Cifrado de bloques en Criptografia

Valentina S. Vispo

Octubre 2023

Índice

1 AES

1.1 Sobre AES

Posee bloques de 128 bits, claves que pueden ser de 128, 192 y 256 bits.

Los cifrados de bloques para poder aplicarlos en mensajes de size arbitrario hace falta un modo de operacion.

Estos modos tratan confidencialidad, no proveen integridad.

NIST especifica 5 modos y cada uno tiene ventajas y desventajas:

Mode	Advantage	Disadvantage
Electronic CodeBook (ECB)	• Simple • fast • Support for parallel (encryption/decryption)	• Duplicate data in plaintext will be reflected in the ciphertext • The plaintext can be operated by deleting or replacing the ciphertext. • If the ciphertext packet is damaged, it will affect the plaintext. • Can't resist replay attacks • Should not be used
Cipher Block Chaining (CBC)	• Support for parallel computing (decryption) • Ability to decrypt any ciphertext packet • Duplicate data in plaintext will not be reflected in the ciphertext	• Do not Support for parallel computing (Encryption) • Wrong blocks affect all following blocks
Cipher-FeedBack (CFB)	• No padding • Support for parallel computing (decryption) • Ability to decrypt any ciphertext packet • Can be prepared for encryption and decryption first	• Do not Support for parallel computing (Encryption) • Can't resist replay attacks • Wrong blocks affect all following blocks
Output-FeedBack (OFB)	• No padding • Can prepare encryption and decryption in advance • Encryption and decryption use the same structure • Bad blocks only affect the current block	• Do not Support for parallel computing • Mallory can change some ciphertext damaged plaintext
CounYeR (CTR)	• No padding • Can prepare encryption and decryption in advance • Support for parallel computing • Encryption and decryption use the same structure • Bad blocks only affect the current block	• Mallory can change some ciphertext damaged plaintext

1.1.1 Relleno

En la practica los esquemas requieren que siempre se agregue relleno, aunque el mensaje original tiene el size adecuado.

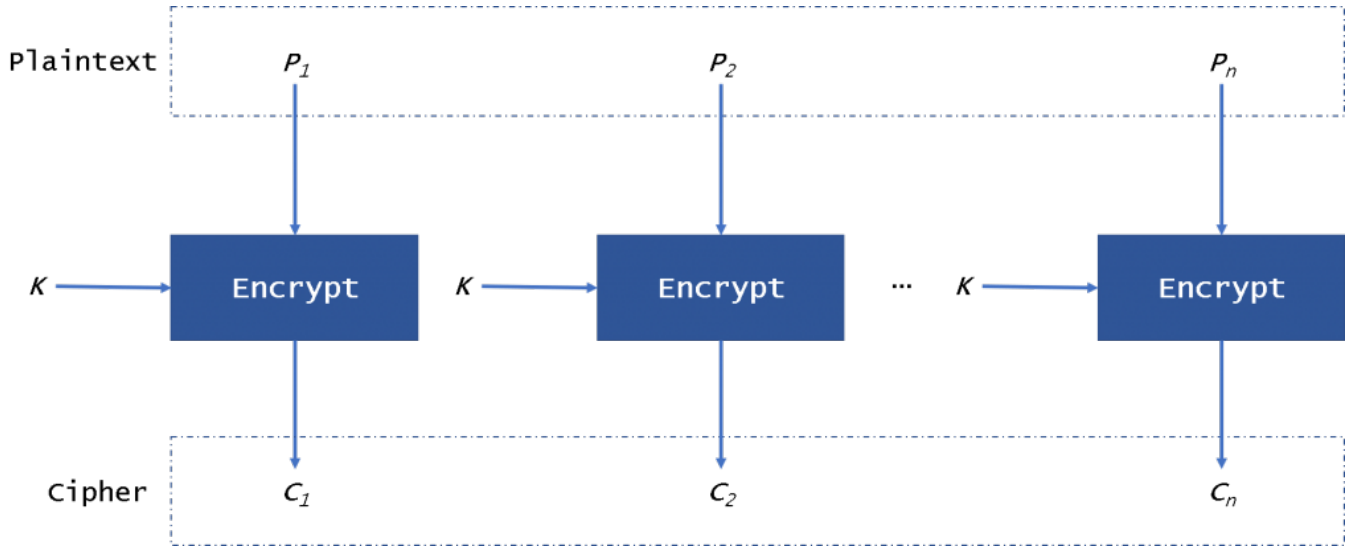
Por este relleno obligatorio, siempre el texto cifrado sera mas largo que el texto claro.

1.2 AES modo ECB

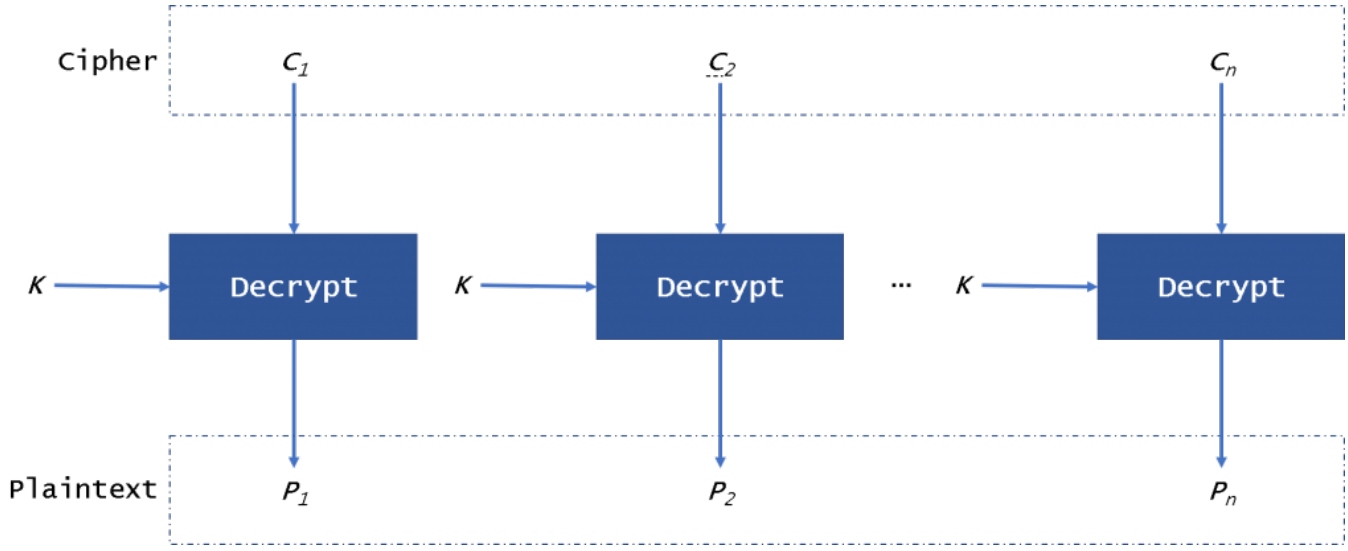
Cuando se utiliza AES en modo ECB (Electronic Code Book) el cifrado sobre el texto original se aplica sobre cada bloque en forma individual

Cifrado

$$C_i = E_k(P_i)$$



Descifrado



1.2.1 Propiedades

1. Rapido
2. Acceso directo: no es necesario descifrar los bloques anteriores
3. Paralelizable

4. Un error de cifrado sobre C_i solo afecta al texto P_i

1.2.2 Problemas

1. No oculta adecuadamente los patrones existentes en el texto claro
2. No admite procesamiento
3. Maleable: un atacante puede borrar o intercambiar bloques de texto cifrado, alterando asi el texto claro recibido

En conclusion, no es seguro y no deberia utilizarse en la practica.

2 Desafio: Falsificación en modo ECB

2.1 Enunciado

En este desafío, el servidor permite a un usuario registrarse con un correo electrónico, y devuelve un perfil, opcionalmente cifrado con AES en modo ECB. Si se le pasa un segundo par encrypted=true en la consulta, el servidor devuelve el mismo perfil, cifrado con AES en modo ECB, y codificado en base64.

Se debe enviar un mensaje cifrado de manera tal que al descifrarlo incluya el par $role = admin$.

- El algoritmo utilizado es AES, por lo que los bloques son de 128 bits (16 bytes)
- El texto cifrado enviado como respuesta debe descifrarse a una consulta (query string) válida:
  - Debe tener exactamente un campo *user*, con una dirección de correo válida.
  - Debe contener al menos un campo *id*
  - Debe contener al menos un campo *role* con el valor *admin*
  - Puede contener otros pares atributo-valor aparte de los mencionados
- Como el modo utilizado es ECB, se utiliza relleno (padding). El relleno utilizado es el estandarizado en PKCS7: se aplican tantos bytes como sea necesario para que el mensaje sea múltiplo del tamaño de bloque, y el valor de esos bytes es la longitud del relleno. Si el mensaje original era múltiplo del tamaño de bloque, el relleno es un bloque adicional con todos sus bytes con el valor 16. Al descifrar el mensaje, el relleno debe ser válido.

2.2 Analisis

Los cifrados de bloques operan sobre grupos de bits de longitud fija, denominados bloques, en otra secuencia del mismo size. Son inversibles, y cada bloque de texto cifrado mapea a un unico bloque de texto claro (permutacion).

Sean E una funcion de cifrado, D una funcion de descifrado, K una clave, P un bloque de texto claro y C un texto cifrado tenemos que:

Cifrar	—	**Descifrar**
$E_k(P) = C$	—	$D_k(C) = P$

El idea es alterar el mensaje cifrado de manera tal que al descifrarlo incluya el par $role = admin$.

Esta alteración es posible porque el modo ECB es maleable. Es posible intercambiar o insertar bloques en el texto cifrado de manera tal que al descifrarlo el texto claro haya sido modificado de manera predecible.

La clave para resolver este desafío es comprender que podemos armar un mensaje falsificado combinando bloques que contienen texto seleccionado por nosotros. En este caso, el único texto bajo nuestro control es el correo electrónico, por lo que será necesario enviar distintas direcciones de manera tal que ciertos bloques contengan el texto que nosotros deseamos.

La query tiene el siguiente formato: $user = valen.vispo@gmail.com\&id = 555\&role = user$

$user =$ — tiene 5b

$\&id = 287\&role = user$ — tiene 17b

Necesito enviar $admin = true$, para ello necesito dividir en bloques de 16b en los cuales:

1. Uno de estos bloques debe terminar en *role* (4b)

2. Mientras que otro debe empezar con *admin* (5b)

Para asi poder manipular el orden de los bloques para falsificar el mensaje.

Puntos importantes:

1. El size del bloque determina la cantidad de bloques posibles — en nuestro caso 128 bits

2. El size del bloque, a su vez, determina el numero de permutaciones posibles — en nuestro caso $2^{128}!$

3. El size de la clave determina la cantidad de permutaciones que puede producir el cifrado. Ademas este size determina el esfuerzo requerido para un ataque de fuerza bruta — en nuestro caso se requiren 2^{128} opercioness de cifrado

4. El ultimo bloque posee padding

2.3 Solucion

Pasos

1. Realizar 2 requests donde manipulo el email a cambiar. Una en texto plano y otra encriptada.

2. El texto encriptado esta codeado en b64, asi que es necesario decodearlo. Ademas debo obtener los cloques de 16 bytes.

```
# Request [1]
Email to send: ddddddddrole@bbbbbbbbbbbbbb.admin
Texto Claro: user=ddddddddrole@bbbbbbbbbbbbbb.admin&id=810&role=user

# [Request [1] Texts plain to encrypt]
Fisrt: user=ddddddddrole — No interesa utilizar
Second: @bbbbbbbbbbbbbb. — No interesa utilizar
```

Third: admin&**id**=810&rol
Potential 4th: e=user – No interesa utilizar

Encrypted response: rWR8i9JRhwDPhUxcHypEqJ2xMVTIgNe5PDailFsDoFM/428
BMpKtb8NyWiUFrI/cC7r4y4jU2WLA8obIeKaQ9w==

[Request [1] Encrypt]
Fisrt: b'\xadd|\x8b\xd2Q\x87\x00\xcf\x85L\\\x1f*D\xa8' – No interesa utilizar
Second: b'\x9d\xb11T\xc8\x80\xd7\xb9<6\xa2\x94[\x03\xa0S' – No interesa utilizar
Third: b'?\xe3o\x012\x92\xado\xc3rZ%\x05\xac\x8f\xdc'
Potential 4th: b'\x0b\xba\xf8\xcb\x88\xd4\xd9b\xc0\xf2\x86\xc8x\xa6\x90\xf7' – Sera el utilizado para el padding!

Request [2]
Email to send: ddddddddddrole@bbbbbbbb.admin
Texto Claro: user=ddddddddddrole@bbbbbbbb.admin&**id**=694&role=user

[Request [2] Texts plain to encrypt]
Fisrt: user=ddddddddddd – No interesa utilizar
Second: role@bbbbbbbb.ad – No interesa utilizar
Third: **min**&**id**=694&role=
Fourth: user – No interesa utilizar

Encrypted response:
XjVLV9OLZxZMJ9w0eMAS9XJTgp9IR2FV13DW4b9DDvCpR9wPKWpFgndTLKIGzGzuSJE2o
+xVKBmB3tJWRfNg==

[Request [2] Encrypt]
Fisrt: b"^5KW\xd3\x8bg\x16L'\xdc4x\xc0\x12\xf5" – No interesa utilizar
Second: b'rS\x82\x9fHGaU\xd7p\xd6\xe1\xbfC\x0e\xf0' – No interesa utilizar
Third: b'\xa9G\xdc\x0f)jE\x82wS,\xa2\x06\xccl\xee'
Potential 4th: b'H\x916\xa2\xf4\xfe\xc5R\x81\x98\x1d\xed%d_6' – No interesa utilizar

3. Guardo el par de text plain y encripted text

```
# De esta forma puedo agregar a la query lo siguiente: role=admin
relleno_email = "ddddddd" # 7b
relleno_at = "@bbbbbbbbbbbbbb." # 16b
email_to_change = f"{relleno_email}role{relleno_at}admin"
# ddddddddrole@bbbbbbbbbbbbbb.admin
```

4. Vuelvo a realizar el paso 1, 2 y 3 para generar el otro bloque

```
# De esta forma puedo agregar a la query lo siguiente: role=admin
relleno_email_2 = "ddddddddddd" # 11b
relleno_at_2 = "@bbbbbbbb." # 10b
email_to_change_2 = f"{relleno_email}role{relleno_at}admin"
```

ddddddddddddrole@bbbbbbbb.admin

5. Luego selecciono los bloques cifrados que me interesan para generar el siguiente mensaje de texto plano:

'user=ddddddddddd'
'role@bbbbbbbbbad'
'min&id=925&role='
'admin&id=810&rol'
'e=user'

user=ddddddddddddrole@bbbbbbbbbadmin&id=925&role=admin&id=810&role=user

r_1c_i: corresponde a la primer request — r_2c_i: corresponde a la segunda request

Luego el mensaje sera:

$$m = r_2c_1||r_2c_2||r_2c_3||r_1c_3||r_1c_4$$

b'
XjVLV9OLZxZMJ9w0eMAS9XJTgp9IR2FV13DW4b9DDvCpR9wPKWpFgndTLKIGzGzuP
+NvATKSrW/DclolBayP3Au6+MuI1NliwPKGyHimkPc='

2.3.1 Como correr la solución

```
python3 challenges/5_ECB_forge.py
```

2.3.2 Evitar el problema

Como ya se menciono anteriormente, **AES en modo ECB no debe ser utilizado porque no es seguro**. La forma de evitar el problema es **utilizar AES con otro modo que si sea seguro**.

2.4 Herramientas

En este desafio en particular no se uso herramientas extras.

2.5 Referencias

1. https://ciberseguridad.diplomatura.unc.edu.ar/cripto/doc/ecb-forge.html
2. https://tex.stackexchange.com/questions/120789/factorial-spacing-inside-equation
3. Eliminar padding
4. Advanced Encryption Standard - AES
5. https://es.wikipedia.org/wiki/Modos_de_operaci%C3%B3n_de_una_unidad_de_cifrado_por_bloques
6. Notas de clase - borrador