Encryption Modes

Siyu Zheng (Vera), Yulin Chen (Mike), Chaoran Yuan (Richard)

【Abstract】 The contents will include ECB, CBC, CFB and OFB modes. And as the last part, a simple known-plaintext attack is tested and different situations concerning error propagation in different modes are compared.

1. Introduction

Encryption modes or modes of operation, which are very fundamental in block cipher, refer to the many ways to make the input of an encryption algorithm different.

But before we step into these modes, a brief introduction of Block cipher must be the first to come.

Block cipher is an important part of many encryption systems. It can be utilized in random number generation, stream ciphers, MAC, Hash and so on. In a block cipher, what we do first is to break the plaintext X into blocks (or say, groups) with the same length n. And then, as is shown in the following picture, we deal with the text block by block. After encryption, we transmit the ciphertext of this group to the decryption side.

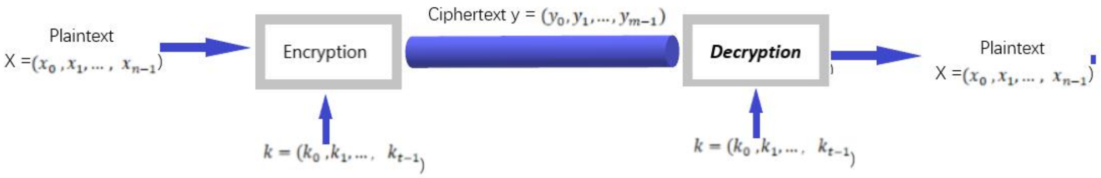

Plaintext
X =$(x_0, x_1, \dots, x_{n-1})$
Encryption
Ciphertext y = $(y_0, y_1, \dots, y_{m-1})$
Decryption
Plaintext
X =$(x_0, x_1, \dots, x_{n-1})$
$k = (k_0, k_1, \dots, k_{t-1})$
$k = (k_0, k_1, \dots, k_{t-1})$

Fig.1.1 Block Cipher

In a word, block cipher is about encrypting and decrypting a text separately.

Then, to be specific, let's come to these two standards of block cipher, DES and AES. Data encryption standard (DES) was set as Federal Information Processing Standard in 1977. Its block size, which means the length of one block, is 64 bits while the key size is 56 bits. DES uses confusion and diffusion to prevent statistical analysis. And to improve its safety, which was quite limited due to its key size, double and even triple DES were introduced.

Besides, AES, which stands for Advanced encryption standard, was designed for replacing DES. It has longer and flexible block size and key size which means that the user can choose a specific size. It can be 128, 192 or 256 bits. The encryption process of AES is made of four parts, which are ByteSub, ShiftRow, MixColumn and AddRoundKey. And what makes another new part in AES is the counter (CTR) mode. These two standards are widely used.

And then, it's time to turn our focus back to encryption modes. The questions we will try to answer are that why do we need these modes in block ciphers and why so many kinds of modes? To begin with, as have been introduced, block ciphers deal with blocks separately. If the text includes two same blocks, we had better prevent them to be encrypted into the same ciphertext, because if we choose to ignore the danger brought by duplication, it would make the system too easy to break into. That's why those experts introduced encryption modes to guide us dealing with these blocks. And different modes were designed for DES initially because that's how we use block ciphers in different situations.

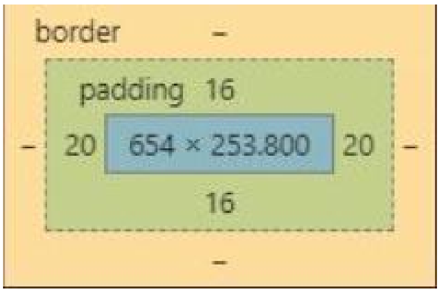


Fig.1.2 (a) Padding in a picture

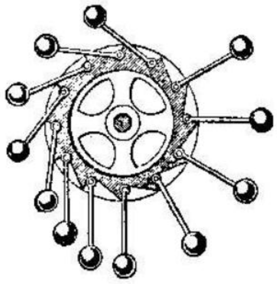


Fig.1.2 (b) perpetual motion machine

The next part is about two concepts in encryption modes. The first is padding. To describe it intuitively. Let's imagine fitting a picture into a frame, as is shown in fig.1.2.(a), the frame is always slightly larger than the size of picture, and what we do is to put some material around the original picture to fit the size of frame. And that part is called padding. In encryption modes, padding refers to extra data added to fit the size of a block. The other concept is initialization vector, or say, IV. If we describe encryption as a perpetual motion machine. IV would be the initial power to get the machine started. In encryption modes, IV is some data used in both encryption and decryption of the first block.

2. Electronic CodeBook Mode & Cipher Block Chaining Mode

In order to convey with each other, we need to encrypt our information to prevent adversary to see it. In this way, we have to use different encryption mode. Here are five common encryption methods include Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), and Counter (CTR).

Electronic Codebook (ECB) divide the whole plaintext into several identical length segments, and each segment is encrypted by using the same key. The whole encryption process is simple and effective. By the way, this mode is also deficient. As the same key encrypt each plaintext, the ciphertext lack of diffusion. When using this mode to encrypt the image with a large area of same color, it will encrypt the text block into the same ciphertext block, through which data cannot be hidden. For example, if we use Electronic Codebook mode to encrypt a picture of penguin, we can still see the shape and appearance of this little penguin. In this way, it is not the best mode to encrypt plaintext as encryption protocol.

Cipher Block Chaining (CBC) also separates the whole plaintext into several equal length segments, and XOR (exclusive disjunction) each segment, then using the same key to encrypt each segment. Here is the specific encryption process. Firstly, Cipher Block Chaining mode encrypt the first segment by using the Initialization Vector to XOR. Then, using the key to encrypt the segment so that we can get the first ciphertext segment. Next, using the ciphertext segment to XOR the next plaintext segment until all the plaintexts are encrypted. Absolutely, since the Initialization Vector is used in the first block encryption, every next ciphertext block is different from the previous one and would not appear as repeated ciphertext as the Electronic Codebook mode.

Obviously, each plaintext segment needs to use the encrypted information of the previous ciphertext to encrypt, so encryption cannot be parallelized. Comparatively, decryption can be parallelized. This is due to that we should use the same key to decrypt first, and each segment ciphertext has already know the previous ciphertext. It can be parallelized when decrypting. Absolutely, Cipher Block Chaining encryption mode also has its disadvantages. Its main drawback is that the encryption is sequential and can not be parallelized. In this way, not only encryption would become busy, but also information must be divided into the size of the key. When using this mode to encrypt, we must type very careful, because once the man types the Initialization Vector by mistake, then the encrypted ciphertext would be affected and changed, so that the correct plaintext cannot be decrypted.

Comparing these two encryption modes, although they have their own disadvantages, the good point of electronic are simple structure and convenient encryption, while cipher block chaining is characterized by complex encryption but not easy to be stolen by adversary.

3. Cipher FeedBack Mode & Output FeedBack Mode

The next mode we are going to introduce is Cipher FeedBack (CFB) Mode. It can change block passwords into self-synchronized stream passwords, very similar to CBC, the working process is also very similar, the decryption process of CFB is almost the reverse CBC encryption process:

Ci=EK(Ci-1) xor Pi

Pi=Ek(Ci-1) xor C

C0=IV

As is shown above, the CFB mode changes the encryption output of the same black using a comparingly complex way, and thanks to that, a block cipher is turned into a stream cipher. Thus, it is ideal for encrypting real-time data while padding is not required for the last block. What's more, decryption using the CFB mode can be parallelized and data smaller than the packet can be encrypted in a timely manner.

Output FeedBack (OFB) mode is another mode that change the vector due to certain feedbacks. As with other stream ciphers, the rollover of one bit in ciphertext results in the rollover of the same bit in plaintext. It can change the block password into a synchronous stream password. It produces a block of the key stream and then xor it with the plaintext block to get the ciphertext:

Ci=Pi xor Oi

Pi=Ci xor Oi

Oi=Ek(Oi-1)

O0=IV

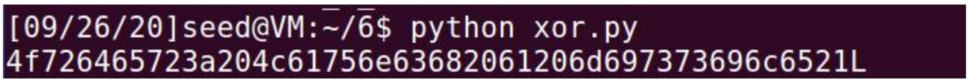
Concerning OFB mode, an active attack on plaintext is possible. Some other disadvantages are that, OFB is not conducive to parallel computing and the error propagation of OFB mode is quite impressive.

4. Error propagation & Attack

Error propagation happens when certain bits or limited parts in the ciphertext are maliciously tampered by an adversary or lost in transmission. Due to the error brought by ciphertext, there will be certain length of wrongly decrypted contents in the plaintext compared with the original one. However, different modes used in the system will decide how far the error is going to propagate or transmit in the text. And this kind of difference is what we are interested in.

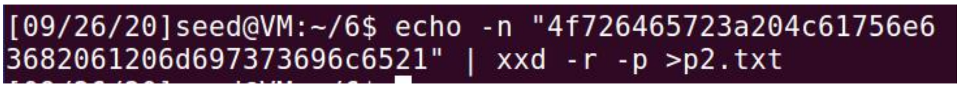
In ECB mode, the error can only affect the block which it belongs because ECB mode deals with blocks separately and there does not exist connections between them. In CBC mode, the error will happen in both the block it belongs and the next encrypted block. That's because the third block does not include the wrong ciphertext block in decryption.
In CFB mode, the error will affect many blocks after the wrong one until the error is pushed out of the IV. And in OFB mode, as the decryption process only uses the ciphertext in the last step to XOR, thus no error propagation is about to happen.

What we should notice while using encryption modes is that, we should always change the initialization vector. With the same IV, we may always get the same ciphertext. And to show the danger of repeating IV, here comes a simple but effective known-plaintext attack. Supposing I am the adversary and already know the plaintext P1 and its ciphertext C1. Now, after receiving the ciphertext C2, we can get plaintext P2 easily if these two encryptions have used the same IV.



```
[09/26/20]seed@VM:~/6$ python xor.py
4f726465723a204c61756e63682061206d697373696c6521L
```

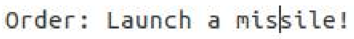
Fig.4.1 (a) Get the XOR



```
[09/26/20]seed@VM:~/6$ echo -n "4f726465723a204c61756e6
3682061206d697373696c6521" | xxd -r -p >p2.txt
```

Fig.4.1 (b) Turn back to binary

We first turn P1, C1, C2 to hex and using as is shown in Fig.4.1(a), using the program we write to get the XOR output of P1, C1, C2. Then, in Fig.4.1(b), we turn the output back to binary and write into the file P2.txt. And if we open this file, we can get the secret sent by the system.



Order: Launch a missile!

Fig.4.2 The secret

This simple attack can only be useful in CFB mode using the same IV. If we change the IV, the XOR attack will just not work.

Reference：

【1】SEED Labs. Wenliang Du, Syracuse University. 2018

【2】https://blog.csdn.net/zy_strive_2012/article/details/102520356

【3】https://blog.csdn.net/chengqiuming/article/details/82355772

Labor of division

Siyu Zheng (Vera): CFB & OFB modes.

Yulin Chen (Mike): ECB & CBC modes.

Chaoran Yuan (Richard): The introduction part, error propagation and the attack part.