CRIPTOSISTEMAS SIMÉTRICOS

Práctica 1

Índice

1. Archivo de 1024 relleno de ceros

```
pedro@ubuntu:~/Desktop/spsi/p1$ dd if=/dev/zero of=~/Desktop/spsi/p1/input.bin bs=1 count=128
128+0 records in
128+0 records out
128 bytes (128 B) copied, 0,000798205 s, 160 kB/s
```

Comando:

dd if=/dev/zero of=~/Desktop/spsi/p1/input.bin bs=1 count=128

```
00000000  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000014  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000028  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
0000003C  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000050  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000064  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000078  00 00 00 00  00 00 00 00                                          ........
```

2. Archivo de 1024 relleno de ceros pero con un bit a 1 entre los bits 130 y 150

```
pedro@ubuntu: ~/Desktop/spsi/p1
00000000  00 00 00 00  00 00 00 00  0F 00 00 00  00 00 00 00  00 00 00 00  ....................
00000014  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000028  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
0000003C  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000050  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000064  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ....................
00000078  00 00 00 00  00 00 00 00                                          ........
```

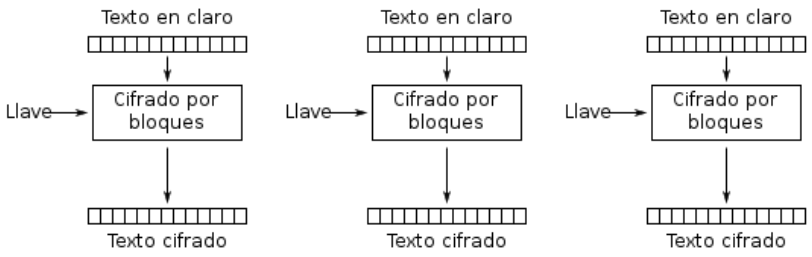

3. AES 256 con clave y vector de inicialización

El vector de inicialización se mantiene fijo a "**0123456789abcdef**".

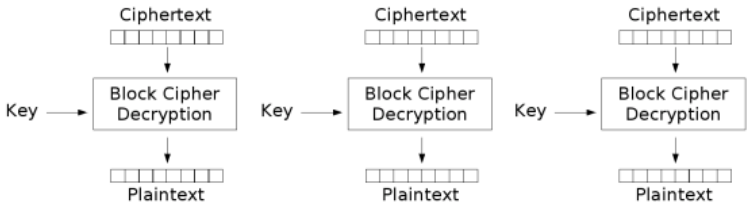
La clave, tiene que ser de 256 bits, o lo que es equivalente a ~~**32**~~ **64** caracteres en hexadecimal, por lo que se va a elegir la siguiente clave en todos los casos "**9876543210fedcbaabcdef0123456789**". **Error: Serían 64 caracteres, ya que 256/4bit = 64 caracteres en hexadecimal.**

a. Modo ECB

En el modo ECB el texto a cifrar se divide en bloques y cada uno de los bloques se cifra por separado, por lo que se si se modifica un bit en uno d ellos bloques, sólo se debería modificar un bloque cifrado.



Cifrado en modo de operación Electronic Codebook (ECB)



Electronic Codebook (ECB) mode decryption

Comandos:

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ecb -K 9876543210fedcbaabcdef0123456789 -iv 01234
56789abcdef -in input.bin -out input_aes_256_ecb_con_key_y_iv.bin
pedro@ubuntu:~/Desktop/spsi/p1$ ls
input1.bin  input_aes_256_ecb_con_key_y_iv.bin  input.bin  symmetric.pdf
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ecb -K 9876543210fedcbaabcdef0123456789 -iv 01234
56789abcdef -in input1.bin -out input1_aes_256_ecb_con_key_y_iv.bin
pedro@ubuntu:~/Desktop/spsi/p1$ ls
input1_aes_256_ecb_con_key_y_iv.bin  input_aes_256_ecb_con_key_y_iv.bin  symmetric.pdf
input1.bin                           input.bin
pedro@ubuntu:~/Desktop/spsi/p1$
```

Comando para input:

openssl enc -aes-256-ecb -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input.bin -out input_aes_256_ecb_con_key_y_iv.bin

Comando para input1:

openssl enc -aes-256-ecb -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input1.bin -out input1_aes_256_ecb_con_key_y_iv.bin

Resultados:

Input 0:

```
00000000   2C 2D BE 51  CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4  2C 2D BE 51
00000014   CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4  2C 2D BE 51  CE 02 B2 BF
00000028   1A E9 A9 19  9E 90 94 A4  2C 2D BE 51  CE 02 B2 BF  1A E9 A9 19
0000003C   9E 90 94 A4  2C 2D BE 51  CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4
00000050   2C 2D BE 51  CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4  2C 2D BE 51
00000064   CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4  2C 2D BE 51  CE 02 B2 BF
00000078   1A E9 A9 19  9E 90 94 A4  6C 6F 3C 70  1C CC C1 C4  1E 4E 27 6B
0000008C   D8 67 09 E5
```

Input1:

```
00000000   8E 59 B3 DB  C5 94 6D 0F  72 D4 D8 DF  CE B7 D0 96  2C 2D BE 51
00000014   CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4  2C 2D BE 51  CE 02 B2 BF
00000028   1A E9 A9 19  9E 90 94 A4  2C 2D BE 51  CE 02 B2 BF  1A E9 A9 19
0000003C   9E 90 94 A4  2C 2D BE 51  CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4
00000050   2C 2D BE 51  CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4  2C 2D BE 51
00000064   CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4  2C 2D BE 51  CE 02 B2 BF
00000078   1A E9 A9 19  9E 90 94 A4  6C 6F 3C 70  1C CC C1 C4  1E 4E 27 6B
0000008C   D8 67 09 E5
```

Diferencias

Input:

```
1. 00000000   2C 2D BE 51  CE 02 B2 BF  1A E9 A9 19  9E 90 94 A4  2C 2D BE 51
   ,-.Q............,-.Q
```

Input1:

```
1. 00000000   8E 59 B3 DB  C5 94 6D 0F  72 D4 D8 DF  CE B7 D0 96  2C 2D BE 51
   .Y....m.r.......,-.Q
```

Conclusiones

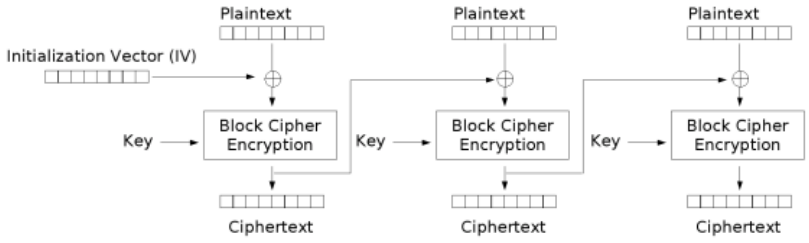
Se puede ver como al cambiar un solo bit, se modifica un solo bloque de 128 bits. Esto es debido a que AES utiliza un tamaño de bloque de 128 bits, y al usar el modo ECB, sólo se ve afectado un bloque.

Sin embargo, se debería haber modificado el segundo bloque y no el primero por lo que nos hace pensar que el bit modificado es inferior a 128, ya que este es el tamaño de bloque usado por AES para el cifrado.

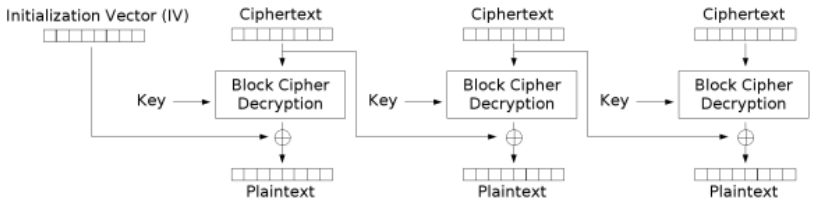
b. Modo CBC

En el modo de operación CBC, al generar los bloques siguientes, se necesita de los bloques anteriores, por lo que, al modificar un solo bloque, se deberían modificar todos los bloque siguientes.

Además, como el tamaño de bloque es de 128 y se ha modificado el bit 32, el primer bloque debería permanecer igual y los siguientes modificados.



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

Comandos:

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-cbc -K 9876543210fedcbaabcdef0123456789 -iv 01234
56789abcdef -in input.bin -out input_aes_256_cbc_con_key_y_iv.bin
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-cbc -K 9876543210fedcbaabcdef0123456789 -iv 01234
56789abcdef -in input1.bin -out input1_aes_256_cbc_con_key_y_iv.bin
pedro@ubuntu:~/Desktop/spsi/p1$ hexedit input_aes_256_cbc_con_key_y_iv.bin
```

Input:

openssl enc -aes-256-cbc -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input.bin -out input_aes_256_cbc_con_key_y_iv.bin

Input 1:

openssl enc -aes-256-cbc -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input1.bin -out input1_aes_256_cbc_con_key_y_iv.bin

Resultados:

Input:

```
00000000   41 D6 05 2B  18 76 20 89  49 A5 AA 15  47 A2 C5 8D  0F 46 BC 9A
00000014   AC 3A 9A BC  CD 45 CB CA  04 5A 89 EE  6C 61 22 04  63 55 27 91
00000028   CC DB 10 D6  C5 2C BD FC  A1 9F 42 F2  DD FF A4 19  8E 64 1C D6
0000003C   6C 28 6B 94  DD 5A 22 0A  52 62 69 35  EA AA 98 2B  CE 67 94 BA
00000050   13 B4 7D 55  D2 7C 72 62  15 C5 8E 18  07 57 D4 CD  54 5C 1B 00
00000064   21 FE CB 16  3C E3 D7 19  2D 46 74 F6  2C 25 C0 8B  F2 05 76 88
00000078   19 1E 8D 14  08 96 A3 D3  E7 9F E1 AB  6F 48 17 94  1D 89 DE 95
0000008C   0A 57 3C 86
```

Input 1:

```
00000000   8A 4A 5A 66  5F DD 35 AB  5F 93 3D 0D  34 36 F4 BB  F0 25 79 68
00000014   2F 94 7E CA  AA CD 2A 79  60 2B 18 8F  A7 53 AA 35  59 F8 57 59
00000028   2E F0 EA 2C  DA CB 27 C2  ED A3 85 35  55 29 13 AB  90 AA 84 AF
0000003C   A2 B3 1E E1  06 5A 7C A7  1A 7B 75 78  AC E3 82 69  F4 DA 6F 71
00000050   40 33 57 53  C3 EF 8C 8C  DB 59 37 E6  A4 F8 90 B2  BE F8 2D F5
00000064   F6 BA AC F8  05 1C CA 7F  07 A6 B2 A7  D3 2B 67 CD  67 D0 EC 30
00000078   BF C8 0A EE  73 1C 1F 1E  F2 41 E9 27  31 64 21 45  F2 99 C4 1A
0000008C   C8 96 BC 0A
```

Diferencias

Input:

```
1. 00000000
   41 D6 05 2B  18 76 20 89  49 A5 AA 15  47 A2 C5 8D  0F 46 BC 9A  A..+.v .I...G....
   F..
2. 00000014   AC 3A 9A BC  CD 45 CB CA  04 5A 89 EE  6C 61 22 04  63 55 27 91
   .:...E...Z..la".cU'.
3. 00000028   CC DB 10 D6  C5 2C BD FC  A1 9F 42 F2  DD FF A4 19  8E 64 1C D6
   .....,....B......d..
4. 0000003C   6C 28 6B 94  DD 5A 22 0A  52 62 69 35  EA AA 98 2B  CE 67 94 BA
   l(k..Z".Rbi5...+.g..
5. 00000050   13 B4 7D 55  D2 7C 72 62  15 C5 8E 18  07 57 D4 CD  54 5C 1B 00
   ..}U.|rb.....W..T\..
6. 00000064   21 FE CB 16  3C E3 D7 19  2D 46 74 F6  2C 25 C0 8B  F2 05 76 88
   !...<...-Ft.,%....v.
7. 00000078   19 1E 8D 14  08 96 A3 D3  E7 9F E1 AB  6F 48 17 94  1D 89 DE 95
   ............oH......
8. 0000008C   0A 57 3C 86
```

Input1:

```
1. 00000000
   8A 4A 5A 66  5F DD 35 AB  5F 93 3D 0D  34 36 F4 BB  F0 25 79 68  .JZf_.5._.=.4
   6...%yh
2. 00000014   2F 94 7E CA  AA CD 2A 79  60 2B 18 8F  A7 53 AA 35  59 F8 57 59
   /.~...*y`+...S.5Y.WY
3. 00000028   2E F0 EA 2C  DA CB 27 C2  ED A3 85 35  55 29 13 AB  90 AA 84 AF
   ...,..'....5U)......
4. 0000003C   A2 B3 1E E1  06 5A 7C A7  1A 7B 75 78  AC E3 82 69  F4 DA 6F 71
   .....Z|..{ux...i..oq
5. 00000050   40 33 57 53  C3 EF 8C 8C  DB 59 37 E6  A4 F8 90 B2  BE F8 2D F5
   @3WS.....Y7.......-.
6. 00000064   F6 BA AC F8  05 1C CA 7F  07 A6 B2 A7  D3 2B 67 CD  67 D0 EC 30
   .............+g.g..0
7. 00000078   BF C8 0A EE  73 1C 1F 1E  F2 41 E9 27  31 64 21 45  F2 99 C4 1A
   ....s....A.'1d!E....
8. 0000008C   C8 96 BC 0A
```

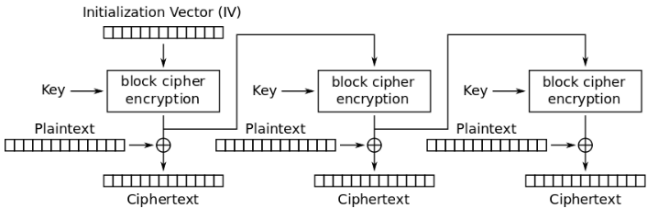
Conclusiones

En este caso, por el tipo de cifrado, se modifica totalmente el archivo pese a que se ha modificado únicamente un bit, ya que para generar los siguientes, coge información de los siguientes, por lo que cuando se modifica un solo bit, todos los bloques siguientes a él, se ven modificados.

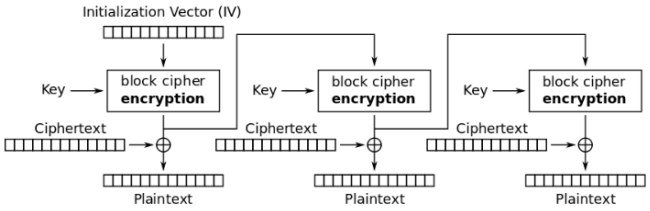
Sin embargo, se puede ver como hay un error, ya que al modificar el bit 132, el primer bloque de 128 debería ser el mismo y no lo es. Esto hace pensar que se ha modificado un bit inferior al 128.

c. Modo OFB

Este modo lo que hace es sumar al mensaje original una secuencia de unos y ceros derivados de los bloques anteriores. El vector de inicialización cambia al cifrar cada uno de lo bloques, por lo que si se modifica un único bit, el resultado debe ser muy parecido al otro mensaje cifrado, ya que sólo variará el bit de la suma.



Output Feedback (OFB) mode encryption



Output Feedback (OFB) mode decryption

Comandos:

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ofb -K 9876543210fedcbaabcdef0123456789 -iv 01234
56789abcdef -in input.bin -out input_aes_256_ofb_con_key_y_iv.bin
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ofb -K 9876543210fedcbaabcdef0123456789 -iv 01234
56789abcdef -in input1.bin -out input1_aes_256_ofb_con_key_y_iv.bin
```

Input:

openssl enc -aes-256-ofb -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input.bin -out input_aes_256_ofb_con_key_y_iv.bin

Input 1:

openssl enc -aes-256-ofb -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input1.bin -out input1_aes_256_ofb_con_key_y_iv.bin

Resultados:

Input:

```
00000000   41 D6 05 2B  18 76 20 89  49 A5 AA 15  47 A2 C5 8D  0F 46 BC 9A
00000014   AC 3A 9A BC  CD 45 CB CA  04 5A 89 EE  6C 61 22 04  63 55 27 91
00000028   CC DB 10 D6  C5 2C BD FC  A1 9F 42 F2  DD FF A4 19  8E 64 1C D6
0000003C   6C 28 6B 94  DD 5A 22 0A  52 62 69 35  EA AA 98 2B  CE 67 94 BA
00000050   13 B4 7D 55  D2 7C 72 62  15 C5 8E 18  07 57 D4 CD  54 5C 1B 00
00000064   21 FE CB 16  3C E3 D7 19  2D 46 74 F6  2C 25 C0 8B  F2 05 76 88
00000078   19 1E 8D 14  08 96 A3 D3
```

Input 1:

```
00000000   41 D6 05 2B  18 76 20 89  46 A5 AA 15  47 A2 C5 8D  0F 46 BC 9A
00000014   AC 3A 9A BC  CD 45 CB CA  04 5A 89 EE  6C 61 22 04  63 55 27 91
00000028   CC DB 10 D6  C5 2C BD FC  A1 9F 42 F2  DD FF A4 19  8E 64 1C D6
0000003C   6C 28 6B 94  DD 5A 22 0A  52 62 69 35  EA AA 98 2B  CE 67 94 BA
00000050   13 B4 7D 55  D2 7C 72 62  15 C5 8E 18  07 57 D4 CD  54 5C 1B 00
00000064   21 FE CB 16  3C E3 D7 19  2D 46 74 F6  2C 25 C0 8B  F2 05 76 88
00000078   19 1E 8D 14  08 96 A3 D3
```

Diferencias

Input:

```
1. 00000000   41 D6 05 2B  18 76 20 89  49
    A5 AA 15  47 A2 C5 8D  0F 46 BC 9A  A..+.v .I...G....F..
```

Input1:

```
1. 00000000   41 D6 05 2B  18 76 20 89  46
    A5 AA 15  47 A2 C5 8D  0F 46 BC 9A  A..+.v .F...G....F..
```

Conclusiones

En este caso, sólo se ha modificado el byte que se ha modificado, quedando los otros tal y como esteban, tal vez, este sea el peor de los tres métodos elegidos, ya que es el que menos propaga los cambios a través del mensaje.

d. Conclusiones

Como se puede ver, al usar un modo u otro de cifrado dentro del cifrado AES, el resultado del archivo cifrado es distinto, siendo el mejor el modo CBC, ya que al modificar un único bit, se modifica el archivo entero y el peor el ECB, ya que cifra bloque a bloque por lo que se pueden sustituir bloques por otros bloque, lo que permitiría modificar quien firma un mensaje si se conocen los bloques correctos, aunque el que peor propaga los cambios es el modo OFB puesto que al modificar un bit sólo se modifica un byte dentro del archivo cifrado por lo que podría ser más fácil de descifrar.

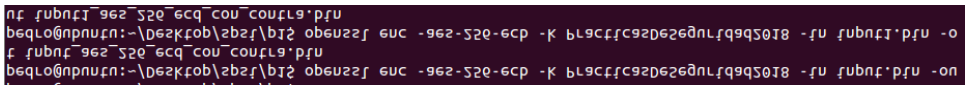
**** Hay otro error en el fichero binario, el bit modificado es el bit 66 y no el 132, por lo que los resultados varían tal y como se han explicado.**

4. AES 256 con contraseña

La contraseña en todos los casos va a ser “PracticasDeSeguridad2018”.

a. Modo ECB

Comandos:



input:

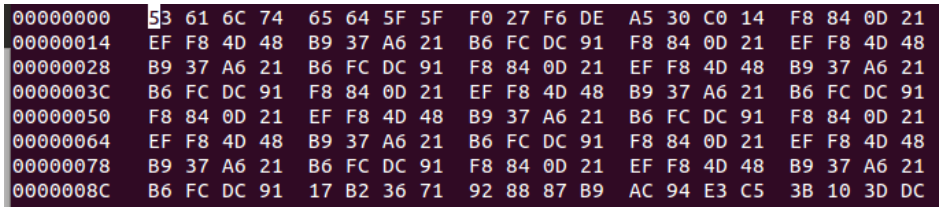
openssl enc -aes-256-ecb -k PracticasDeSeguridad2018 -in input.bin -out input_aes_256_ecd_con_contra.bin

input1:

openssl enc -aes-256-ecb -k PracticasDeSeguridad2018 -in input1.bin -out input1_aes_256_ecd_con_contra.bin

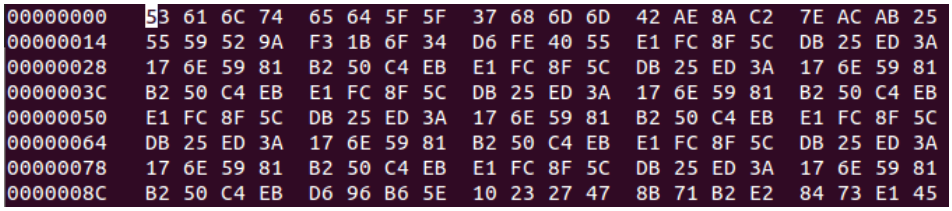
Resultados:

Input:



00000000	53	61	6C	74	65	64	5F	5F	F0	27	F6	DE	A5	30	C0	14	F8	84	0D	21
00000014	EF	F8	4D	48	B9	37	A6	21	B6	FC	DC	91	F8	84	0D	21	EF	F8	4D	48
00000028	B9	37	A6	21	B6	FC	DC	91	F8	84	0D	21	EF	F8	4D	48	B9	37	A6	21
0000003C	B6	FC	DC	91	F8	84	0D	21	EF	F8	4D	48	B9	37	A6	21	B6	FC	DC	91
00000050	F8	84	0D	21	EF	F8	4D	48	B9	37	A6	21	B6	FC	DC	91	F8	84	0D	21
00000064	EF	F8	4D	48	B9	37	A6	21	B6	FC	DC	91	F8	84	0D	21	EF	F8	4D	48
00000078	B9	37	A6	21	B6	FC	DC	91	F8	84	0D	21	EF	F8	4D	48	B9	37	A6	21
0000008C	B6	FC	DC	91	17	B2	36	71	92	88	87	B9	AC	94	E3	C5	3B	10	3D	DC

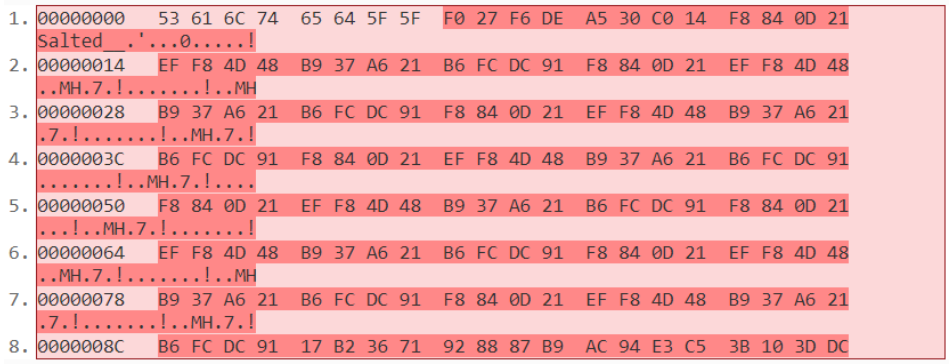
Input1:



00000000	53	61	6C	74	65	64	5F	5F	37	68	6D	6D	42	AE	8A	C2	7E	AC	AB	25
00000014	55	59	52	9A	F3	1B	6F	34	D6	FE	40	55	E1	FC	8F	5C	DB	25	ED	3A
00000028	17	6E	59	81	B2	50	C4	EB	E1	FC	8F	5C	DB	25	ED	3A	17	6E	59	81
0000003C	B2	50	C4	EB	E1	FC	8F	5C	DB	25	ED	3A	17	6E	59	81	B2	50	C4	EB
00000050	E1	FC	8F	5C	DB	25	ED	3A	17	6E	59	81	B2	50	C4	EB	E1	FC	8F	5C
00000064	DB	25	ED	3A	17	6E	59	81	B2	50	C4	EB	E1	FC	8F	5C	DB	25	ED	3A
00000078	17	6E	59	81	B2	50	C4	EB	E1	FC	8F	5C	DB	25	ED	3A	17	6E	59	81
0000008C	B2	50	C4	EB	D6	96	B6	5E	10	23	27	47	8B	71	B2	E2	84	73	E1	45

Diferencias

Input:



```
1. 00000000   53 61 6C 74  65 64 5F 5F  F0 27 F6 DE  A5 30 C0 14  F8 84 0D 21
   Salted__.'...0.....!
2. 00000014   EF F8 4D 48  B9 37 A6 21  B6 FC DC 91  F8 84 0D 21  EF F8 4D 48
   ..MH.7.!.......!..MH
3. 00000028   B9 37 A6 21  B6 FC DC 91  F8 84 0D 21  EF F8 4D 48  B9 37 A6 21
   .7.!.......!..MH.7.!
4. 0000003C   B6 FC DC 91  F8 84 0D 21  EF F8 4D 48  B9 37 A6 21  B6 FC DC 91
   .......!..MH.7.!....
5. 00000050   F8 84 0D 21  EF F8 4D 48  B9 37 A6 21  B6 FC DC 91  F8 84 0D 21
   ...!..MH.7.!.......!
6. 00000064   EF F8 4D 48  B9 37 A6 21  B6 FC DC 91  F8 84 0D 21  EF F8 4D 48
   ..MH.7.!.......!..MH
7. 00000078   B9 37 A6 21  B6 FC DC 91  F8 84 0D 21  EF F8 4D 48  B9 37 A6 21
   .7.!.......!..MH.7.!
8. 0000008C   B6 FC DC 91  17 B2 36 71  92 88 87 B9  AC 94 E3 C5  3B 10 3D DC
```

Input1:

```
1. 00000000   53 61 6C 74  65 64 5F 5F  37 68 6D 6D  42 AE 8A C2  7E AC AB 25
   Salted__7hmmB...~..%
2. 00000014   55 59 52 9A  F3 1B 6F 34  D6 FE 40 55  E1 FC 8F 5C  DB 25 ED 3A
   UYR...o4..@U...\.%.:
3. 00000028   17 6E 59 81  B2 50 C4 EB  E1 FC 8F 5C  DB 25 ED 3A  17 6E 59 81
   .nY..P.....\.%.:.nY.
4. 0000003C   B2 50 C4 EB  E1 FC 8F 5C  DB 25 ED 3A  17 6E 59 81  B2 50 C4 EB
   .P.....\.%.:.nY..P..
5. 00000050   E1 FC 8F 5C  DB 25 ED 3A  17 6E 59 81  B2 50 C4 EB  E1 FC 8F 5C
   ...\.%.:.nY..P.....\
6. 00000064   DB 25 ED 3A  17 6E 59 81  B2 50 C4 EB  E1 FC 8F 5C  DB 25 ED 3A
   .%.:.nY..P.....\.%.:
7. 00000078   17 6E 59 81  B2 50 C4 EB  E1 FC 8F 5C  DB 25 ED 3A  17 6E 59 81
   .nY..P.....\.%.:.nY.
8. 0000008C   B2 50 C4 EB  D6 96 B6 5E  10 23 27 47  8B 71 B2 E2  84 73 E1 45
```

Conclusiones

Como se puede ver, en este caso se modifican todos los bits, esto es debido a que al usar salted, se modifica el vector de inicialización que se obtiene de la contraseña, la cual también se modifica, añadiendo caracteres aleatorios para evitar un ataque por fueza bruta por diccionario.

b. Modo CBC

Comandos:

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-cbc -k PracticasDeSeguridad2018 -in input.bin -ou
t input_aes_256_cbc_con_contra.bin
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-cbc -k PracticasDeSeguridad2018 -in input1.bin -o
ut input1_aes_256_cbc_con_contra.bin
```

Input:

openssl enc -aes-256-cbc -k PracticasDeSeguridad2018 -in input.bin -out input_aes_256_cbc_con_contra.bin

Input 1:

openssl enc -aes-256-cbc -k PracticasDeSeguridad2018 -in input1.bin -out input1_aes_256_cbc_con_contra.bin

Resultados:

Input:

```
00000000   53 61 6C 74  65 64 5F 5F  34 6D E0 A4  75 A0 19 46  20 EC 72 CB
00000014   03 81 0E 25  AA 7B 87 B4  FA DC 7E B9  7B 75 C9 C7  21 F7 45 EF
00000028   57 6A 49 A1  71 3E 2D 91  8F B6 27 48  D8 EA 80 EB  19 B0 60 11
0000003C   68 23 0E D1  A8 10 16 2A  29 69 4D 16  90 32 E4 63  23 AD ED C5
00000050   DA 68 8D CF  D0 27 36 E8  8F 53 7D 6C  75 94 43 44  55 2E 17 7C
00000064   2F 1A EB 60  63 08 B0 02  2A E2 8A 5B  EB E3 D8 E9  FF 04 59 A3
00000078   77 01 C7 1E  C7 BA F5 93  82 F6 30 F0  DF 7C 3A EF  3F F2 9F 6C
0000008C   C0 DE 4D 59  59 47 0B A6  38 85 51 6C  36 D6 1E DC  0F 51 CA C0
```

Input 1:

```
00000000   53 61 6C 74  65 64 5F 5F  41 B7 6A AB  43 54 8C 47  5D B7 76 4B
00000014   BA D4 41 9F  0C F9 D7 88  C9 76 C1 38  CE B1 7A CF  A5 93 3F 7D
00000028   40 6E 6D 41  2C 9D 23 E9  CD D4 4B 4B  5C 39 0F F6  4E 22 7A 24
0000003C   BC 49 81 55  EC 7C 9F AF  21 CF B2 39  3F B2 F6 3A  34 92 9A 23
00000050   87 EB E9 A0  7D E5 BC 6A  5A F0 95 0B  02 9C F9 FD  03 A1 07 74
00000064   E6 E0 25 57  15 39 46 AE  E9 31 78 3B  AD C2 82 24  54 80 46 53
00000078   CF 65 EE 7E  93 3A 4F 8C  16 F3 FF 1D  1E 43 D8 8B  71 9D 1D 37
0000008C   5A DF 48 15  17 BA 7E 05  61 7D DF F1  40 A7 D4 F9  92 02 0C 16
```

Diferencias

Input:

```
00000000   53 61 6C 74  65 64 5F 5F
34 6D E0 A4  75 A0 19 46  20 EC 72 CB  Salted__4m..u..F .r.
00000014   03 81 0E 25  AA 7B 87 B4  FA DC 7E B9  7B 75 C9 C7  21 F7 45 EF
...%.{....~.{u..!.E.
00000028   57 6A 49 A1  71 3E 2D 91  8F B6 27 48  D8 EA 80 EB  19 B0 60 11
WjI.q>-...'H......`.
0000003C   68 23 0E D1  A8 10 16 2A  29 69 4D 16  90 32 E4 63  23 AD ED C5
h#.....*)iM..2.c#...
00000050   DA 68 8D CF  D0 27 36 E8  8F 53 7D 6C  75 94 43 44  55 2E 17 7C
.h...'6..S}lu.CDU..|
00000064   2F 1A EB 60  63 08 B0 02  2A E2 8A 5B  EB E3 D8 E9  FF 04 59 A3
/..`c...*..[......Y.
00000078   77 01 C7 1E  C7 BA F5 93  82 F6 30 F0  DF 7C 3A EF  3F F2 9F 6C
w.........0..|:.?..l
0000008C   C0 DE 4D 59  59 47 0B A6  38 85 51 6C  36 D6 1E DC  0F 51 CA C0
```

Input1:

```
00000000   53 61 6C 74  65 64 5F 5F
41 B7 6A AB  43 54 8C 47  5D B7 76 4B  Salted__A.j.CT.G].vK
00000014   BA D4 41 9F  0C F9 D7 88  C9 76 C1 38  CE B1 7A CF  A5 93 3F 7D
..A......v.8..z...?}
00000028   40 6E 6D 41  2C 9D 23 E9  CD D4 4B 4B  5C 39 0F F6  4E 22 7A 24
@nmA,.#...KK\9..N"z$
0000003C   BC 49 81 55  EC 7C 9F AF  21 CF B2 39  3F B2 F6 3A  34 92 9A 23
.I.U.|..!..9?..:4..#
00000050   87 EB E9 A0  7D E5 BC 6A  5A F0 95 0B  02 9C F9 FD  03 A1 07 74
....}..jZ..........t
00000064   E6 E0 25 57  15 39 46 AE  E9 31 78 3B  AD C2 82 24  54 80 46 53
..%W.9F..1x;...$T.FS
00000078   CF 65 EE 7E  93 3A 4F 8C  16 F3 FF 1D  1E 43 D8 8B  71 9D 1D 37
.e.~.:O......C..q..7
0000008C   5A DF 48 15  17 BA 7E 05  61 7D DF F1  40 A7 D4 F9  92 02 0C 16
```

Conclusiones

Lo mismo que en el caso anterior, por usar salted.

c. Modo OFB

Comandos:

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ofb -k PracticasDeSeguridad2018 -in input.bin -ou
t input_aes_256_ofb_con_contra.bin
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ofb -k PracticasDeSeguridad2018 -in input1.bin -o
```

Input:

openssl enc -aes-256-ofb -k PracticasDeSeguridad2018 -in input.bin -out input_aes_256_ofb_con_contra.bin

Input 1:

openssl enc -aes-256-ofb -k PracticasDeSeguridad2018 -in input1.bin -out input1_aes_256_ofb_con_contra.bin

Resultados:

Input:

```
00000000   53 61 6C 74  65 64 5F 5F  C5 CD A8 0E  92 B7 7D CD  68 6D 34 F0
00000014   CB B2 7F A9  06 9B B8 31  64 31 CC 03  D9 C5 26 1E  9A 35 4E 9A
00000028   7B 4E 05 2A  BF DF 7B 8C  03 75 3F 4C  1B 51 51 89  15 AE 41 0C
0000003C   52 64 56 5B  A6 3F 3D CA  01 89 19 56  AC 4B D3 F2  BD 09 2A 98
00000050   1B 12 68 53  33 0A 63 DA  E2 EC 6E 79  95 09 D1 EC  E8 CB 8A DD
00000064   25 0C 54 E8  B3 EB 82 6C  42 6D A8 5C  75 E1 E8 27  71 64 83 7B
00000078   F6 74 0B C1  C1 E8 77 8A  40 56 BF F9  AF D0 F7 1E  A3 D9 15 1E
0000008C   94 39 EC 51
```

Input 1:

```
00000000   53 61 6C 74  65 64 5F 5F  3A 28 A7 20  8F AB 02 5D  56 27 9C F7
00000014   A3 49 E2 23  BA EC D4 F8  88 EB DB 31  38 DA C7 79  96 E5 EB FD
00000028   61 6C BD 62  E9 AF 01 B1  BF 9C E4 15  AA 4B E7 CB  D3 37 75 E6
0000003C   FA 3A AE BC  A0 2C 66 D8  D0 33 24 EC  85 8F BA 1D  B0 13 EA 92
00000050   79 CF 77 F8  93 1E 92 8E  CC E8 63 C6  93 55 15 48  51 AD 6E 77
00000064   65 78 59 C4  C1 5A 32 17  36 E9 4C 09  4F 3E DE 60  07 7D E5 0E
00000078   AA 8C 15 41  80 F4 D6 47  54 F6 36 4A  DB AA 06 C5  D8 0E F9 B3
0000008C   6C 52 27 DE
```

Diferencias

Input:

```
1. 00000000   53 61 6C 74  65 64 5F 5F
   C5 CD A8 0E  92 B7 7D CD  68 6D 34 F0  Salted__......}.hm4.
2. 00000014   CB B2 7F A9  06 9B B8 31  64 31 CC 03  D9 C5 26 1E  9A 35 4E 9A
   .......1d1....&..5N.
3. 00000028   7B 4E 05 2A  BF DF 7B 8C  03 75 3F 4C  1B 51 51 89  15 AE 41 0C
   {N.*..{..u?L.QQ...A.
4. 0000003C   52 64 56 5B  A6 3F 3D CA  01 89 19 56  AC 4B D3 F2  BD 09 2A 98
   RdV[.?=....V.K....*.
5. 00000050   1B 12 68 53  33 0A 63 DA  E2 EC 6E 79  95 09 D1 EC  E8 CB 8A DD
   ..hS3.c...ny........
6. 00000064   25 0C 54 E8  B3 EB 82 6C  42 6D A8 5C  75 E1 E8 27  71 64 83 7B
   %.T....lBm.\u..'qd.{
7. 00000078   F6 74 0B C1  C1 E8 77 8A  40 56 BF F9  AF D0 F7 1E  A3 D9 15 1E
   .t....w.@V..........
8. 0000008C   94 39 EC 51
```

Input1:

```
1. 00000000   53 61 6C 74  65 64 5F 5F
   3A 28 A7 20  8F AB 02 5D  56 27 9C F7  Salted__:(. ...]V'..
2. 00000014   A3 49 E2 23  BA EC D4 F8  88 EB DB 31  38 DA C7 79  96 E5 EB FD
   .I.#.......18..y....
3. 00000028   61 6C BD 62  E9 AF 01 B1  BF 9C E4 15  AA 4B E7 CB  D3 37 75 E6
   al.b.........K...7u.
4. 0000003C   FA 3A AE BC  A0 2C 66 D8  D0 33 24 EC  85 8F BA 1D  B0 13 EA 92
   .:...,f..3$.........
5. 00000050   79 CF 77 F8  93 1E 92 8E  CC E8 63 C6  93 55 15 48  51 AD 6E 77
   y.w.......c..U.HQ.nw
6. 00000064   65 78 59 C4  C1 5A 32 17  36 E9 4C 09  4F 3E DE 60  07 7D E5 0E
   exY..Z2.6.L.O>.`.}..
7. 00000078   AA 8C 15 41  80 F4 D6 47  54 F6 36 4A  DB AA 06 C5  D8 0E F9 B3
   ...A...GT.6J........
8. 0000008C   6C 52 27 DE
```

Conclusiones

Al igual que en los casos anteriores.

d. Conclusiones

Al usar una contraseña, con la opción salted, se modifican todos los bits ya que el vector de inicialización y la clave usada cambian al añadirle a la contraseña caracteres aleatorios.

5. AES 256 con contraseña -nosalt

La contraseña en todos los casos va a ser “PracticasDeSeguridad2018”.

```
pedro@ubuntu:~/Desktop/spsi/p1$ #Punto 5: Contraseña, nosalt
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ecb -k PracticasDeSeguridad2018 -in input.bin -ou
t input_aes_256_ecd_con_contra_nosalt.bin -nosalt;
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ecb -k PracticasDeSeguridad2018 -in input1.bin -o
ut input1_aes_256_ecd_con_contra_nosalt.bin -nosalt;
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-cbc -k PracticasDeSeguridad2018 -in input.bin -ou
t input_aes_256_cbc_con_contra_nosalt.bin -nosalt;
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-cbc -k PracticasDeSeguridad2018 -in input1.bin -o
ut input1_aes_256_cbc_con_contra_nosalt.bin -nosalt;
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ofb -k PracticasDeSeguridad2018 -in input.bin -ou
t input_aes_256_ofb_con_contra_nosalt.bin -nosalt;
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-256-ofb -k PracticasDeSeguridad2018 -in input1.bin -o
ut input1_aes_256_ofb_con_contra_nosalt.bin -nosalt;
pedro@ubuntu:~/Desktop/spsi/p1$
```

Cuando se usa contraseña, para evitar ataques por fuerza bruta por diccionario, a la contraseña se le añaden caracteres aleatorios, así aunque la contraseña elegida sea débil, al poner más caracteres, se hace más robusta.

Con el la opción -nosalt se evita esto.

***Se ha corregido el error del archivo input1.bin, y se ha modificado el bit 130, por lo que los resultados ahora deben ser correctos.**

a. Modo ECB

Comandos:

openssl enc -aes-256-ecb -k PracticasDeSeguridad2018 -in input.bin -out input_aes_256_ecd_con_contra_nosalt.bin -nosalt

openssl enc -aes-256-ecb -k PracticasDeSeguridad2018 -in input1.bin -out input1_aes_256_ecd_con_contra_nosalt.bin -nosalt

Resultados:

Input:

```
00000000   BB 5B DC 31  40 E9 CD 85  CB E7 29 12  A7 67 C1 82  BB 5B DC 31
00000014   40 E9 CD 85  CB E7 29 12  A7 67 C1 82  BB 5B DC 31  40 E9 CD 85
00000028   CB E7 29 12  A7 67 C1 82  BB 5B DC 31  40 E9 CD 85  CB E7 29 12
0000003C   A7 67 C1 82  BB 5B DC 31  40 E9 CD 85  CB E7 29 12  A7 67 C1 82
00000050   BB 5B DC 31  40 E9 CD 85  CB E7 29 12  A7 67 C1 82  BB 5B DC 31
00000064   40 E9 CD 85  CB E7 29 12  A7 67 C1 82  BB 5B DC 31  40 E9 CD 85
00000078   CB E7 29 12  A7 67 C1 82  40 6B 3C 67  55 C1 1C BA  1A FA B7 03
0000008C   6F 19 65 08
```

Input1:

```
00000000   BB 5B DC 31  40 E9 CD 85  CB E7 29 12  A7 67 C1 82  1E 27 D8 29
00000014   D1 AB B5 48  4B 34 26 74  51 F4 4F 5A  BB 5B DC 31  40 E9 CD 85
00000028   CB E7 29 12  A7 67 C1 82  BB 5B DC 31  40 E9 CD 85  CB E7 29 12
0000003C   A7 67 C1 82  BB 5B DC 31  40 E9 CD 85  CB E7 29 12  A7 67 C1 82
00000050   BB 5B DC 31  40 E9 CD 85  CB E7 29 12  A7 67 C1 82  BB 5B DC 31
00000064   40 E9 CD 85  CB E7 29 12  A7 67 C1 82  BB 5B DC 31  40 E9 CD 85
00000078   CB E7 29 12  A7 67 C1 82  40 6B 3C 67  55 C1 1C BA  1A FA B7 03
0000008C   6F 19 65 08
```

Diferencias

Input:

```
1. 00000000   BB 5B DC 31  40 E9 CD 85  CB E7 29 12  A7 67 C1 82  BB 5B DC 31
   .[.1@.....)..g...[.1
2. 00000014   40 E9 CD 85  CB E7 29 12  A7 67 C1 82  BB 5B DC 31  40 E9 CD 85
   @.....)..g...[.1@...
```

Input1:

```
1. 00000000   BB 5B DC 31  40 E9 CD 85  CB E7 29 12  A7 67 C1 82  1E 27 D8 29
   .[.1@.....)..g...'.)
2. 00000014   D1 AB B5 48  4B 34 26 74  51 F4 4F 5A  BB 5B DC 31  40 E9 CD 85
   ...HK4&tQ.OZ.[.1@...
```

Conclusiones

Como se puede ver en el modo ECB, sólo se modifica el bloque de 128 bits que se ha modificado. Además, como el bit modificado es el 130, el bloque que se modifica es el segundo.

Finalmente, se puede ver que si no se usa la opción de salt, el resultado es similar que si se usa clave con vector de inicialización.

b. Modo CBC

Comandos:

Input:

openssl enc -aes-256-cbc -k PracticasDeSeguridad2018 -in input.bin -out input_aes_256_cbc_con_contra_nosalt.bin -nosalt

Input 1:

openssl enc -aes-256-cbc -k PracticasDeSeguridad2018 -in input1.bin -out input1_aes_256_cbc_con_contra_nosalt.bin -nosalt

Resultados:

Input:

```
00000000   F0 59 A5 6E  65 4A 97 EF  DF 3B E5 45  54 F8 86 18  C8 8A 31 D9
00000014   5C 68 9B D4  7A 6D 4C E4  30 AA 9B BA  70 F1 E0 20  FA CD 2C D0
00000028   34 3C 37 1A  18 50 78 6B  28 87 22 B1  55 06 35 4A  68 2D 1C CD
0000003C   70 FC DB D4  A6 2A C1 E3  68 CD E6 4C  3A EA 6D EC  A9 E1 A8 96
00000050   BA CE 02 6B  10 83 F8 05  CC 3F 4A 8F  E2 A1 3B 2E  44 7F C6 CD
00000064   2F E9 7C 63  4C E5 18 08  D5 0B B6 22  13 DA 35 4A  CC E0 F8 A8
00000078   5A 3D 90 67  F6 35 A6 AB  ED 00 0F DD  C6 41 80 C5  3B 78 36 D7
0000008C   61 72 B0 1C
```

Input 1:

```
00000000   F0 59 A5 6E  65 4A 97 EF  DF 3B E5 45  54 F8 86 18  6B EC 3A 99
00000014   5D 71 C3 2B  AC 4C 26 DA  5F 1B C3 32  92 F0 94 D7  7A CA 1A C7
00000028   F0 94 65 23  CF 45 B9 2B  86 01 6B 35  7D 52 F8 B2  55 83 2C BC
0000003C   8A E6 85 6B  90 36 69 FD  2E 9E DA E1  23 DE 48 F6  96 CF C5 E8
00000050   89 63 42 8C  5A 60 5F 74  64 32 34 90  08 35 0F 85  6D F6 92 9D
00000064   A5 BE E7 87  C7 9E 78 67  DA A1 D9 22  72 9B EF 49  03 D1 19 7C
00000078   9B EB 41 DF  77 04 00 86  68 DD 0A 42  DF BF 17 AD  97 68 12 0B
0000008C   A5 E1 B8 83
```

Diferencias

Input:

```
00000000   F0 59 A5 6E  65 4A 97 EF  DF 3B E5 45  54 F8 86 18  C8 8A 31 D9
.Y.neJ...;.ET.....1.
00000014
5C 68 9B D4  7A 6D 4C E4  30 AA 9B BA  70 F1 E0 20  FA CD 2C D0  \h..zmL.0...p..
 ..,.
00000028   34 3C 37 1A  18 50 78 6B  28 87 22 B1  55 06 35 4A  68 2D 1C CD
4<7..Pxk(.".U.5Jh-..
0000003C   70 FC DB D4  A6 2A C1 E3  68 CD E6 4C  3A EA 6D EC  A9 E1 A8 96
p....*..h..L:.m.....
00000050   BA CE 02 6B  10 83 F8 05  CC 3F 4A 8F  E2 A1 3B 2E  44 7F C6 CD
...k.....?J...;.D...
00000064   2F E9 7C 63  4C E5 18 08  D5 0B B6 22  13 DA 35 4A  CC E0 F8 A8
/.|cL......"..5J....
00000078   5A 3D 90 67  F6 35 A6 AB  ED 00 0F DD  C6 41 80 C5  3B 78 36 D7
Z=.g.5.......A..;x6.
0000008C   61 72 B0 1C
```

Input1:

```
00000000   F0 59 A5 6E  65 4A 97 EF  DF 3B E5 45  54 F8 86 18  6B EC 3A 99
.Y.neJ...;.ET...k.:.
00000014
5D 71 C3 2B  AC 4C 26 DA  5F 1B C3 32  92 F0 94 D7  7A CA 1A C7  ]q.+.L&._..2....
z...
00000028   F0 94 65 23  CF 45 B9 2B  86 01 6B 35  7D 52 F8 B2  55 83 2C BC
..e#.E.+..k5}R..U.,.
0000003C   8A E6 85 6B  90 36 69 FD  2E 9E DA E1  23 DE 48 F6  96 CF C5 E8
...k.6i.....#.H.....
00000050   89 63 42 8C  5A 60 5F 74  64 32 34 90  08 35 0F 85  6D F6 92 9D
.cB.Z`_td24..5..m...
00000064   A5 BE E7 87  C7 9E 78 67  DA A1 D9 22  72 9B EF 49  03 D1 19 7C
......xg..."r..I...|
00000078   9B EB 41 DF  77 04 00 86  68 DD 0A 42  DF BF 17 AD  97 68 12 0B
..A.w...h..B.....h..
0000008C   A5 E1 B8 83
```

Conclusiones

En este caso, como cabría esperar, se modifican todos los bloques a partir del cual se ha modificado el bit, quedando iguale, únicamente el primer bloque de 128 bits.

c. Modo OFB

Comandos:

Input:

openssl enc -aes-256-ofb -k PracticasDeSeguridad2018 -in input.bin -out input_aes_256_ofb_con_contra_nosalt.bin -nosalt

Input 1:

openssl enc -aes-256-ofb -k PracticasDeSeguridad2018 -in input1.bin -out input1_aes_256_ofb_con_contra_nosalt.bin -nosalt

Resultados:

Input:

```
00000000   F0 59 A5 6E  65 4A 97 EF  DF 3B E5 45  54 F8 86 18  C8 8A 31 D9
00000014   5C 68 9B D4  7A 6D 4C E4  30 AA 9B BA  70 F1 E0 20  FA CD 2C D0
00000028   34 3C 37 1A  18 50 78 6B  28 87 22 B1  55 06 35 4A  68 2D 1C CD
0000003C   70 FC DB D4  A6 2A C1 E3  68 CD E6 4C  3A EA 6D EC  A9 E1 A8 96
00000050   BA CE 02 6B  10 83 F8 05  CC 3F 4A 8F  E2 A1 3B 2E  44 7F C6 CD
00000064   2F E9 7C 63  4C E5 18 08  D5 0B B6 22  13 DA 35 4A  CC E0 F8 A8
00000078   5A 3D 90 67  F6 35 A6 AB
```

Input 1:

```
00000000   F0 59 A5 6E  65 4A 97 EF  DF 3B E5 45  54 F8 86 18  CC 8A 31 D9
00000014   5C 68 9B D4  7A 6D 4C E4  30 AA 9B BA  70 F1 E0 20  FA CD 2C D0
00000028   34 3C 37 1A  18 50 78 6B  28 87 22 B1  55 06 35 4A  68 2D 1C CD
0000003C   70 FC DB D4  A6 2A C1 E3  68 CD E6 4C  3A EA 6D EC  A9 E1 A8 96
00000050   BA CE 02 6B  10 83 F8 05  CC 3F 4A 8F  E2 A1 3B 2E  44 7F C6 CD
00000064   2F E9 7C 63  4C E5 18 08  D5 0B B6 22  13 DA 35 4A  CC E0 F8 A8
00000078   5A 3D 90 67  F6 35 A6 AB
```

Diferencias

Input:

```
1. 00000000   F0 59 A5 6E  65 4A 97 EF  DF 3B E5 45  54 F8 86 18  C8
    8A 31 D9  .Y.neJ...;.ET.....1.
```

Input1:

```
1. 00000000   F0 59 A5 6E  65 4A 97 EF  DF 3B E5 45  54 F8 86 18  CC
    8A 31 D9  .Y.neJ...;.ET.....1.
```

Conclusiones

De nuevo, puede verse cómo sólo se modifica el bit que se ha modificado, quedando el resto del mensaje igual.

d. Conclusiones

En esta sección se puede ver como si se usa contraseña con la opción -nosalt, los resultados son similares a si se usa clave y vector de inicialización.

Por otro lado, puede verse como realmente se ha corregido el error del bit.

6. AES 192 en modo OFB con vector de inicialización y clave

El vector de inicialización se mantiene fijo a "**0123456789abcdef**".
La clave, tiene que ser de 192 bits por, lo que es equivalente a 24 caracteres en hexadecimal, por lo que se va a elegir la siguiente contraseña en todos los casos "**9876543210fedcbaabcdef01**".

Comando:

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-192-ofb -K 9876543210fedcbaabcdef01 -iv 0123456789abc
def -in input.bin -out output.bin
```

openssl enc -aes-192-ofb -K 9876543210fedcbaabcdef01 -iv 0123456789abcdef -in input.bin -out output.bin

7. Descifrar output.bin

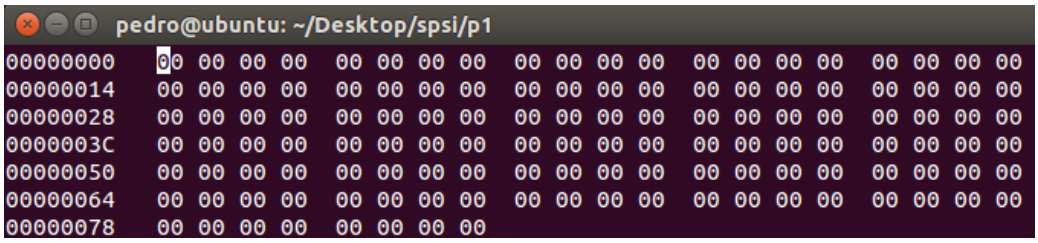
Para descifrar se usa la opción -d

Comandos

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-192-ofb -K 9876543210fedcbaabcdef01 -iv 0123456789abc
def -in output.bin -out output_desnc.bin -d
```

openssl enc -aes-192-ofb -K 9876543210fedcbaabcdef01 -iv 0123456789abcdef -in output.bin -out output_desnc.bin -d

Resultado:



Como se puede ver, el archivo queda tal y como estaba antes de cifrarse

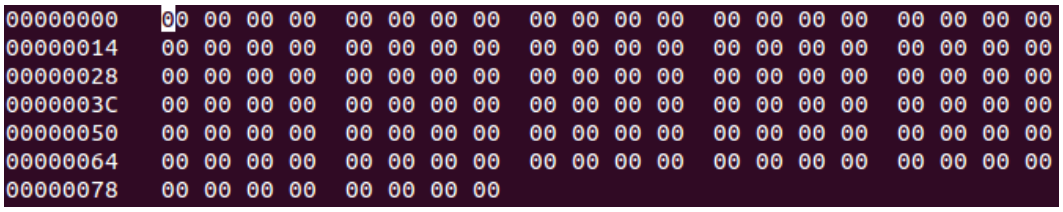
8. Cifrado por segunda vez de output.bin

Comandos

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-192-ofb -K 9876543210fedcbaabcdef01 -iv 0123456789abc
def -in output.bin -out output2.bin
```

openssl enc -aes-192-ofb -K 9876543210fedcbaabcdef01 -iv 0123456789abcdef -in output.bin -out output2.bin

Resultado



Conclusiones

Se vuelve a obtener el mismo archivo que cuando se cifró, esto es debido a que para descifrar se llevan a cabo las mismas operaciones que para cifrar, es decir, se suma con un XOR, por lo por lo que el cifrado del cifrado es el archivo descifrado, al igual que la suma de la suma en binario, deja el número como estaba.

9. AES 192 en modo OFB con contraseña

La contraseña es “PracticasDeSeguridad2018”.

Comandos

```
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-192-ofb -k PracticasDeSeguridad2018 -in input.bin -ou
t output_contra.bin
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-192-ofb -k PracticasDeSeguridad2018 -in output_contra
.bin -out output_contra_desc.bin -d
pedro@ubuntu:~/Desktop/spsi/p1$ openssl enc -aes-192-ofb -k PracticasDeSeguridad2018 -in output_contra
.bin -out output_contra2.bin
```

openssl enc -aes-192-ofb -k PracticasDeSeguridad2018 -in input.bin -out output_contra.bin

openssl enc -aes-192-ofb -k PracticasDeSeguridad2018 -in output_contra.bin -out output_contra_desc.bin -d

openssl enc -aes-192-ofb -k PracticasDeSeguridad2018 -in output_contra.bin -out output_contra2.bin

Resultados:

Archivo cifrado:

```
00000000  53 61 6C 74  65 64 5F 5F  85 A7 87 86  81 5B CF 3F  E0 9C 05 20
00000014  42 81 E3 1D  50 3A 77 67  81 8F 56 0E  86 2D A6 28  5B 17 7F 5C
00000028  60 C7 8E 72  ED 8A C1 08  20 6F D0 FE  B7 94 7B C0  5D AD A7 ED
0000003C  5F A0 5B C1  31 3D 8F F5  E8 64 7E 6A  0C EC 3A DA  FF 04 AE C1
00000050  BD 29 97 9A  88 57 A3 63  4F 45 C0 92  20 22 35 7D  F6 17 6C 8C
00000064  33 76 92 CB  92 10 04 36  AB 23 6E AA  98 62 3F 87  F2 27 F1 A0
00000078  60 EB A9 99  E0 CF E8 2F  B6 32 62 CF  2A 9B 6B 09  FB 33 32 69
0000008C  D2 73 30 33
```

Archivo descifrado:

```
00000000  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000014  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000028  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000003C  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000050  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000064  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000078  00 00 00 00  00 00 00 00
```

Archivo cifrado por segunda vez:

```
00000000  53 61 6C 74  65 64 5F 5F  B4 16 4A BA  15 21 9E F7  2A AC 42 C3
00000014  E3 EB 81 B7  0E D5 F4 92  03 06 2E 29  93 4C FE F4  94 8A 5F 7B
00000028  4F 03 A1 10  4A AD F0 03  2E 78 9B 73  0B 52 E1 A3  F3 52 F2 4E
0000003C  63 97 C9 80  9D 80 49 62  A6 93 46 E4  90 B8 92 58  D5 F7 2F 4E
00000050  8F B2 78 28  F7 B2 62 AC  E1 4D B0 73  CB 72 3B 38  AA 14 C5 29
00000064  96 59 C7 A4  DC AE F3 D5  3F 8A E2 49  0C 39 06 CB  88 D9 E7 4D
00000078  C4 92 80 D6  F6 C1 7B C0  F5 C5 2E CF  A2 CC B0 C7  E6 D3 B3 95
0000008C  3B C9 46 72  39 A2 AA 83  D1 89 E2 BD  70 FA AF 5B  1E A5 18 AC
```

Conclusiones

En este caso, tal y como se puede ver, al cifrar por segunda vez el resultado no es el archivo original como cabría esperar, esto es debido a los caracteres que se añaden cuando se usa el cifrado con contraseña para evitar los ataques por fuerza bruta con diccionario y a que el vector de inicialización también cambia, ya que se extrae de la contraseña.

Si se quiere evitar esto, se puede usar la opción -nosalt y el resultado sí que sería el esperado, se descifraría el mensaje al igual que ocurre cuando se usa clave y vector de inicialización.

*Ver archivos adjuntos.

10. Cifrado Camellia

Es un tipo de cifrado muy parecido a AES, ambos usan cifrado simétrico, con tamaño de bloque de cifrado es de 128 bits, con un tamaño de clave a elegir entre 128, 192 o 256 bits.

Al igual que AES el número de rondas depende de del tamaño de la clave, para una clave de 128bits se usan 18 rondas mientras que si la clave es de 192 o 256 se usan 24 rondas.

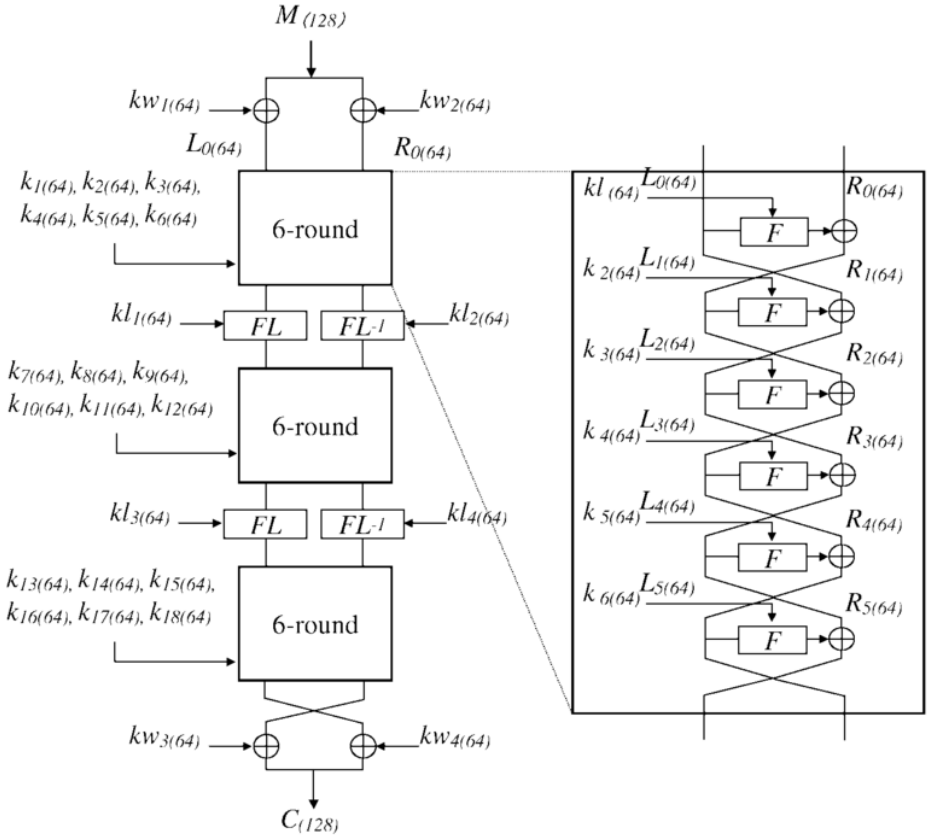


Imagen 1: Cifrado de bloque del algoritmo de cifrado Camilla para una clave de 128bits

11. Resultados con Camellia

Los resultados deben salir idénticos a AES pues lo único que cambia es el número de rondas que se hacen.

a. Clave y vector de inicialización

1) ECB

Comandos

Input:

openssl enc -camellia-256-ecb -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input.bin -out input_camellia_256_ecb_con_key_y_iv.bin;

Input1:

openssl enc -camellia-256-ecb -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input1.bin -out input1_camellia_256_ecb_con_key_y_iv.bin;

Resultados

Input:

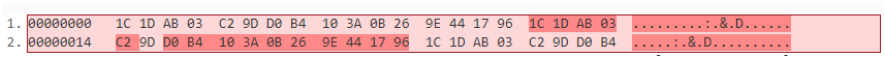
```
1C 1D AB 03  C2 9D D0 B4  10 3A 0B 26  9E 44 17 96  1C 1D AB 03
C2 9D D0 B4  10 3A 0B 26  9E 44 17 96  1C 1D AB 03  C2 9D D0 B4
10 3A 0B 26  9E 44 17 96  1C 1D AB 03  C2 9D D0 B4  10 3A 0B 26
9E 44 17 96  1C 1D AB 03  C2 9D D0 B4  10 3A 0B 26  9E 44 17 96
1C 1D AB 03  C2 9D D0 B4  10 3A 0B 26  9E 44 17 96  1C 1D AB 03
C2 9D D0 B4  10 3A 0B 26  9E 44 17 96  1C 1D AB 03  C2 9D D0 B4
10 3A 0B 26  9E 44 17 96  D5 3A 42 02  98 15 C4 96  37 71 92 FF
23 F5 2C 27
```

Input1:

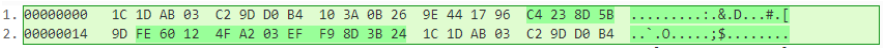
```
1C 1D AB 03  C2 9D D0 B4  10 3A 0B 26  9E 44 17 96  C4 23 8D 5B
9D FE 60 12  4F A2 03 EF  F9 8D 3B 24  1C 1D AB 03  C2 9D D0 B4
10 3A 0B 26  9E 44 17 96  1C 1D AB 03  C2 9D D0 B4  10 3A 0B 26
9E 44 17 96  1C 1D AB 03  C2 9D D0 B4  10 3A 0B 26  9E 44 17 96
1C 1D AB 03  C2 9D D0 B4  10 3A 0B 26  9E 44 17 96  1C 1D AB 03
C2 9D D0 B4  10 3A 0B 26  9E 44 17 96  1C 1D AB 03  C2 9D D0 B4
10 3A 0B 26  9E 44 17 96  D5 3A 42 02  98 15 C4 96  37 71 92 FF
23 F5 2C 27
```

Diferencias

Input:



Input1:



Conclusiones

Como era esperable, el resultado es idéntico a AES. Sólo cambia el bloque de 128 bits donde se ha modificado el bit del mensaje.

En este caso y a diferencia del resultado anterior, se puede ver que el bit elegido está en una posición correcta.

2) CBC

Comandos

Input:

openssl enc -camellia-256-cbc -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input.bin -out input_camellia_256_cbc_con_key_y_iv.bin;

Input1:

openssl enc -camellia-256-cbc -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input1.bin -out input1_camellia_256_cbc_con_key_y_iv.bin;

Resultados

Input:

```
03 4F 69 F0  07 DC CE 1C  32 CE 70 DE  7D B5 C1 34  65 02 5D 06
D2 63 CA E6  68 DC 32 E7  80 B6 38 8A  EC 87 56 48  7C 25 8B 41
6F 0D 8B BB  48 F7 0F 91  3C 24 D5 BA  56 80 24 F3  E0 FA 09 DA
B3 BB 71 52  6E 28 75 7F  86 B3 CE 77  07 16 DB 3F  02 32 A5 90
98 BC 40 18  3A 94 D6 45  2A 23 67 D3  BA 50 2F 27  10 71 FE 57
4E 5C E3 AF  0A B4 21 66  7C 65 F4 BA  27 E3 F6 B2  6C FD 2F EE
34 11 EE 6D  2B 19 B5 45  BD 4C FD 31  36 96 B6 6A  30 8A EF 16
23 23 2C 5B
```

Input1:

```
03 4F 69 F0  07 DC CE 1C  32 CE 70 DE  7D B5 C1 34  E2 BC 0A E1
E9 E2 31 D7  1F EB 60 39  D7 E0 DA E6  3B 36 6A 68  65 24 3C 7B
54 26 78 F3  FD 80 8A 27  68 DF 67 11  FC D8 3E 8B  AC 6A D4 F9
6E 27 5B DD  55 B1 63 2A  ED 8E D5 D8  EC 12 8C 7C  47 87 A0 25
A0 21 C1 48  4B 4E F0 82  9E 58 0D EF  BF 56 FF 1E  74 2E C0 7D
07 83 C6 57  6E 11 ED F3  10 8C 80 D0  0C 09 DF 19  38 AE 7B 98
D9 D2 7B 73  5A DC 8D 3F  9A 3E DE 77  47 EA E4 9E  F3 88 FE F0
6C 48 81 B0
```

Diferencias

Input:

```
00000000  03 4F 69 F0  07 DC CE 1C  32 CE 70 DE  7D B5 C1 34  65 02 5D 06  .Oi.....2.p.}..4e.].
00000014  D2 63 CA E6  68 DC 32 E7  80 B6 38 8A  EC 87 56 48  7C 25 8B 41  .c..h.2...8...VH|%.A
00000028  6F 0D 8B BB  48 F7 0F 91  3C 24 D5 BA  56 80 24 F3  E0 FA 09 DA  o...H...<$..V.$.....
0000003C  B3 BB 71 52  6E 28 75 7F  86 B3 CE 77  07 16 DB 3F  02 32 A5 90  ..qRn(u....w...?.2..
00000050  98 BC 40 18  3A 94 D6 45  2A 23 67 D3  BA 50 2F 27  10 71 FE 57  ..@.:..E*#g..P/'.q.W
00000064  4E 5C E3 AF  0A B4 21 66  7C 65 F4 BA  27 E3 F6 B2  6C FD 2F EE  N\....!f|e..'...l./.
00000078  34 11 EE 6D  2B 19 B5 45  BD 4C FD 31  36 96 B6 6A  30 8A EF 16  4..m+..E.L.16..j0...
0000008C  23 23 2C 5B
```

Input1:

```
00000000  03 4F 69 F0  07 DC CE 1C  32 CE 70 DE  7D B5 C1 34  E2 BC 0A E1  .Oi.....2.p.}..4....
00000014  E9 E2 31 D7  1F EB 60 39  D7 E0 DA E6  3B 36 6A 68  65 24 3C 7B  ..1...`9....;6jhe$<{
00000028  54 26 78 F3  FD 80 8A 27  68 DF 67 11  FC D8 3E 8B  AC 6A D4 F9  T&x....'h.g...>..j..
0000003C  6E 27 5B DD  55 B1 63 2A  ED 8E D5 D8  EC 12 8C 7C  47 87 A0 25  n'[.U.c*.......|G..%
00000050  A0 21 C1 48  4B 4E F0 82  9E 58 0D EF  BF 56 FF 1E  74 2E C0 7D  .!.HKN...X...V..t..}
00000064  07 83 C6 57  6E 11 ED F3  10 8C 80 D0  0C 09 DF 19  38 AE 7B 98  ...Wn...........8.{.
00000078  D9 D2 7B 73  5A DC 8D 3F  9A 3E DE 77  47 EA E4 9E  F3 88 FE F0  ..{sZ..?.>.wG.......
0000008C  6C 48 81 B0
```

Conclusiones

Como se puede ver, se modifican todos los bloques siguientes a partir del bloque donde se modifica el bit tal y como lo haría AES.

3) OFB

Comandos

Input:

openssl enc -camellia-256-ofb -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input.bin -out input_camellia_256_ofb_con_key_y_iv.bin;

Input1:

openssl enc -camellia-256-ofb -K 9876543210fedcbaabcdef0123456789 -iv 0123456789abcdef -in input1.bin -out input1_camellia_256_ofb_con_key_y_iv.bin;

Resultados

Input:

```
03 4F 69 F0  07 DC CE 1C  32 CE 70 DE  7D B5 C1 34  65 02 5D 06
D2 63 CA E6  68 DC 32 E7  80 B6 38 8A  EC 87 56 48  7C 25 8B 41
6F 0D 8B BB  48 F7 0F 91  3C 24 D5 BA  56 80 24 F3  E0 FA 09 DA
B3 BB 71 52  6E 28 75 7F  86 B3 CE 77  07 16 DB 3F  02 32 A5 90
98 BC 40 18  3A 94 D6 45  2A 23 67 D3  BA 50 2F 27  10 71 FE 57
4E 5C E3 AF  0A B4 21 66  7C 65 F4 BA  27 E3 F6 B2  6C FD 2F EE
34 11 EE 6D  2B 19 B5 45
```

Input1:

```
03 4F 69 F0  07 DC CE 1C  32 CE 70 DE  7D B5 C1 34  61 02 5D 06
D2 63 CA E6  68 DC 32 E7  80 B6 38 8A  EC 87 56 48  7C 25 8B 41
6F 0D 8B BB  48 F7 0F 91  3C 24 D5 BA  56 80 24 F3  E0 FA 09 DA
B3 BB 71 52  6E 28 75 7F  86 B3 CE 77  07 16 DB 3F  02 32 A5 90
98 BC 40 18  3A 94 D6 45  2A 23 67 D3  BA 50 2F 27  10 71 FE 57
4E 5C E3 AF  0A B4 21 66  7C 65 F4 BA  27 E3 F6 B2  6C FD 2F EE
34 11 EE 6D  2B 19 B5 45
```

Diferencias

Input:

```
1. 00000000  03 4F 69 F0  07 DC CE 1C  32 CE 70 DE  7D B5 C1 34  65 02 5D 06  .Oi.....2.p.}..4e.].
```

Input1:

```
1. 00000000  03 4F 69 F0  07 DC CE 1C  32 CE 70 DE  7D B5 C1 34  61 02 5D 06  .Oi.....2.p.}..4a.].
```

Conclusiones

De nuevo, sólo se modifica el bit que se ha modificado.

4) Conclusiones

Como era de esperar, Camellia se comporta como AES. Sin embargo, en esta parte hemos podido ver cómo hubieran salido los resultados de AES si se hubiese modificado el bit correcto.

b. Contraseña con salt

1) ECB

Comandos

Input:

openssl enc -camellia-256-ecb -k PracticasDeSeguridad2018 -in input.bin -out input_camellia_256_ecd_con_contra.bin;

Input1:

openssl enc -camellia-256-ecb -k PracticasDeSeguridad2018 -in input1.bin -out input1_camellia_256_ecd_con_contra.bin;

Resultados

Input:

```
53 61 6C 74  65 64 5F 5F  2E EE 89 4E  B5 E1 83 A4  D9 AA 9A 4C  Salted__...N.......L
DB 79 D0 23  DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  .y.#. .._T.....L.y.#
DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  DD 20 BF FE  . .._T.....L.y.#. ..
5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  DD 20 BF FE  5F 54 B8 D0  _T.....L.y.#. .._T..
D9 AA 9A 4C  DB 79 D0 23  DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  ...L.y.#. .._T.....L
DB 79 D0 23  DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  .y.#. .._T.....L.y.#
DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  DD 20 BF FE  . .._T.....L.y.#. ..
5F 54 B8 D0  C9 1A 14 CD  FF D9 1D E3  FD B1 31 AB  41 92 53 2E  _T...........1.A.S.
```

Input1:

```
53 61 6C 74  65 64 5F 5F  F7 31 78 FE  7D AF 0A DC  44 21 53 67  Salted__.1x.}...D!Sg
A0 DC F1 CE  18 A3 85 D4  DE 3E 45 A5  2B 69 A2 B2  43 AF 9F 9B  .........>E.+i..C...
ED 98 1A 9E  F4 B8 15 BE  44 21 53 67  A0 DC F1 CE  18 A3 85 D4  ........D!Sg........
DE 3E 45 A5  44 21 53 67  A0 DC F1 CE  18 A3 85 D4  DE 3E 45 A5  .>E.D!Sg.........>E.
44 21 53 67  A0 DC F1 CE  18 A3 85 D4  DE 3E 45 A5  44 21 53 67  D!Sg.........>E.D!Sg
A0 DC F1 CE  18 A3 85 D4  DE 3E 45 A5  44 21 53 67  A0 DC F1 CE  .........>E.D!Sg....
18 A3 85 D4  DE 3E 45 A5  44 21 53 67  A0 DC F1 CE  18 A3 85 D4  .....>E.D!Sg........
DE 3E 45 A5  33 40 0B 18  16 1D 53 5F  DB 3A D2 AB  51 34 82 0B  .>E.3@....S_.:..Q4..
```

Diferencias

Input:

```
00000000  53 61 6C 74  65 64 5F 5F  2E EE 89 4E  B5 E1 83 A4  D9 AA 9A 4C  Salted__...N.......L
00000014  DB 79 D0 23  DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  .y.#. .._T.....L.y.#
00000028  DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  DD 20 BF FE  . .._T.....L.y.#. ..
0000003C  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  DD 20 BF FE  5F 54 B8 D0  _T.....L.y.#. .._T..
00000050  D9 AA 9A 4C  DB 79 D0 23  DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  ...L.y.#. .._T.....L
00000064  DB 79 D0 23  DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  .y.#. .._T.....L.y.#
00000078  DD 20 BF FE  5F 54 B8 D0  D9 AA 9A 4C  DB 79 D0 23  DD 20 BF FE  . .._T.....L.y.#. ..
0000008C  5F 54 B8 D0  C9 1A 14 CD  FF D9 1D E3  FD B1 31 AB  41 92 53 2E
```

Input1:

```
00000000  53 61 6C 74  65 64 5F 5F  F7 31 78 FE  7D AF 0A DC  44 21 53 67  Salted__.1x.}...D!Sg
00000014  A0 DC F1 CE  18 A3 85 D4  DE 3E 45 A5  2B 69 A2 B2  43 AF 9F 9B  .........>E.+i..C...
00000028  ED 98 1A 9E  F4 B8 15 BE  44 21 53 67  A0 DC F1 CE  18 A3 85 D4  ........D!Sg........
0000003C  DE 3E 45 A5  44 21 53 67  A0 DC F1 CE  18 A3 85 D4  DE 3E 45 A5  .>E.D!Sg.........>E.
00000050  44 21 53 67  A0 DC F1 CE  18 A3 85 D4  DE 3E 45 A5  44 21 53 67  D!Sg.........>E.D!Sg
00000064  A0 DC F1 CE  18 A3 85 D4  DE 3E 45 A5  44 21 53 67  A0 DC F1 CE  .........>E.D!Sg....
00000078  18 A3 85 D4  DE 3E 45 A5  44 21 53 67  A0 DC F1 CE  18 A3 85 D4  .....>E.D!Sg........
0000008C  DE 3E 45 A5  33 40 0B 18  16 1D 53 5F  DB 3A D2 AB  51 34 82 0B
```

Conclusiones

Como se puede ver, el cifrado cambia totalmente por culpa del salt, lo único que no cambia es la parte que doce que se unas salt, es decir, los primeros 64 bits.

Esto sucedía igual en AES

2) CBC

Comandos

Input:

openssl enc -camellia-256-cbc -k PracticasDeSeguridad2018 -in input.bin -out input_camellia_256_cbc_con_contra.bin;

Input1:

openssl enc -camellia-256-cbc -k PracticasDeSeguridad2018 -in input1.bin -out input1_camellia_256_cbc_con_contra.bin;

Resultados

Input:

```
53 61 6C 74  65 64 5F 5F  FD 00 DD 8B  BC 99 5D B9  DB A1 A5 B6  Salted__......].....
9F 26 F3 F8  30 68 7E 0B  1B 45 60 23  5B 89 F4 0F  EA 5D F6 C5  .&..0h~..E`#[....]..
52 E1 C1 34  F2 7E 26 72  B4 19 0D 2C  6C 18 FF 59  BA B2 C5 1A  R..4.~&r...,l..Y....
20 2C 74 3E  10 5B 18 62  6F 3F C9 DA  64 64 05 31  9F 50 EC 20   ,t>.[.bo?..dd.1.P.
71 34 53 A4  4F F2 EC 04  F7 17 7C 4F  49 7C 9C 62  54 E2 85 53  q4S.O.....|OI|.bT..S
B6 7B 13 B1  6D 14 9D ED  0D F8 38 86  11 9F 06 7C  4A 4E 0A DA  .{..m.....8....|JN..
F0 AC C8 E7  FE 12 42 85  75 3E A3 C8  0C 03 8D 2B  45 39 6E B4  ......B.u>.....+E9n.
7D A0 49 E2  B1 0E 6C 8B  13 CF F1 EC  42 1A 95 BA  9B 05 0B 56  }.I...l.....B......V
```

Input1:

```
53 61 6C 74  65 64 5F 5F  E7 D5 D2 D8  11 2B 07 2B  1E CE 27 E4  Salted__.....+.+..'.
7D F9 A0 D6  EE 05 D2 D9  E3 72 D2 78  6F 9E BA ED  2C 17 57 06  }........r.xo...,.W.
37 71 BF 20  09 6C 53 6E  8A 67 CC D9  57 E6 83 31  B6 FB D4 D4  7q. .lSn.g..W..1....
32 94 B1 BD  CA 32 4A 95  44 A2 41 22  44 38 1C 95  6B 42 1D 8C  2....2J.D.A"D8..kB..
B3 25 C8 05  70 BB EE 40  FD 72 D8 6F  18 86 0B AC  DC B3 EE 2D  .%..p..@.r.o.......-
86 FE AD 36  7B D2 61 27  74 99 6C 71  40 6D C0 8D  C2 50 29 21  ...6{.a't.lq@m...P)!
0D 7D 48 F7  B7 A5 C1 66  DD BE 00 76  8A AA A2 8C  4A 8B E8 4E  .}H....f...v....J..N
8E 40 D6 14  19 D4 A4 8B  28 85 32 94  84 77 EC 93  20 25 78 EC  .@......(.2..w.. %x.
```

Diferencias

Input:

```
00000000  53 61 6C 74  65 64 5F 5F  FD 00 DD 8B  BC 99 5D B9  DB A1 A5 B6  Salted__......].....
00000014  9F 26 F3 F8  30 68 7E 0B  1B 45 60 23  5B 89 F4 0F  EA 5D F6 C5  .&..0h~..E`#[....]..
00000028  52 E1 C1 34  F2 7E 26 72  B4 19 0D 2C  6C 18 FF 59  BA B2 C5 1A  R..4.~&r...,l..Y....
0000003C  20 2C 74 3E  10 5B 18 62  6F 3F C9 DA  64 64 05 31  9F 50 EC 20   ,t>.[.bo?..dd.1.P.
00000050  71 34 53 A4  4F F2 EC 04  F7 17 7C 4F  49 7C 9C 62  54 E2 85 53  q4S.O.....|OI|.bT..S
00000064  B6 7B 13 B1  6D 14 9D ED  0D F8 38 86  11 9F 06 7C  4A 4E 0A DA  .{..m.....8....|JN..
00000078  F0 AC C8 E7  FE 12 42 85  75 3E A3 C8  0C 03 8D 2B  45 39 6E B4  ......B.u>.....+E9n.
0000008C  7D A0 49 E2  B1 0E 6C 8B  13 CF F1 EC  42 1A 95 BA  9B 05 0B 56
```

Input1:

```
00000000  53 61 6C 74  65 64 5F 5F  E7 D5 D2 D8  11 2B 07 2B  1E CE 27 E4  Salted__.....+.+..'.
00000014  7D F9 A0 D6  EE 05 D2 D9  E3 72 D2 78  6F 9E BA ED  2C 17 57 06  }........r.xo...,.W.
00000028  37 71 BF 20  09 6C 53 6E  8A 67 CC D9  57 E6 83 31  B6 FB D4 D4  7q. .lSn.g..W..1....
0000003C  32 94 B1 BD  CA 32 4A 95  44 A2 41 22  44 38 1C 95  6B 42 1D 8C  2....2J.D.A"D8..kB..
00000050  B3 25 C8 05  70 BB EE 40  FD 72 D8 6F  18 86 0B AC  DC B3 EE 2D  .%..p..@.r.o.......-
00000064  86 FE AD 36  7B D2 61 27  74 99 6C 71  40 6D C0 8D  C2 50 29 21  ...6{.a't.lq@m...P)!
00000078  0D 7D 48 F7  B7 A5 C1 66  DD BE 00 76  8A AA A2 8C  4A 8B E8 4E  .}H....f...v....J..N
0000008C  8E 40 D6 14  19 D4 A4 8B  28 85 32 94  84 77 EC 93  20 25 78 EC
```

Conclusiones

Igual que el caso anterior

3) OFB

Comandos

Input:

openssl enc -camellia-256-ofb -k PracticasDeSeguridad2018 -in input.bin -out input_camellia_256_ofb_con_contra.bin;

Input1:

openssl enc -camellia-256-ofb -k PracticasDeSeguridad2018 -in input1.bin -out input1_camellia_256_ofb_con_contra.bin;

Resultados

Input:

```
53 61 6C 74  65 64 5F 5F  F3 7C 4F 54  94 84 1A 07  8A 5C F3 B6  Salted__.|OT.....\..
04 DB B3 B1  90 53 26 D0  1D 82 83 3B  F9 0C 7D AD  6D B7 C2 EE  .....S&....;..}.m...
35 25 18 0D  93 F4 F9 EA  66 A6 58 56  42 E9 B9 CC  0D 64 E9 10  5%......f.XVB....d..
C4 EB B1 8B  69 4C B4 33  56 B4 EB E3  76 8F 7A C5  3C 91 68 AE  ....iL.3V...v.z.<.h.
C8 CC D7 19  0C 5B 60 71  A5 41 CD 67  9B A3 D1 EC  BB 77 EB 39  .....[`q.A.g.....w.9
5D 8D 33 49  01 29 09 C3  50 34 80 CB  5A 5F 8A 29  E2 7B 51 C3  ].3I.)..P4..Z_.).{Q.
01 25 B9 E7  B8 27 5A 59  77 D3 7B B3  B7 D5 94 AE  B6 14 E9 AF  .%...'ZYw.{.........
E3 6B DA D0                                                      .k..
```

Input1:

```
53 61 6C 74  65 64 5F 5F  59 E1 ED 02  4B 59 7B F5  69 10 F7 99  Salted__Y...KY{.i...
EB 93 7A 6E  A4 3B 89 E3  D1 C5 CF 25  9D FB 72 90  05 D2 83 7B  ..zn.;.....%..r....{
C4 9A B7 FC  15 F9 73 87  AE C7 25 F9  7D 72 3F 85  EF B9 55 04  ......s...%.}r?...U.
12 03 3E 03  0B 44 D0 20  ED CB 16 38  0C 43 F2 4D  51 20 4C AB  ..>..D. ...8.C.MQ L.
07 5E 0E 51  39 07 FD AF  9E B9 85 26  7C 4E F8 FE  45 89 27 EE  .^.Q9......&|N..E.'.
99 1E EA 21  46 3D 89 C4  1D E2 75 19  16 A4 B5 7B  D6 08 D7 CC  ...!F=....u....{....
04 92 D4 B3  CE 9E 89 8A  BE CB 1F FC  6B BC 33 0E  94 E0 78 DD  ............k.3...x.
98 78 1E 85                                                      .x..
```

Diferencias

Input:

```
00000000  53 61 6C 74  65 64 5F 5F  F3 7C 4F 54  94 84 1A 07  8A 5C F3 B6  Salted__.|OT.....\..
00000014  04 DB B3 B1  90 53 26 D0  1D 82 83 3B  F9 0C 7D AD  6D B7 C2 EE  .....S&....;..}.m...
00000028  35 25 18 0D  93 F4 F9 EA  66 A6 58 56  42 E9 B9 CC  0D 64 E9 10  5%......f.XVB....d..
0000003C  C4 EB B1 8B  69 4C B4 33  56 B4 EB E3  76 8F 7A C5  3C 91 68 AE  ....iL.3V...v.z.<.h.
00000050  C8 CC D7 19  0C 5B 60 71  A5 41 CD 67  9B A3 D1 EC  BB 77 EB 39  .....[`q.A.g.....w.9
00000064  5D 8D 33 49  01 29 09 C3  50 34 80 CB  5A 5F 8A 29  E2 7B 51 C3  ].3I.)..P4..Z_.).{Q.
00000078  01 25 B9 E7  B8 27 5A 59  77 D3 7B B3  B7 D5 94 AE  B6 14 E9 AF  .%...'ZYw.{.........
0000008C  E3 6B DA D0                                                      .k..
```

Input1:

```
00000000  53 61 6C 74  65 64 5F 5F  59 E1 ED 02  4B 59 7B F5  69 10 F7 99  Salted__Y...KY{.i...
00000014  EB 93 7A 6E  A4 3B 89 E3  D1 C5 CF 25  9D FB 72 90  05 D2 83 7B  ..zn.;.....%..r....{
00000028  C4 9A B7 FC  15 F9 73 87  AE C7 25 F9  7D 72 3F 85  EF B9 55 04  ......s...%.}r?...U.
0000003C  12 03 3E 03  0B 44 D0 20  ED CB 16 38  0C 43 F2 4D  51 20 4C AB  ..>..D. ...8.C.MQ L.
00000050  07 5E 0E 51  39 07 FD AF  9E B9 85 26  7C 4E F8 FE  45 89 27 EE  .^.Q9......&|N..E.'.
00000064  99 1E EA 21  46 3D 89 C4  1D E2 75 19  16 A4 B5 7B  D6 08 D7 CC  ...!F=....u....{....
00000078  04 92 D4 B3  CE 9E 89 8A  BE CB 1F FC  6B BC 33 0E  94 E0 78 DD  ............k.3...x.
0000008C  98 78 1E 85                                                      .x..
```

Conclusiones

Mismo caso que con AES y que el caso anterior. El salt hace que se cambie la clave y el vector de inicialización, por lo que cambia el criptograma.

c. Contraseña sin salt

1) ECB

Comandos

Input:

openssl enc -camellia-256-ecb -k PracticasDeSeguridad2018 -in input.bin -out input_camellia_256_ecd_con_contra_nosalt.bin -nosalt;

Input1:

openssl enc -camellia-256-ecb -k PracticasDeSeguridad2018 -in input1.bin -out input1_camellia_256_ecd_con_contra_nosalt.bin -nosalt;

Resultados

Input:

```
D1 24 B7 C1  F3 09 41 9D  55 AA D0 63  82 08 A8 EF  D1 24 B7 C1
F3 09 41 9D  55 AA D0 63  82 08 A8 EF  D1 24 B7 C1  F3 09 41 9D
55 AA D0 63  82 08 A8 EF  D1 24 B7 C1  F3 09 41 9D  55 AA D0 63
82 08 A8 EF  D1 24 B7 C1  F3 09 41 9D  55 AA D0 63  82 08 A8 EF
D1 24 B7 C1  F3 09 41 9D  55 AA D0 63  82 08 A8 EF  D1 24 B7 C1
F3 09 41 9D  55 AA D0 63  82 08 A8 EF  D1 24 B7 C1  F3 09 41 9D
55 AA D0 63  82 08 A8 EF  F7 36 0A 15  80 6E 56 92  1C 31 4D C5
04 6C 73 23
```

Input1:

```
D1 24 B7 C1  F3 09 41 9D  55 AA D0 63  82 08 A8 EF  78 57 2E B1
D0 FD C8 84  55 D8 D7 72  27 C3 C6 CA  D1 24 B7 C1  F3 09 41 9D
55 AA D0 63  82 08 A8 EF  D1 24 B7 C1  F3 09 41 9D  55 AA D0 63
82 08 A8 EF  D1 24 B7 C1  F3 09 41 9D  55 AA D0 63  82 08 A8 EF
D1 24 B7 C1  F3 09 41 9D  55 AA D0 63  82 08 A8 EF  D1 24 B7 C1
F3 09 41 9D  55 AA D0 63  82 08 A8 EF  D1 24 B7 C1  F3 09 41 9D
55 AA D0 63  82 08 A8 EF  F7 36 0A 15  80 6E 56 92  1C 31 4D C5
04 6C 73 23
```

Diferencias

Input:

```
1. 00000000  D1 24 B7 C1  F3 09 41 9D  55 AA D0 63  82 08 A8 EF  D1 24 B7 C1  .$....A.U..c.....$..
2. 00000014  F3 09 41 9D  55 AA D0 63  82 08 A8 EF  D1 24 B7 C1  F3 09 41 9D  ..A.U..c.....$....A.
```

Input1:

```
1. 00000000  D1 24 B7 C1  F3 09 41 9D  55 AA D0 63  82 08 A8 EF  78 57 2E B1  .$....A.U..c....xW..
2. 00000014  D0 FD C8 84  55 D8 D7 72  27 C3 C6 CA  D1 24 B7 C1  F3 09 41 9D  ....U..r'....$....A.
```

Conclusiones

Lo primero que se puede apreciar es que el tamaño del criptograma es más pequeño ya que no hace falta meter el salt.

Además, se puede ver como en este caso el criptograma se comporta como cuando se usa vector de inicialización y clave, modificando únicamente el bloque en el que se ha cambiado el bit.

Finalmente, y como era de esperar, se comporta como AES.

2) CBC

Comandos

Input:

openssl enc -camellia-256-cbc -k PracticasDeSeguridad2018 -in input.bin -out input_camellia_256_cbc_con_contra_nosalt.bin -nosalt;

Input1:

openssl enc -camellia-256-cbc -k PracticasDeSeguridad2018 -in input1.bin -out input1_camellia_256_cbc_con_contra_nosalt.bin -nosalt;

Resultados

Input:

```
D0 FA 2D A3  5B 2F 60 B3  B5 58 02 87  45 02 40 28  D9 F5 66 1E
08 38 C3 73  9B 77 43 7C  D0 87 85 60  6B A8 12 DE  B5 FD 14 D9
64 F2 61 7F  B7 52 94 16  F2 F1 0B 8A  F4 3F 49 17  42 8B D1 5E
E3 7C 42 C5  F5 9F B0 F0  E5 EE 6C AC  58 38 DE 43  77 20 10 24
38 22 9C 98  62 A8 4D 90  65 33 D8 10  FD 79 A7 AA  AB 5F E3 A8
A1 84 1B 27  E0 B0 A7 8F  70 E7 6C 2A  FB 93 AF 35  80 72 ED 01
A1 12 5D 2C  58 E1 74 D3  4B 18 95 F5  AD 52 6F 1B  5B 82 B3 6B
06 33 D4 8E
```

Input1:

```
D0 FA 2D A3  5B 2F 60 B3  B5 58 02 87  45 02 40 28  FC 7F C4 48
81 62 DB AD  D6 DD FC C6  C5 F3 4A AC  D9 49 CF F1  6C 54 AB 88
63 D5 3C 5D  00 DD 98 1D  AD 21 E5 6A  48 10 97 D6  2C B0 B0 65
86 8C E8 69  04 F9 C0 30  AC 09 C2 9D  80 4D D4 10  A2 10 FA C0
DA 29 03 54  4E 3A A1 21  0E FD D0 1A  65 1F 6C 61  15 D7 0C 26
70 45 E6 24  14 93 E4 48  74 4D 4A 06  C0 9C 36 DA  A0 78 A8 FF
4A 06 44 87  07 7E 6C 21  80 24 E0 D2  5A C8 BF F7  0C 77 A6 AD
DB 67 E7 34
```

Diferencias

Input:

```
1. 00000000  D0 FA 2D A3  5B 2F 60 B3  B5 58 02 87  45 02 40 28  D9 F5 66 1E  ..-.[/`..X..E.@(..f.
2. 00000014  08 38 C3 73  9B 77 43 7C  D0 87 85 60  6B A8 12 DE  B5 FD 14 D9  .8.s.wC|...`k.......
3. 00000028  64 F2 61 7F  B7 52 94 16  F2 F1 0B 8A  F4 3F 49 17  42 8B D1 5E  d.a..R.......?I.B..^
4. 0000003C  E3 7C 42 C5  F5 9F B0 F0  E5 EE 6C AC  58 38 DE 43  77 20 10 24  .|B.......l.X8.Cw .$
5. 00000050  38 22 9C 98  62 A8 4D 90  65 33 D8 10  FD 79 A7 AA  AB 5F E3 A8  8"..b.M.e3...y..._..
6. 00000064  A1 84 1B 27  E0 B0 A7 8F  70 E7 6C 2A  FB 93 AF 35  80 72 ED 01  ...'....p.l*...5.r..
7. 00000078  A1 12 5D 2C  58 E1 74 D3  4B 18 95 F5  AD 52 6F 1B  5B 82 B3 6B  ..],X.t.K....Ro.[..k
8. 0000008C  06 33 D4 8E
```

Input1:

```
1. 00000000  D0 FA 2D A3  5B 2F 60 B3  B5 58 02 87  45 02 40 28  FC 7F C4 48  ..-.[/`..X..E.@(...H
2. 00000014  81 62 DB AD  D6 DD FC C6  C5 F3 4A AC  D9 49 CF F1  6C 54 AB 88  .b........J..I..lT..
3. 00000028  63 D5 3C 5D  00 DD 98 1D  AD 21 E5 6A  48 10 97 D6  2C B0 B0 65  c.<].....!.jH...,..e
4. 0000003C  86 8C E8 69  04 F9 C0 30  AC 09 C2 9D  80 4D D4 10  A2 10 FA C0  ...i...0.....M......
5. 00000050  DA 29 03 54  4E 3A A1 21  0E FD D0 1A  65 1F 6C 61  15 D7 0C 26  .).TN:.!....e.la...&
6. 00000064  70 45 E6 24  14 93 E4 48  74 4D 4A 06  C0 9C 36 DA  A0 78 A8 FF  pE.$...HtMJ...6..x..
7. 00000078  4A 06 44 87  07 7E 6C 21  80 24 E0 D2  5A C8 BF F7  0C 77 A6 AD  J.D..~l!.$..Z....w..
8. 0000008C  DB 67 E7 34
```

Conclusiones

Similar al anterior, pero con el modo CBC. Todos los bloques a partir del modificado, se modifican también y comportamiento similar a AES.

3) OFB

Comandos

Input:

openssl enc -camellia-256-ofb -k PracticasDeSeguridad2018 -in input.bin -out input_camellia_256_ofb_con_contra_nosalt.bin -nosalt;
Input1:
openssl enc -camellia-256-ofb -k PracticasDeSeguridad2018 -in input1.bin -out input1_camellia_256_ofb_con_contra_nosalt.bin -nosalt;

Resultados

Input:

```
D0 FA 2D A3  5B 2F 60 B3  B5 58 02 87  45 02 40 28  D9 F5 66 1E
08 38 C3 73  9B 77 43 7C  D0 87 85 60  6B A8 12 DE  B5 FD 14 D9
64 F2 61 7F  B7 52 94 16  F2 F1 0B 8A  F4 3F 49 17  42 8B D1 5E
E3 7C 42 C5  F5 9F B0 F0  E5 EE 6C AC  58 38 DE 43  77 20 10 24
38 22 9C 98  62 A8 4D 90  65 33 D8 10  FD 79 A7 AA  AB 5F E3 A8
A1 84 1B 27  E0 B0 A7 8F  70 E7 6C 2A  FB 93 AF 35  80 72 ED 01
A1 12 5D 2C  58 E1 74 D3
```

Input1:

```
D0 FA 2D A3  5B 2F 60 B3  B5 58 02 87  45 02 40 28  DD F5 66 1E
08 38 C3 73  9B 77 43 7C  D0 87 85 60  6B A8 12 DE  B5 FD 14 D9
64 F2 61 7F  B7 52 94 16  F2 F1 0B 8A  F4 3F 49 17  42 8B D1 5E
E3 7C 42 C5  F5 9F B0 F0  E5 EE 6C AC  58 38 DE 43  77 20 10 24
38 22 9C 98  62 A8 4D 90  65 33 D8 10  FD 79 A7 AA  AB 5F E3 A8
A1 84 1B 27  E0 B0 A7 8F  70 E7 6C 2A  FB 93 AF 35  80 72 ED 01
A1 12 5D 2C  58 E1 74 D3
```

Diferencias

Input:

```
1. 00000000  D0 FA 2D A3  5B 2F 60 B3  B5 58 02 87  45 02 40 28  D9 F5 66 1E  ..-.[/`..X..E.@(..f.
```

Input1:

```
1. 00000000  D0 FA 2D A3  5B 2F 60 B3  B5 58 02 87  45 02 40 28  DD F5 66 1E  ..-.[/`..X..E.@(..f.
```

4) Conclusiones

Como se ha podido ver, en este apartado al igual que en los anteriores el resultado es idéntico a AES.