C7-P1. Criptografía simétrica

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

- a. Defina el tamaño de bloque
- b. Reciba el "texto plano" como parámetro en formato binario
- c. Reciba F como parámetro
 - Defina una función F de prueba
- d. Reciba K, como parámetro
 - Defina una llave K₁ de prueba
- e. Retorne el "texto cifrado" como parámetro en formato binario

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

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

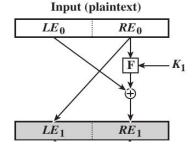


NOTA 2: Al ejecutar su programa sin parámetros ejecuta directamente el test del punto 2.

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

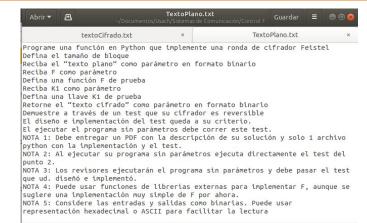




C7-P1 Explicación del Código

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).



```
def main():
    TamanoBloque = 32
    #Lectura del archivo y lo transforma e texto en formato binario
    f = open("TextoPlano.txt", "rb")
    textoBinario = f.read()
    f.close()

#Defino la llave
    key = b"Llave de 16bytes"

#Se separa el texto en bloques de 32 bits
    blocks = separarBloque(textoBinario, TamanoBloque)

#Se realiza el cifrado feistel
    textoCifrado = feistel(blocks, TamanoBloque, key, xor)
```



C7-P1 Explicación del Código

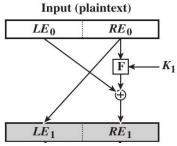
Explicación cifrador Feistel y función F:

EL cifrador Feistel funciona de la misma forma que el dibujo inicial, es decir, primero se divide el bloque es 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.





```
def feistel(blocks, TamanoBloque, key, f):
12
13
         mitad = int(TamanoBloque/2)
         resultado= b""
14
15
         for block in blocks:
             L = block[:mitad]
                 block[mitad:]
             L = (xor(L, f(key, R)))
             resultado += L + R
21
         return resultado
     def xor(a , b):
         output=b""
         for a, b in zip(a, b):
             output += bytes([a ^
                                   b1)
29
         return output
```

C7-P1 Explicación del Código

Explicación segunda parte del main y test:

Cuando se realiza el cifrado Feistel, se obtiene el texto cifrado, el cual se muestra en la imagen de abajo (textoCifrado.txt). Se puede apreciar que no cambió el texto del todo, se pueden ver algunas palabras o letras siguen iguales, esto se debe a que solo se realiza una ronda en el algoritmo, si se realizaran más rondas, en el texto cifrado no se entendería nada

Al texto cifrado mencionado, ahora se le realiza un descifrado, para realizarlo se utiliza el mismo algoritmo de Feistel, con la misma llave y con la misma función F.

Después de obtener el texto descifrado, se le limpian los espacios vacíos y se realiza el test. El cual no es más que comparar el texto inicial con el descifrado y ver si son iguales. Con esto se puede comprobar que el cifrador es reversible. (Al ser estos iguales).



```
#Se realiza el cifrado feistel
textoCifrado = feistel(blocks, TamanoBloque, key, xor)

w = open("textoCifrado.txt", "wb")
w.write(textoCifrado)
w.close()
print("Archivo Cifrado Creado")

blocks = separarBloque(textoCifrado, TamanoBloque)
textoDescifrado = feistel(blocks, TamanoBloque, key, xor)

textoDescifrado = limpiarEspacios(textoDescifrado)

print("TEST \n")
print("Se descifra el texto Cifrado y se compara este con el texto inicial para ver si son iguales")
if textoBainario == textoDescifrado:
    print("Son iguales\n")
else:
    print("No son iguales\n")

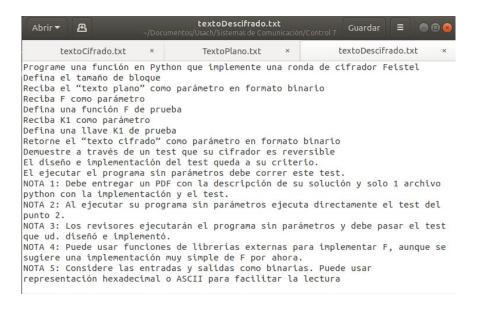
wd = open("textoDescifrado.txt", "wb")
wd.vrite(textoDescifrado)
wd.close()
print("Archivo Descrifrado Creado")
```

```
textoCifrado.txt
  Abrir ▼
!x2(2\CD\CZ qina el tamaño dL #\F8\8A\CZeeR[]]<! `ze=el "texto planF
                                                                  FA\93\84e&e()~4nië>etro en
formato 👭 f`zxi{ۥ 🙉 😘 😘 🚜 f~a7Z como parámetro/ 😘 🚱 🚗 . %! ᠪም l'Ojdigión F de prueba6 v/
198098Es98c=parámetro
DefinHldjgejhK[%]66]7+re prueba
Retorne hi9e e1&&/bygmgrado" como par 68\AZ,qx<j7,0 @ @sk}wto binario
DemueZ8f}e5!b&p1t'6be un test que su }{yu-e100 []]}su0ue100ersible
El diseÔ(CO(FZ)e-lyl >tc ueión del test quF&
99h rze1199lc799o.
El ejecutar eA>нg"bv"#6uux #arámetros debe 🔐
                                              lKT nd1ie not6b.
nota 1: Debe eMv3ffcar &<d[M0, waon la descripcincallak sx|obh.meo, س y solo 1
archD:g:e5u`*1b`us\86\AGa implementaciócl*l
(1 ('/ PRINKUA 2: Al ejecutarLou$ BOWE fq&"4pybfx parámetros ejeJ9p{e0hd1r&fx1{re el test del
punw}v%k2 F*g:9Los revisores ejN}uoer 1#o+Wqgrama sin parámHj3|z i 7 tv,a$uar el test
que uX.$\frac{9}{2}i=ue"\frac{97\frac{97}{2}}{2}\S\frac{90}{2}M>plementó.
MOTA 🔞5(I~ s d!eob6qcnciones de libre 7 👭 سم pq19!blf7#ara implementar c%3r0;{qex(w5b}qiere una
implemeRte5h_86\89M ,.iyfdqple de F por ahoWi*@No\Es$i.Nijdidere las entradB?#f+2w`(&y1
bozo binarias. PuedH/}3<d 9%corbiación hexadecimKa"p) CGXzs8ja0facilitar la lec9370 de
```

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.



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 yer si son iguales

Se descifra el texto Cifrado y se compara este con el texto inicial para ver si son iguales Son iguales

Archivo Descrifrado 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



Nombre:

RUT:

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 mod n //Para cifrar el texto
M = C^d mod 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:

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

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 mod
$$\varphi(n) = 1$$

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 $\phi(35)$ 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 mod $\varphi(n)$ = 1

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

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

Entonces el valor de M = 5