

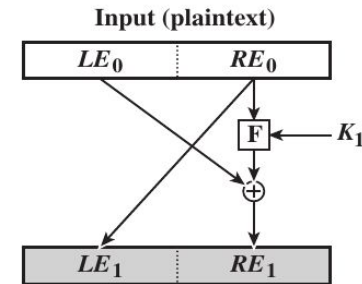
C7-P1. Criptografía simétrica

1. Programe una función en Python que implemente una ronda de cifrador Feistel

- Defina el tamaño de bloque
- Reciba el “texto plano” como parámetro en formato binario
- Reciba F como parámetro
 - Defina una función F de prueba
- Reciba K_1 como parámetro
 - Defina una llave K_1 de prueba
- Retorne el “texto cifrado” como parámetro en formato binario

2. Demuestre a través de un test que su cifrador es reversible

- El diseño e implementación del test queda a su criterio.
- El ejecutar el programa sin parámetros debe correr este test.



NOTA 1: Debe entregar un PDF con la descripción de su solución y solo 1 archivo python con la implementación y el test.

NOTA 2: Al ejecutar su programa sin parámetros ejecuta directamente el test del punto 2.

NOTA 3: Los revisores ejecutarán el programa sin parámetros y debe pasar el test que ud. diseñó e implementó.

NOTA 4: Puede usar funciones de librerías externas para implementar F , aunque se sugiere una implementación muy simple de F por ahora.

NOTA 5: Considere las entradas y salidas como binarias. Puede usar representación hexadecimal o ASCII para facilitar la lectura

C7-P1 Explicación del Código

Explicación primera parte del main:

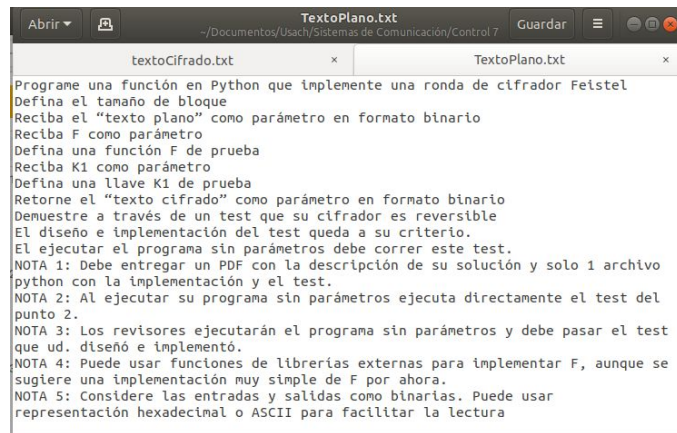
- Primero se define el tamaño del bloque, el cual será de 32 bytes.

- Luego se lee el texto en formato binario. El texto ingresado en el que se muestra en la imagen de arriba, que no es que más que el enunciado de este control.

- Se define la llave, la cual será de 16 bytes.

- Antes de realizar el cifrador Feistel, el texto binario se separa en bloques (del tamaño indicado anteriormente)

- Desarrollar el cifrado feistel, ingresando como parámetros los bloques, el tamaño de los bloques, la llave y la función xor (explicada en la siguiente diapositiva).



```
35
36 def main():
37     TamanoBloque = 32
38     #Lectura del archivo y lo transforma e texto en formato binario
39     f = open("TextoPlano.txt", "rb")
40     textoBinario = f.read()
41     f.close()
42
43     #Defino la llave
44     key = b"Llave de 16bytes"
45
46     #Se separa el texto en bloques de 32 bits
47     blocks = separarBloque(textoBinario, TamanoBloque)
48
49     #Se realiza el cifrado feistel
50     textoCifrado = feistel(blocks, TamanoBloque, key, xor)
51
```

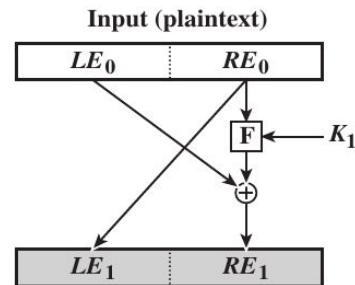
C7-P1 Explicación del Código

Explicación cifrador Feistel y función F:

EL cifrador Feistel funciona de la misma forma que el dibujo inicial, es decir, primero se divide el bloque en dos partes (para este caso 16 bytes tiene cada parte), después se asigna a las variables L y R, la parte izquierda y derecha respectivamente.

Nuestra función F de prueba es simplemente una puerta XOR (función xor en el código), en la cual se ingresa la llave "key" y el lado derecho del bloque (es decir la variable R). Al resultado de eso se le aplica una nueva puerta XOR, pero ahora entre L y lo que sale de F.

Finalmente se juntan las dos mitades del bloque. Esto se realiza por la cantidad de bloques que se obtuvieron del texto inicial.

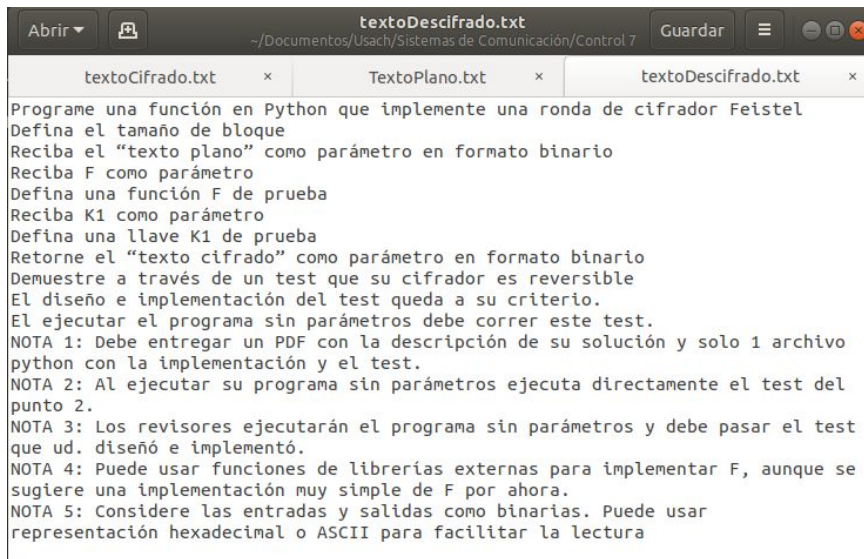


```
11
12 def feistel(blocks, TamanoBloque, key, f):
13     mitad = int(TamanoBloque/2)
14     resultado= b""
15     for block in blocks:
16         L = block[:mitad]
17         R = block[mitad:]
18         #La funcion F de prueba es una funcion Xor
19         L = (xor(L, f(key, R)))
20
21         resultado += L + R
22
23     return resultado
24
25 def xor(a , b):
26     output=b""
27     for a, b in zip(a, b):
28         output += bytes([a ^ b])
29     return output
30
```


C7-P1 Resultados

Al ejecutar el programa, nos creará el texto cifrado, el texto descifrado y mostrará por la consola el resultado del test, si ambos textos son iguales arrojará la opción de que son iguales sino, dirá que no lo son.

En la imagen de al lado se puede observar que el texto descifrado es igual al inicial.



```
Programa una función en Python que implemente una ronda de cifrador Feistel
Defina el tamaño de bloque
Reciba el "texto plano" como parámetro en formato binario
Reciba F como parámetro
Defina una función F de prueba
Reciba K1 como parámetro
Defina una llave K1 de prueba
Retorne el "texto cifrado" como parámetro en formato binario
Demuestre a través de un test que su cifrador es reversible
El diseño e implementación del test queda a su criterio.
El ejecutar el programa sin parámetros debe correr este test.
NOTA 1: Debe entregar un PDF con la descripción de su solución y solo 1 archivo
python con la implementación y el test.
NOTA 2: Al ejecutar su programa sin parámetros ejecuta directamente el test del
punto 2.
NOTA 3: Los revisores ejecutarán el programa sin parámetros y debe pasar el test
que ud. diseñó e implementó.
NOTA 4: Puede usar funciones de librerías externas para implementar F, aunque se
sugiere una implementación muy simple de F por ahora.
NOTA 5: Considere las entradas y salidas como binarias. Puede usar
representación hexadecimal o ASCII para facilitar la lectura
```

```
alberto@alberto-HP-Pavilion-Laptop-15-cc5xx:~/Documentos/Usach/Sistemas de Comunicación/Control 7$ python3 CifradorFeistel.py
Archivo Cifrado Creado
TEST
Se descifra el texto Cifrado y se compara este con el texto inicial para ver si son iguales
Son iguales
Archivo Descifrado Creado
```

C7-P2. Criptografía asimétrica

1. En un sistema de llave pública que usa RSA ud. intercepta un mensaje cifrado $C=10$ que se envió a un usuario cuya llave pública es $e=5$, $n=35$
 - a. ¿Cuál es el texto plano M ?

NOTA 1: Debe entregar un PDF con la descripción de su solución

C7-P2. Criptografía asimétrica

- a. Primero se debe comprender como es el cifrado y descifrado en el algoritmo RSA, donde M sería el texto plano como se indica, y C sería el texto cifrado, que siguen lo siguiente:

$C = M^e \bmod n$ //Para cifrar el texto

$M = C^d \bmod n$ //Para descifrar el texto cifrado

Ya se tiene el valor de e y el valor de n, solo falta el valor de d. obtener este valor, se debe cumplir la relación:

$$M = M^{(ed)} \bmod n$$

La relación anterior se cumple si e y d son inversos multiplicativos módulo $\varphi(n)$, donde $\varphi(n)$ es la función φ de Euler. La relación entre e y d es expresada como:

$$ed \bmod \varphi(n) = 1$$

Ahora se sabe que si se tiene dos números primos p y q, entonces $\varphi(p \cdot q) = (p - 1) \cdot (q - 1)$, entonces como se sabe que $n=35$, para encontrar el valor de $\varphi(n)$ se tienen que encontrar dos números que sean primos que multiplicados den 35.

C7-P2. Criptografía asimétrica

Para determinar $\varphi(35)$ se deben buscar que números primos menores que 35 son primos y multiplicados, estos puede ser: 2,3,5,7,11,13,17,19,23,29,31. Los cuales al ser pocos se obtiene que son el 5 y el 7 = 35, entonces con la fórmula:

$$\varphi(p \cdot q) = (p - 1) \cdot (q - 1) = \varphi(7 \cdot 5) = (7 - 1) \cdot (5 - 1) = 24.$$

Otra forma de encontrar este valor es con los números que son relativamente primos a 35, que en total son 24 números. Entonces ya obteniendo el valor de $\varphi(35) = 24$, se puede obtener el valor de d en la relación $ed \bmod \varphi(n) = 1$

$$5d \bmod 24 = 1 \rightarrow d = 5,$$

por lo tanto $M = C^d \bmod n$ es igual a $M = 10^5 \bmod 35 = 5$

Entonces el valor de M = 5