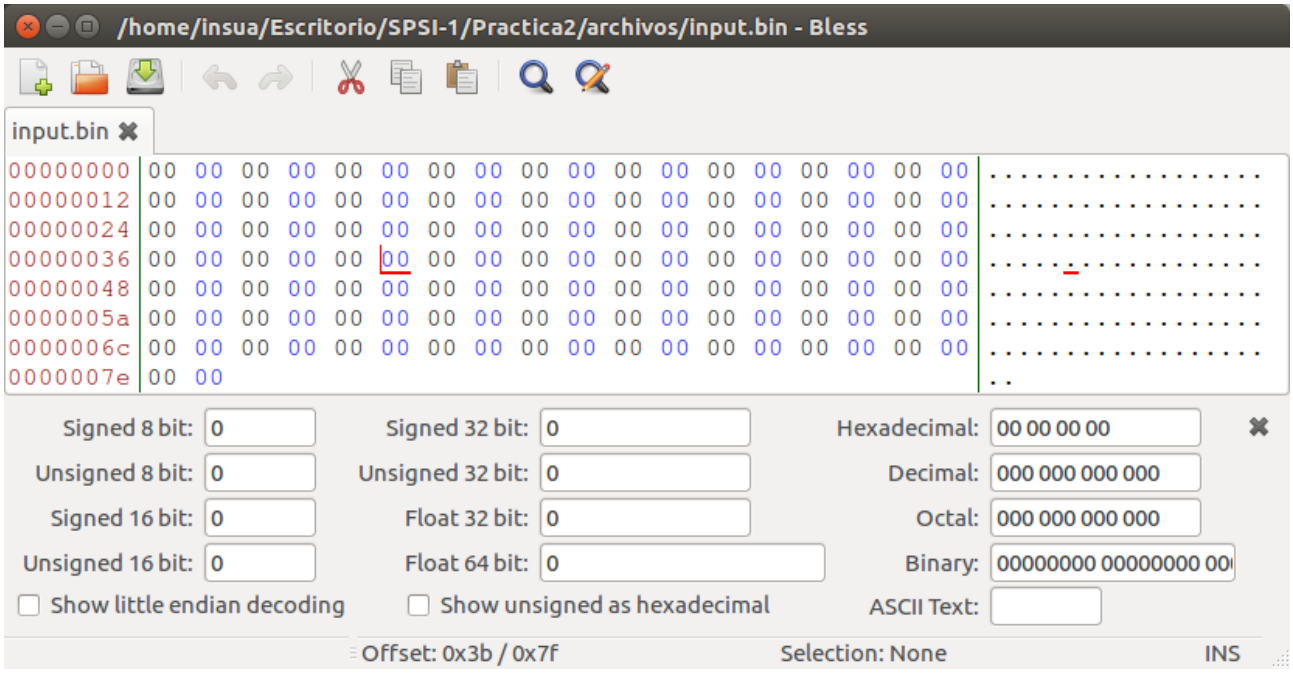
SEGURIDAD Y PROTECCIÓN DE SISTEMAS INFORMÁTICOS

PRÁCTICA 2

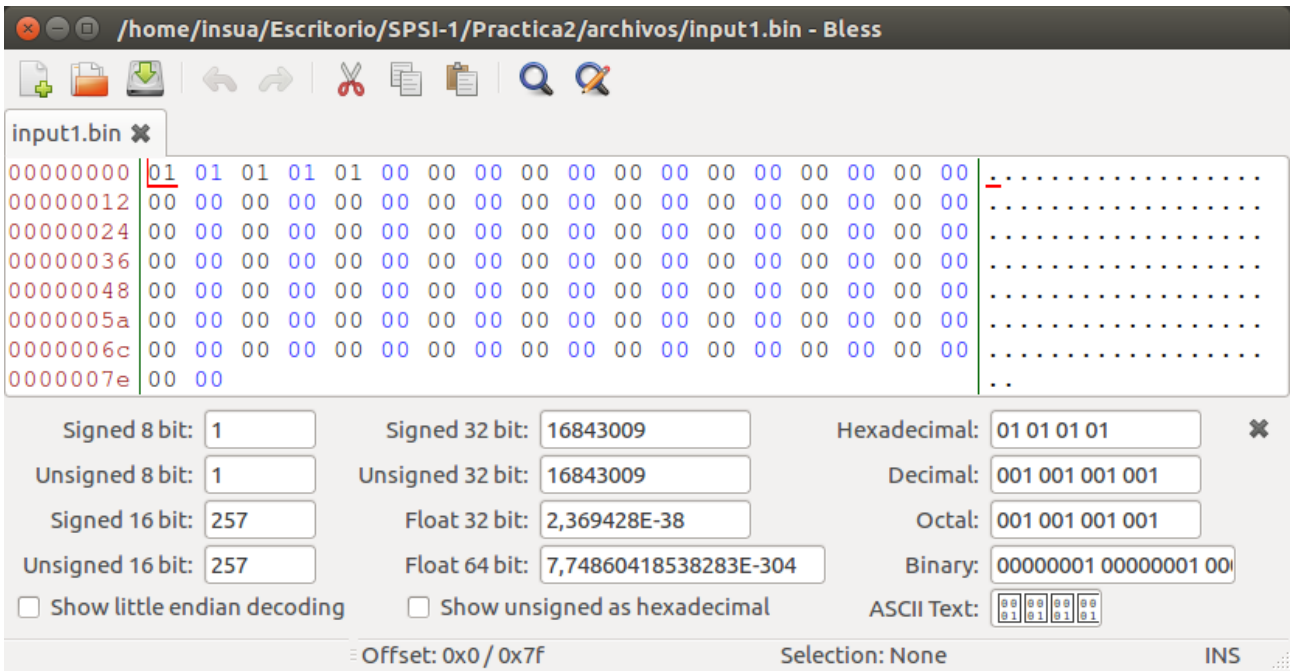
Jose Manuel Martínez de la Insua.

1.



Captura correspondiente al fichero input.bin

2.

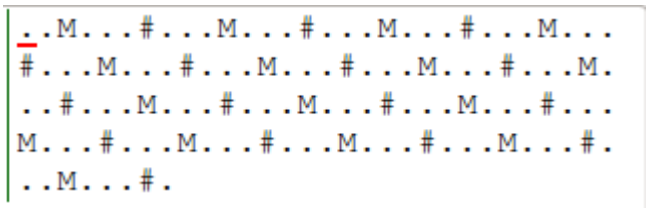


Captura correspondiente al fichero input1.bin

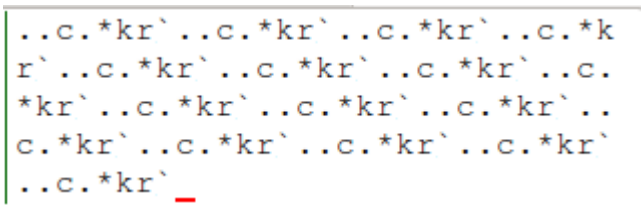
3 .

ECB

Clave débil:



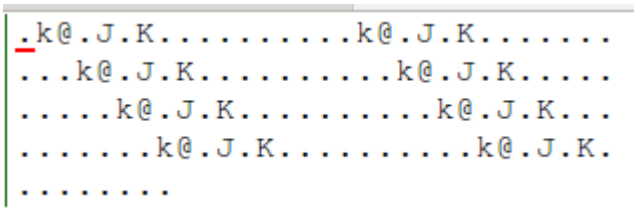
Clave semi-débil:



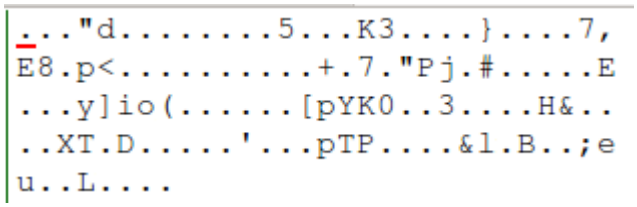
Tanto para el cifrado con clave débil, como para el cifrado con clave semi-débil se repite el mismo patrón. Esto es así porque el algoritmo ECB separa en bloques y aplica un cifrado la misma clave a cada uno de los bloques, de modo que como todos los bloques son exactamente iguales y cada bloque se cifra de la misma manera, el resultado es la repetición del mismo cifrado 16 veces.

CBC

Clave débil:



Clave semi-débil:

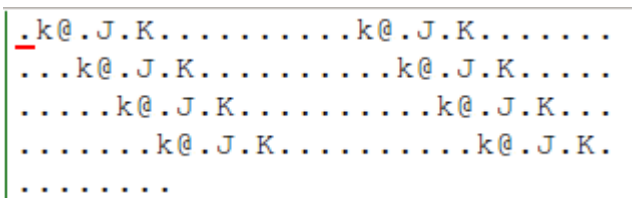


Clave débil se repite el mismo patrón ya que al sumar el vector de inicialización con el valor 0, obtenemos el propio vector de inicialización, inmediatamente después de cifrar dicho vector, vuelve a sumar con 0 y vuelve a cifrar ... y así sucesivamente.

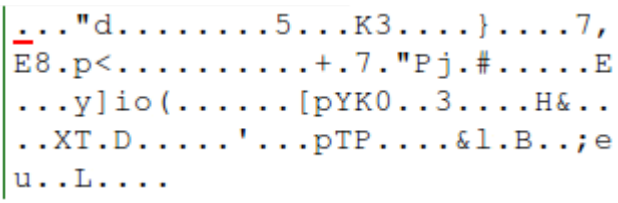
La razón por la cual el patrón es el mismo es que al ser clave débil, cifrar dos veces con la misma clave equivale a descifrar, luego siempre estamos cifrando y descifrando el mismo valor y es por eso por lo que con clave semi-débil no se obtiene el mismo resultado, ya que las claves débiles vienen dadas en parejas, es decir, para descifrar el texto cifrado con una clave débil necesitamos cifrar con su pareja (y no con ella misma como en el caso de la clave débil), luego al cifrar y sumar se obtienen resultados diferentes.

OFB

Clave débil:



Clave semi-débil:

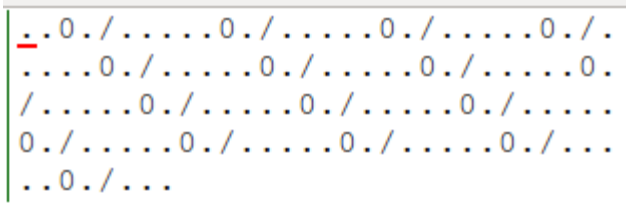


En este caso podemos ver que para el cifrado con clave débil también se repiten los bloques de 8 de la forma: el primer bloque igual al tercero, al quinto al séptimo y así sucesivamente, de igual forma lo hacen los bloques pares. Esto se debe a que se cifran las semillas (vectores) y después se suman (XOR) con su correspondiente bloque. Al ser clave débil, cifrar con la misma clave equivale a descifrar, y al ser cada uno de los bloques 0, la suma con la semilla siempre es igual a la propia semilla; dicho todo esto, podemos afirmar que los bloques pares equivalen al vector de inicialización y los impares al vector de inicialización cifrado.

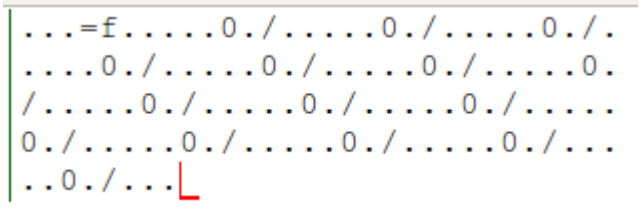
Análogamente podemos ver que para la clave semi-débil esto no se cumple y es por la misma razón que en el apartado anterior, como no ciframos con la pareja de la clave semi-débil, al cifrar vamos obteniendo semillas diferentes.

4.

Input 0:



Input 1:



Al ser cifrado ECB, sabemos que divide en bloques y a cada uno de ellos le aplica el mismo cifrado (con la misma clave, en este caso clave fuerte = 0E329232EA6D0D73), esto es, independientemente de la clave elegida el resultado va a ser una secuencia de bloques idénticos para el caso de input0, ya que todos los Bytes son 0. Para el caso de input1, todos los bloques son exactamente iguales exceptuando uno, el que corresponde al carácter 1.

5.

Input 0:

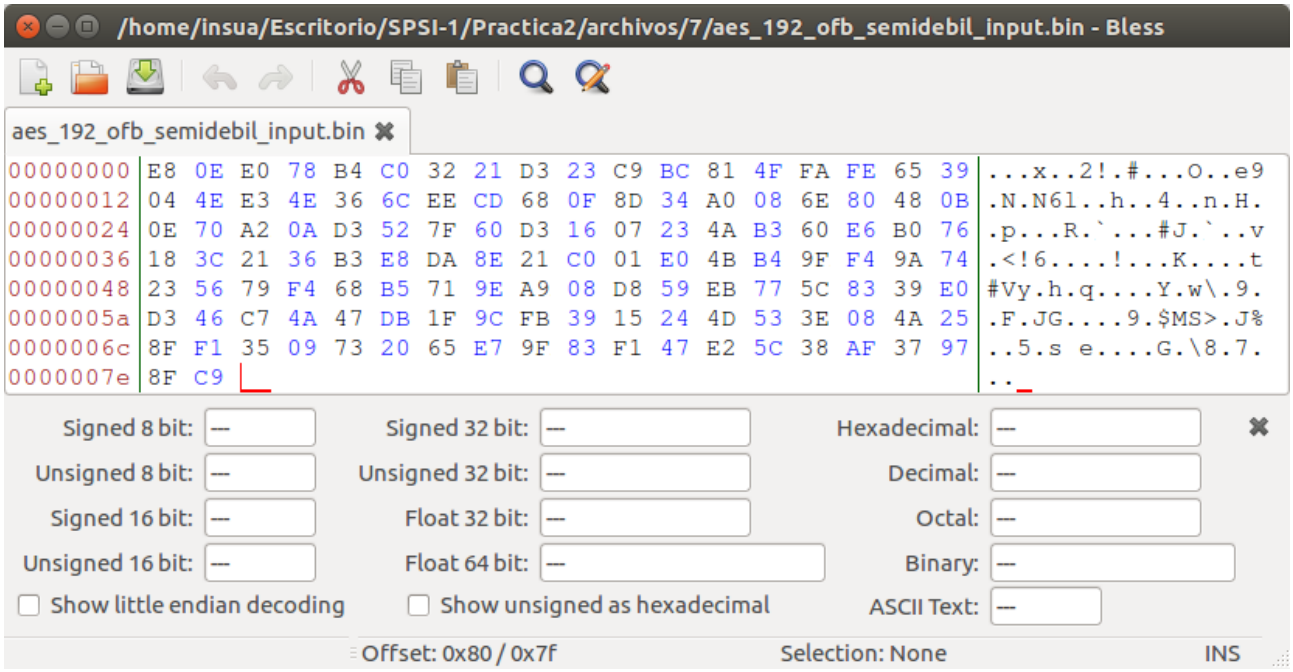
```
N....:zN...Q.:!.o....K.....<..
W..G{Z.#C=.>.H_...&.o...mq=..<
4`..+.*.{.:.R$..2.R:Xt.V...3(.
............/c..u)..D.38z.....
...v...v_
```

Input 1:

```
......>...47Q$.....M...W..o..A
.......~......bg..-.Q...};..V"
.a....f.........-Q.o...g=.=...
..xH..c..S=1/2HY.....7OI.g.z%S
. M.2X.'_
```

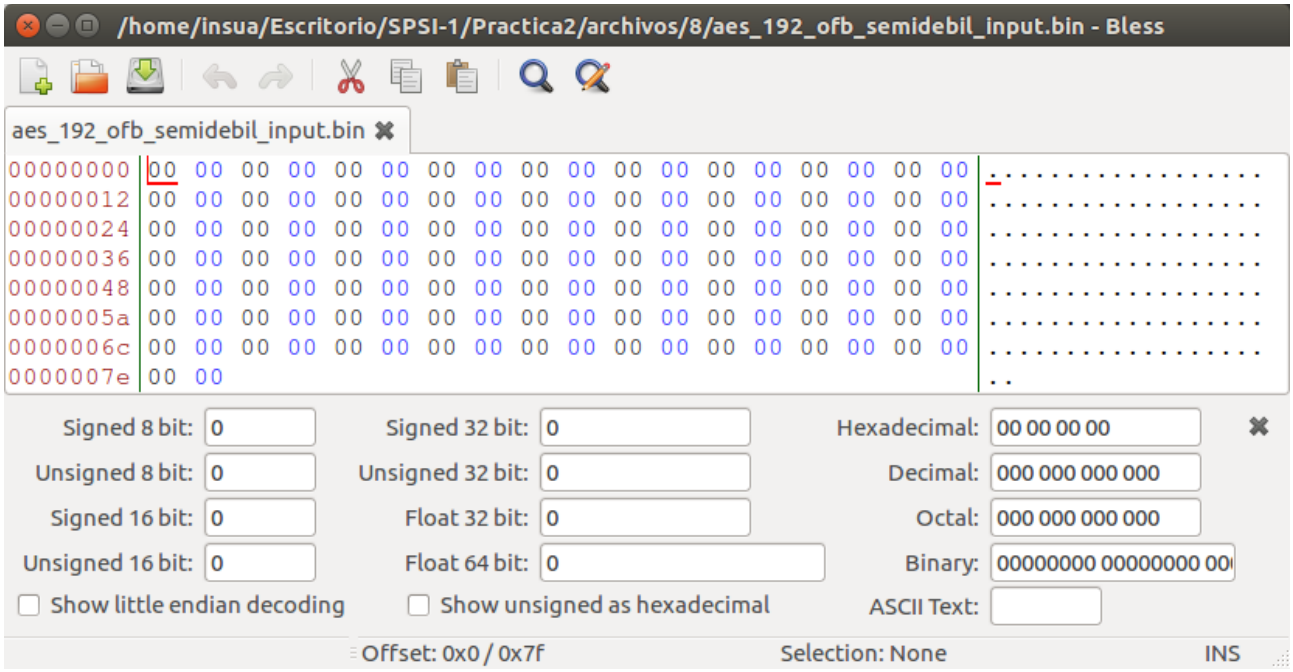
Al ser cifrado CBC, sabemos que divide en bloques, y la salida de cada bloque se suma a la entrada de la siguiente, de modo que ,como hemos cifrado con clave fuerte = 0E329232EA6D0D73, sabemos que no hay otra clave que pueda “anular” a nuestra clave y por tanto cada uno de los bloques cifrados es diferente al anterior tanto para input0 como para input1.

7.



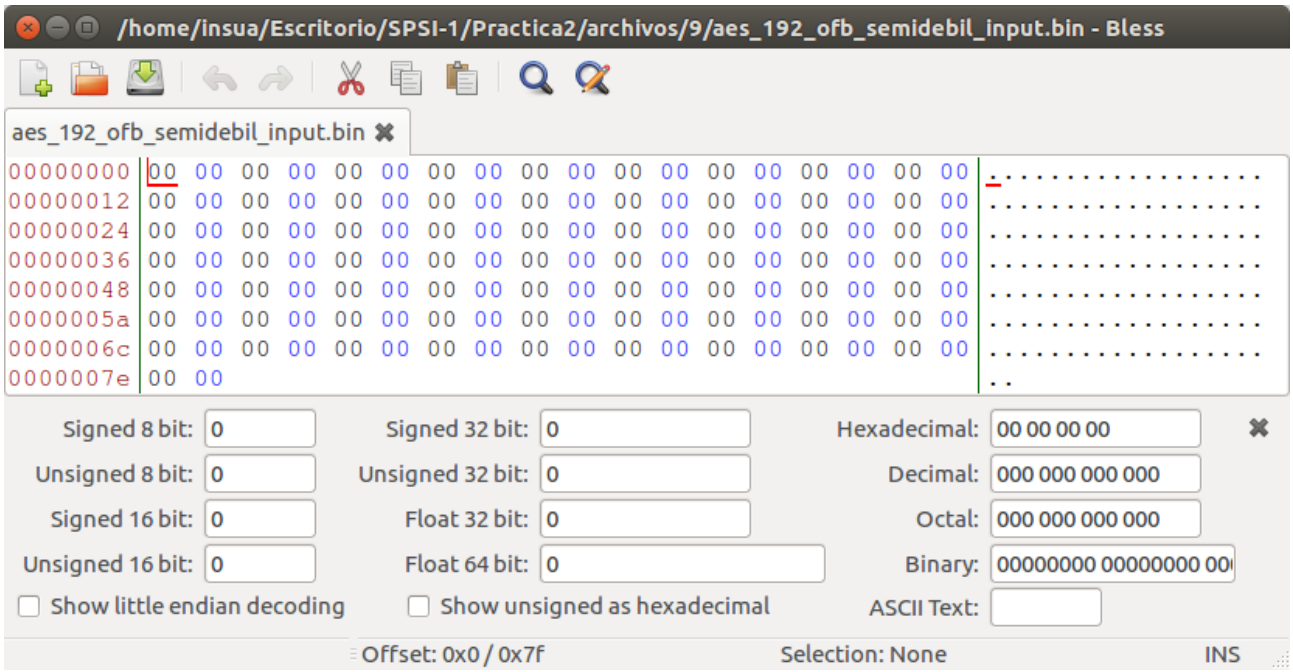
Captura correspondiente al cifrado aes-192-ofb de input.bin.

8.



Captura correspondiente al descifrado aes-192-ofb con la misma clave y vector de inicialización.

9.



Cifrar dos veces con la misma clave y el mismo vector de inicialización en aes-192-ofb equivale a descifrar.

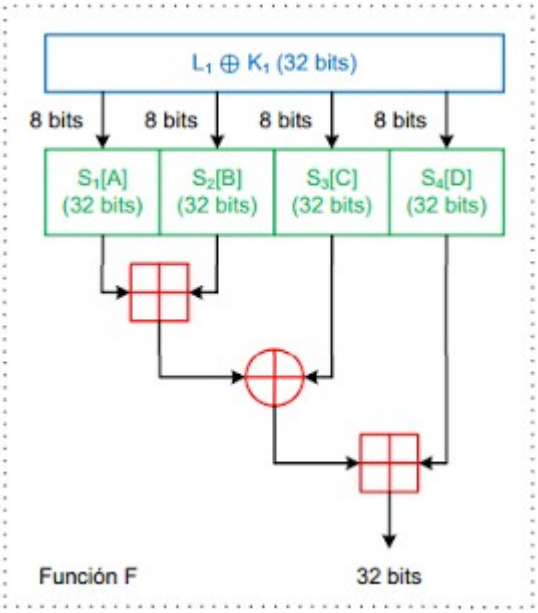
10.

Cifrados por Bloque – Blowfish

Este algoritmo esta compuesto por 18 semiclaves (K) y 4 cajas (S).

Primero tendremos que describir la función F, la cual se encarga de sustituir los valores resultantes de las operaciones XOR utilizando las cajas S.

La función F divide el grupo de 32bits en 4 grupos de 8bits el bloque a y bloque b se buscan en las cajas sustitución(representadas con la letra "S" en el diagrama), el valor representado por el primer octeto de la primer caja se suma al valor representado de el segundo octeto de la segunda caja y al resultado se saca el modulo de 2^{32}, posteriormente a este resultado se aplica una operación XOR con el valor representado por el tercer bloque en la tercer caja y al resultado de esto se le suma el valor representado por el cuarto octeto en la cuarta caja y al final se vuelve a aplicar el modulo 2^{32}.

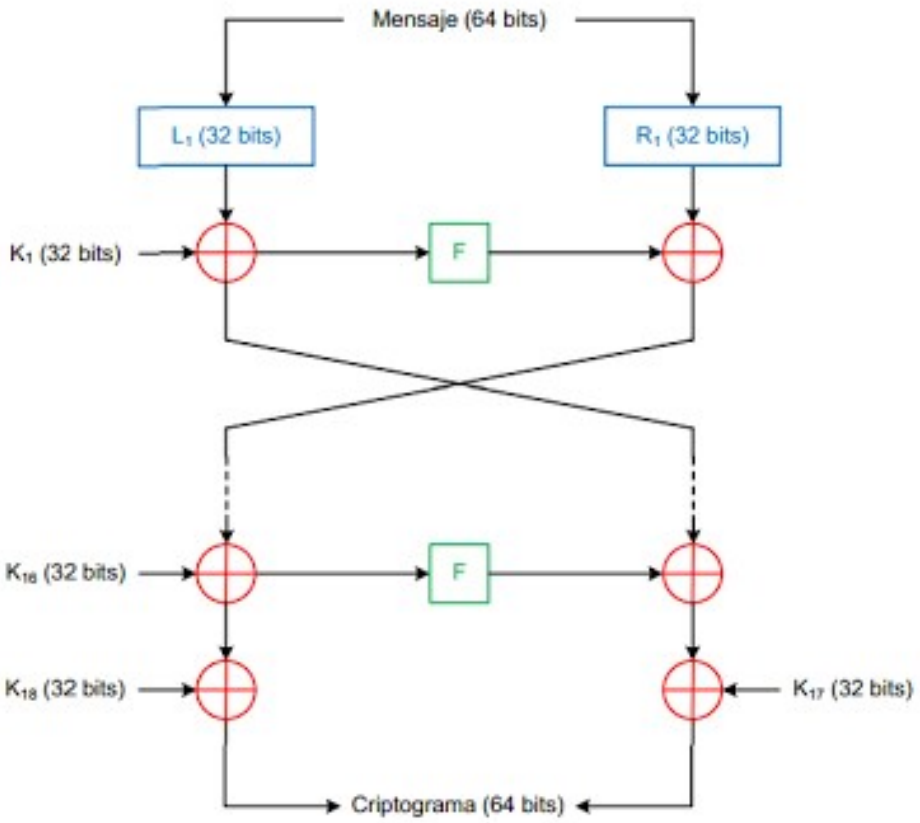


$$f(L_i \oplus K_i) = ((((S_1[A] + S_2[B]) \mod 2^{32}) \oplus S_3[C]) + S_4[D]) \mod 2^{32}$$

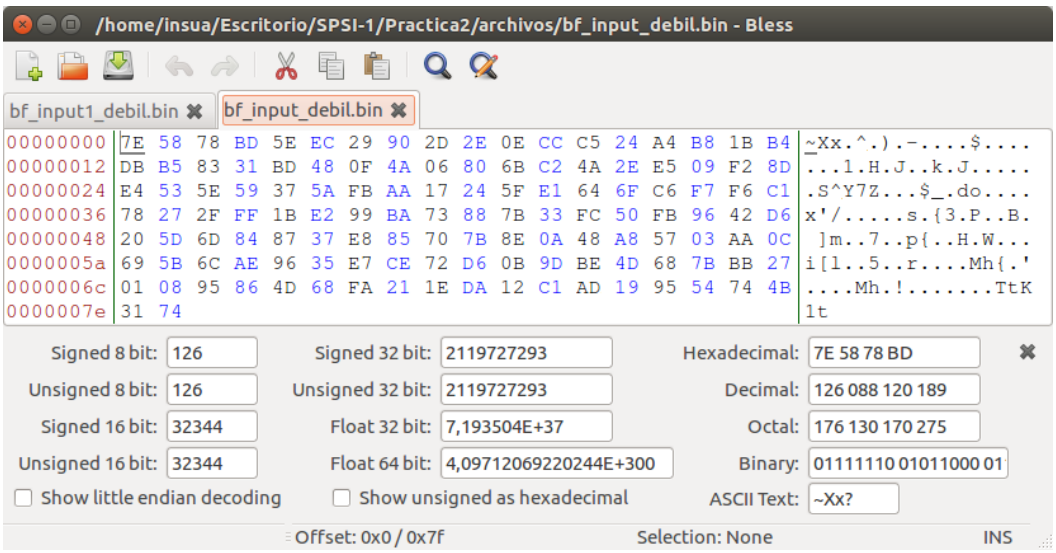
Una vez entendido que es la función F del Blowfish paso a describir el algoritmo de cifrado:

- Un bloque de 64bits se divide en dos bloques de 32bits (L y R)
- Se realiza una operación XOR a los primeros 32bits(L) con la primera subclave(K) y se realiza la función F con el resultado de la operación.
- Se realiza una operación XOR con la segunda parte de los 32 bits(R) con el resultado de la función F(L).

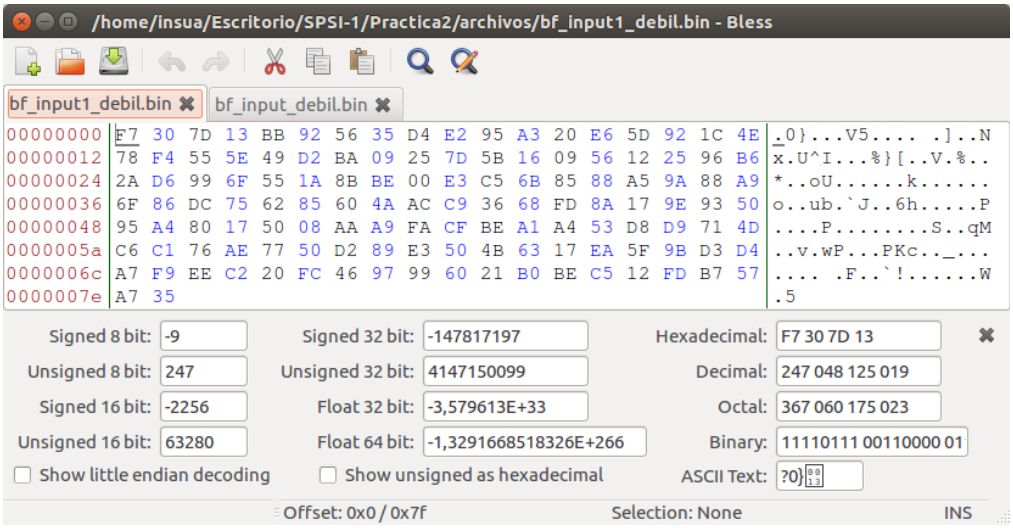
- Se intercambian posiciones (R pasa a ser L y L pasa a ser R) y se repite el proceso anterior por 16 iteraciones.
- Al terminar la ultima iteración no se realizara el intercambio.
- Se realiza la operación XOR entre el valor alojado en L y la subclave 18.
- Se realiza la operación XOR entre el valor alojado en R y la subclave 17.
- Se unen los dos fragmentos del bloque para generar nuevamente un bloque de 64bits ya cifrado.



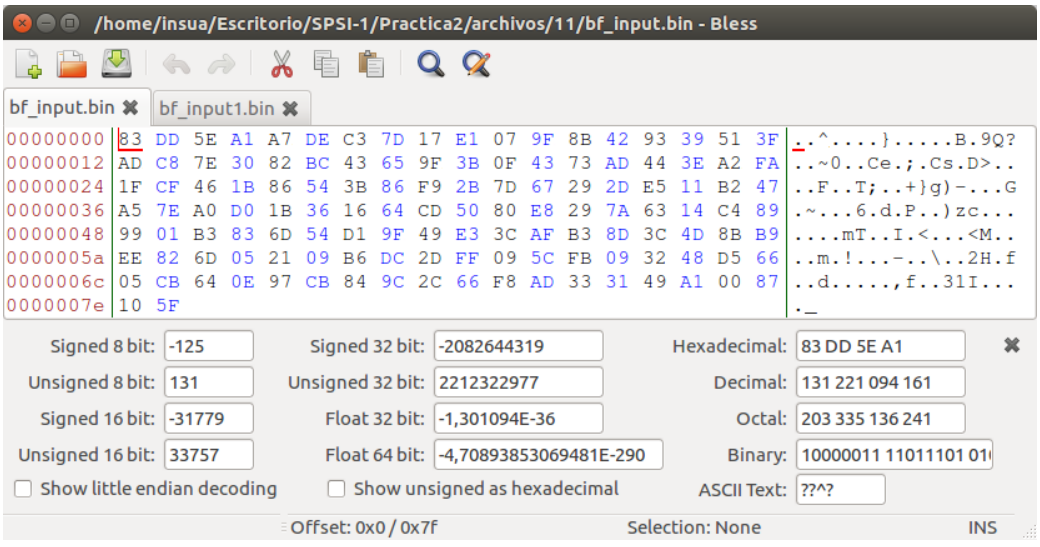
Input0.bin cifrado con Blowfish y clave débil.



Input1.bin cifrado con Blowfish y clave débil.



Input0.bin cifrado con Blowfish y clave semi-débil.



Input1.bin cifrado con Blowfish y clave semi-débil.