forced to stay home, work from home, study from home, etc. It's a good thing we have so many technologies nowadays that we can still connect with others through the power of the internet and other information technology. However, if we use it just as is, our valuable information can be stolen, such as your credit card information. In order to fight against such misfortune, we need to encrypt our information. So that during its transmission, other people cannot view that data until the recipient accepts it then decrypt it.

Not Today is a block cipher algorithm that is implemented using good encryption characteristics called confusion and diffusion. Not Today also uses a feistel network with 15 iterations and a round function that has bit rotations, shifting, exclusive-or, substitution, and transposition..

2. Background

2.1. Block Cipher

A block cipher is a scheme which groups plaintext to a specific sized block then maps it to the same sized ciphertext block according to an encryption function[1]. Because plaintext are grouped into specific sized blocks, if the plaintext length is not a multiple of the block size then padding is added. The encryption function can be symmetric or asymmetric as long as it is injective (maps one input to at most one output)[2]. Encryption key of block cipher method need not be as long as the plaintext, in this case the encryption key can be repeated or used to generate block-specific keys (commonly known as round key). There are five modes of operation for block cipher.

2.1.1. Electronic Code Book (ECB)

Electronic code book mode of operation encrypts every block of plaintext independently of each other. It is named after the fact that the encryption process could be replaced by a code book containing mappings from plaintext to ciphertext. Figure 1 illustrates the encryption schematic of ECB.

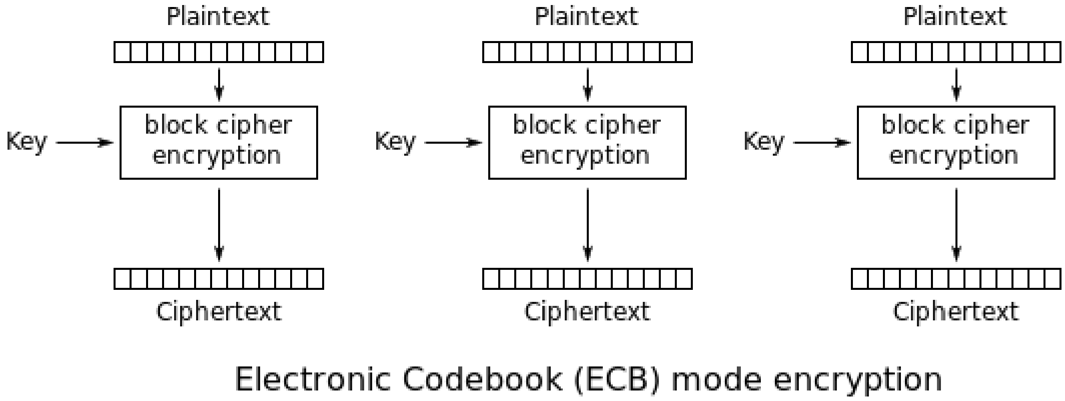

Figure 1. *Encryption schematic of ECB*

While it is easy and parallelizable, ECB is considerably the weakest mode of operation. It produces the same block of ciphertext with the same plaintext input. This makes statistical attack possible and produces a common phenomenon known as "ECB penguin" that still shows the outline of the original image. Another weakness of ECB is that it is prone to bit/byte flip to alter the plaintext due to the minimal effect of one ciphertext block's changes.

2.1.2. Cipher Block Chaining (CBC)

Cipher block chaining mode of operation encrypts every block of plaintext with regard to previous block. Specifically, it xor current block's plaintext with previous block's ciphertext on encryption and do the reverse on decryption. The first block of plaintext is xor'd with an initialization vector, ideally random and distinct for every encryption. Figure 2 and 3 illustrates the encryption and decryption schematic of CBC respectively.

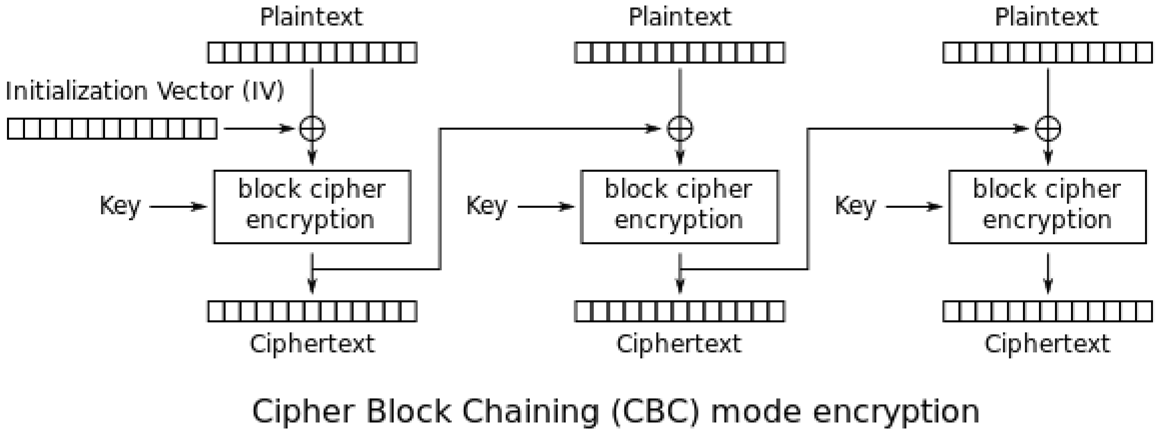

Figure 2. *Encryption schematic of CBC*

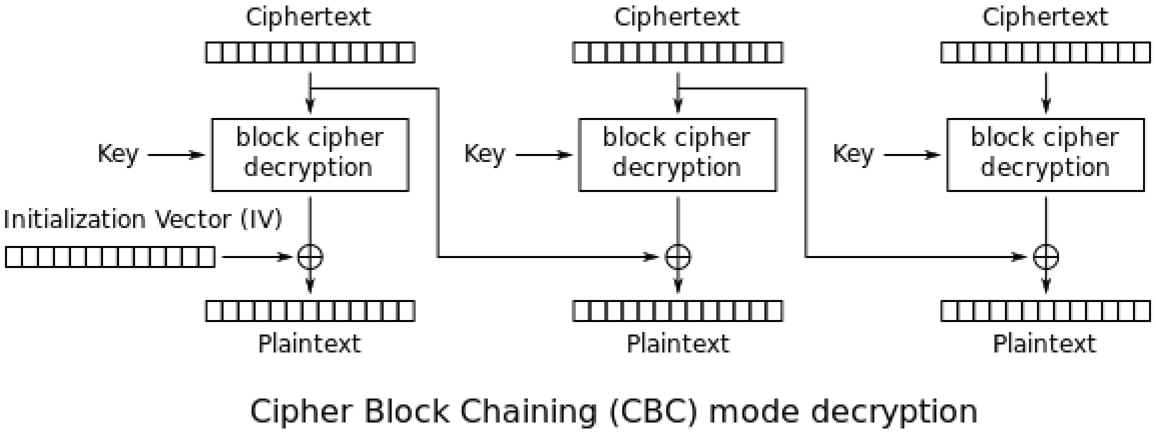

Figure 3. *Decryption schematic of CBC*

CBC is better than ECB in terms of security as it produces different ciphertext from different plaintext in different block arrangements. But this comes at the cost of inability to parallelize, which makes the encryption and decryption process more time consuming.

2.1.3. Cipher Feedback (CFB)

Cipher feedback mode of operation encrypts every n unit of an m sized block ($n \leq m$) with feedback from the previous encrypted unit. This mode was introduced to encrypt incomplete data in scenarios such as voice call. It requires a queue for every block with size of n bit that will shift s bit everytime a new data arrives. Due to its chained operation, it has similar strengths and weaknesses to CBC. Figure 4 and 5 illustrates the encryption and decryption schematic of CFB with n = 8 and s = 1.

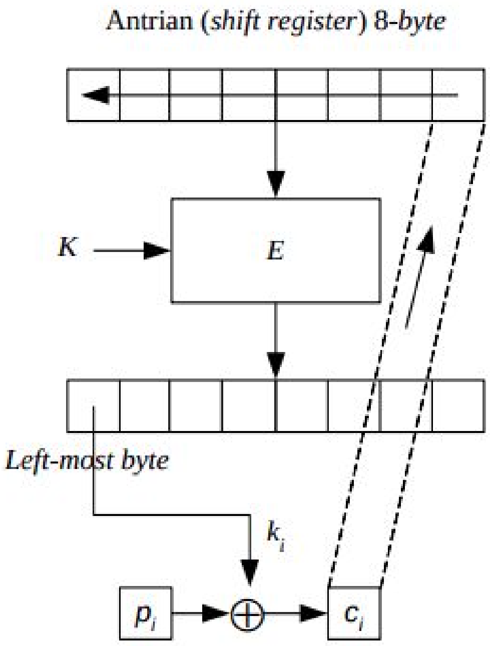

Figure 4. *Encryption schematic of CFB*

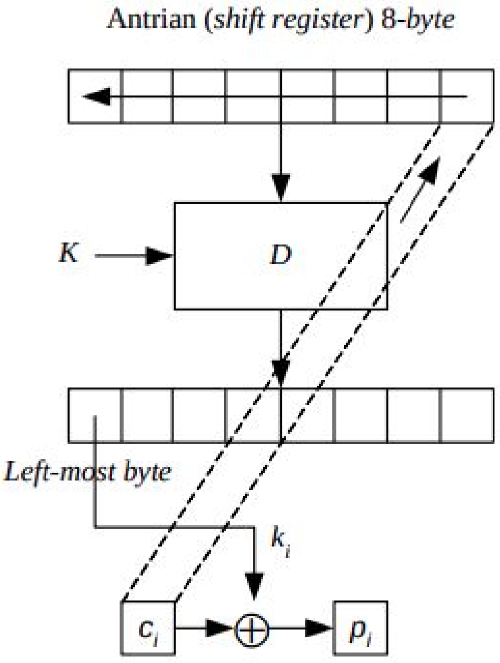

Figure 5. *Decryption schematic of CFB*

2.1.4. Output Feedback (OFB)

Output feedback mode of operation works very similarly to CFB with the difference of using output queue as a feedback source instead of encrypted unit. This makes a rather unique implication, OFB is the only chained mode of operation in which a bit/byte flip only affects current block's plaintext instead of affecting both current and next block's plaintext. It also implies that OFB is the only chained mode of operation vulnerable to bit/byte flip attack. Figure 6 and 7 illustrates the encryption and decryption schematic of OFB with n = 8 and s = 1.

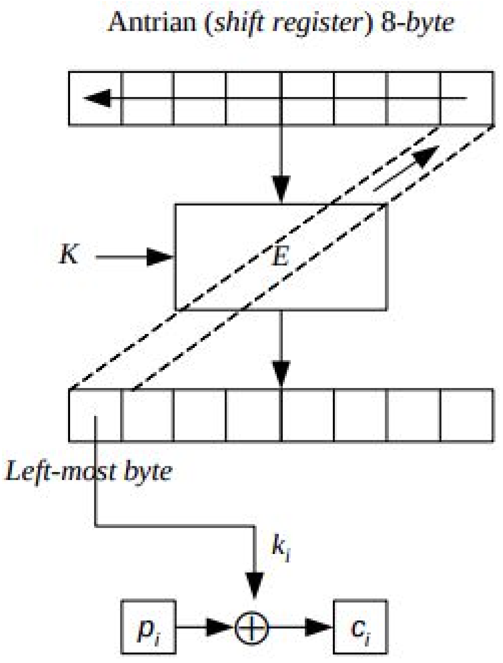

Figure 6. *Encryption schematic of CFB*

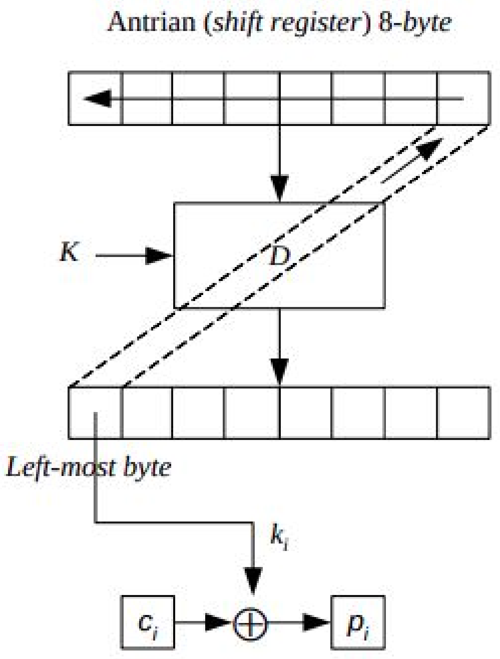

Figure 7. *Decryption schematic of CFB*

2.1.5. Counter Mode

Counter mode of operation encrypts every block of plaintext individually with regards to its position, hence the name counter. A block's position is encoded in the same block size as plaintext and ciphertext's block size. It is then fed into the encryption algorithm. Figure 8 and 9 illustrates the encryption and decryption schematic of counter mode.

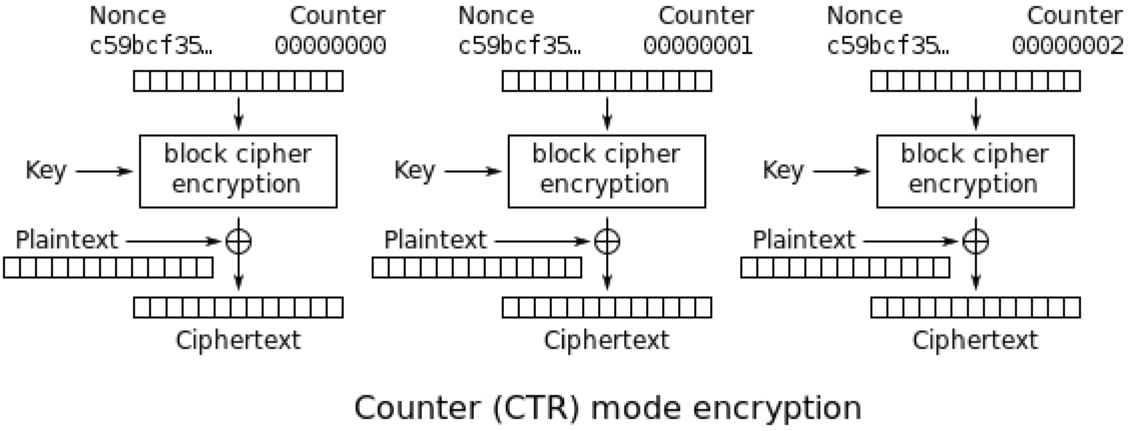

Figure 8. *Encryption schematic of counter mode*

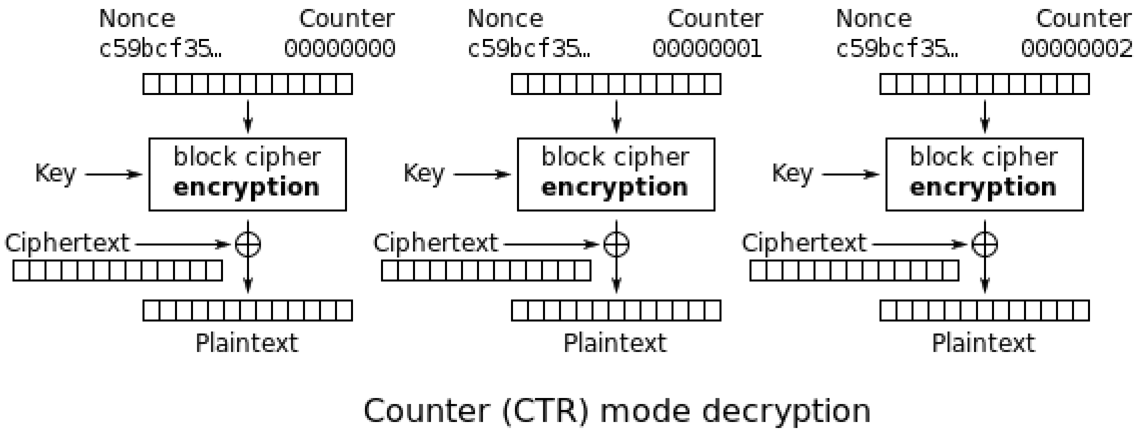

Figure 9. *Decryption schematic of counter mode*

Counter mode is better ECB as it produces different ciphertext from different plaintext in different block positions. But it is still worse than the chained methods because counter value is predictable. It is also prone to bit/byte flip because alteration of the current block's ciphertext only affects current block's plaintext.

2.2. Confusion and Diffusion

There are several aspects to creating a secure cryptography. In "A Mathematical Theory of Cryptography", Shannon identified two of the most important aspects of secure cryptography, confusion and diffusion[3]. Confusion is a concept that disconnects any relation between ciphertext and plaintext. In DES, it is implemented using S-box to map some input into another output seemingly at random. Diffusion is a concept that tells whether a single part change in plaintext or key should have as much effect on the ciphertext as possible. It is commonly implemented as some sort of permutation between elements. Both *confusion* and *diffusion* makes cryptanalysis harder to do.

2.3. Feistel Network

Feistel network is a scheme commonly used in block ciphers due to its flexibility. It consists of multiple rounds with the same structure. There are two main components of a feistel network, a key scheduling algorithm and an F function. Key scheduling algorithm is used to generate distinct round keys for every round. F function is used to encrypt half of the input at every round, this function is the same for every round. Some notable encryption standards using feistel network schemes include DES, GOST, and Blowfish. Figure 10 and 11 illustrates the encryption and decryption scheme of feistel network in round i.

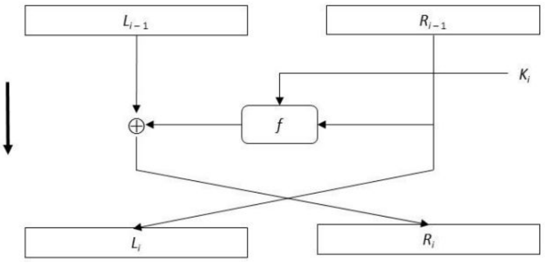

Figure 10. *Encryption* scheme *of feistel network in round i*

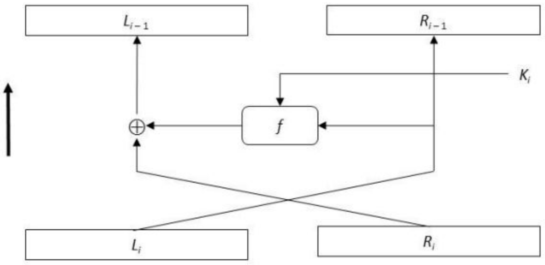

Figure 11. *Decryption schematic of feistel network in round i*

Figure 10 and 11 illustrates clearly where the flexibility of feistel network came from, reversible encryption/decryption and arbitrary F function. Both of these are made possible by reversing the output on every round and xoring the result of encrypting half of the input such that

$$L_i,\ R_i = R_{i-1},\ L_{i-1} \oplus f(R_{i-1}, K_i)$$

3. Proposed Algorithm

We propose NotToday, an encryption algorithm inspired by DES and AES. The inner workings of NotToday used a feistel network scheme due to its flexibility. In this chapter we define the key scheduling algorithm and round function algorithm needed to construct the feistel network.

3.1. Key Scheduling Algorithm

NotToday's round keys are calculated in iteration of rounds. The round key for i-th iteration is calculated with the following formula:

1. External key is divided into three parts L, M, R
2. If $i \leq 3$ then,
   - 2.a. $K_1 = (L \oplus M) >> 1$
   - 2.b. $K_2 = (L \oplus R) >> 2$
   - 2.c. $K_3 = (M \oplus R) >> 3$
3. Else,
   - 3.a. if $i \bmod 3 = 1$ then $K_i = (K_{x+1} \oplus K_{x+2}) >> i$
   - 3.b. if $i \bmod 3 = 2$ then $K_i = (K_{x+1} \oplus K_{x+3}) >> i$
   - 3.c. if $i \bmod 3 = 0$ then $K_i = (K_{x+2} \oplus K_{x+3}) >> i$

   with i is the current iteration, $x = (i \operatorname{div} 3 - 1) \times 3$, and $y >> n$ means circular right shift of y by n bit.

The formula is repeated until the specified amount of round keys is generated. We recommend at least 15 round keys are generated, one for each round.

3.2. Round Function Algorithm

NotToday's round function consists of 6 stages:

1. First, we convert the half-block and key into an 8x8 key table, the key table filling will start with filling all columns in a row then move on to the next row.
2. Every row of the half-block table will be rolled right circularly based on the first row of the key table of the same row.
3. A new key table will be made with its row value are the result of bitwise XOR of the row on the same position from the half-block and key tables.
4. The new key table will be converted to one-line key
5. Substitute with S-box
6. Then use roll shift right by the value of the last element on the result

This is used as the F function in the feistel network. We recommend using at least 15 round of

4. Experiment and Results

4.1. Experiment

This experiment is done on the operation modes available, which are ECB, CBC, and Counter. In this experiment, we measure the execution time during encryption and decryption. The message we use in this experiment is 64 bytes and 1024 bytes. The measured time (in seconds) in this experiment is the following

Mode	Process	64 bytes	1024 bytes
ECB	Encryption	0.00024581	0.010112
	Decryption	0.00029349	0.01349
CBC	Encryption	0.00027323	0.010789
	Decryption	0.00030184	0.011185
Counter	Encryption	0.0002625	0.011389
	Decryption	0.00036788	0.014789

4.2. Brute Force Analysis

Brute force attack is done by trying all key combinations to decrypt a ciphertext. It is guaranteed you will find the exact key for the decryption. But it comes with a cost, it takes too much time to find the key. In Not Today algorithm, we use a key with a size of 196 bit. That means the key combinations are as many as 2^{196}. Using the assumption that a computer can find and try 10^9 combinations in a second. Assuming $2^{10} \approx 10^3$, then $2^{196} \approx 2^6 \times 10^{57}$. It means the computer needs $2^6 \times 10^{48}$ seconds or 2.03×10^{42} years to complete the task. Since it takes a long time to finish the attack, this algorithm is safe from brute force attack.

4.3. Confusion and Diffusion Analysis

Confusion analysis is done by comparing the frequency of plaintext and ciphertext. Comparing plaintext and ciphertext frequency, it can be seen that the ciphertext histogram is evenly distributed among all possible values, indicating no relationship between plaintext and ciphertext. Diffusion analysis is done by creating a slightly different plaintext as demonstrated in the table below.

No	Plaintext	Ciphertext (Hex)
1	BBBBBBBBBBBBBBBB	29 9f a8 a8 76 3e 7a d4 32 a1 b8 69 bc d2 00 cf
2	ABBBBBBBBBBBBBBB	2a 9c ab ab 75 3d 79 d7 31 a2 bb 6a bf d1 03 cc

5. Conclusion

This algorithm is good enough in encrypting and decrypting messages. It is also implemented with confusion and diffusion in mind. With its long key size, it makes brute force attack long to finish. The speed of the encryption and decryption is fast enough so that it can be used without using a strong hardware for the computation. The suggestion for the next work is to use more analysis in order to find the cons of this algorithm.

6. References

[1] Munir, Rinaldi. (2020). Kriptografi Modern (Bagian 4: Prinsip Perancangan Block Cipher).

[2] Menezes, Alfred J.; van Oorschot, Paul C.; Vanstone, Scott A. (1996). "Chapter 7: Block Ciphers". Handbook of Applied Cryptography

[3] Anderson, David R. (2008). "Information Theory and Entropy". Model Based Inference in the Life Sciences: A Primer on Evidence