Proyecto CLCript
Cuadernos de Laboratorio de Criptografía. Entrega nº 1. Ultima actualización 24/10/18
Autor: Dr. Jorge Ramió Aguirre (@criptored)
Prácticas con algoritmos DES y AES: rellenos y modos de cifra

- Software safeDES: http://www.criptored.upm.es/software/sw_m001j.htm
- Software AESPhere: http://www.criptored.upm.es/software/sw_m001p.htm
- Lectura de interés: https://en.wikipedia.org/wiki/Padding_(cryptography)
- Tablas y códigos: http://www.criptored.upm.es/descarga/Codigos_y_tablas_de_uso_frecuente_en_criptografia.pdf

Objetivos:

1. Observar dos tipos de relleno, zero padding usado en el algoritmo DES y PKCS7 usado en el algoritmo AES.
2. Comprobar el cifrado en mod ECB y CBC.

I. DES modo ECB

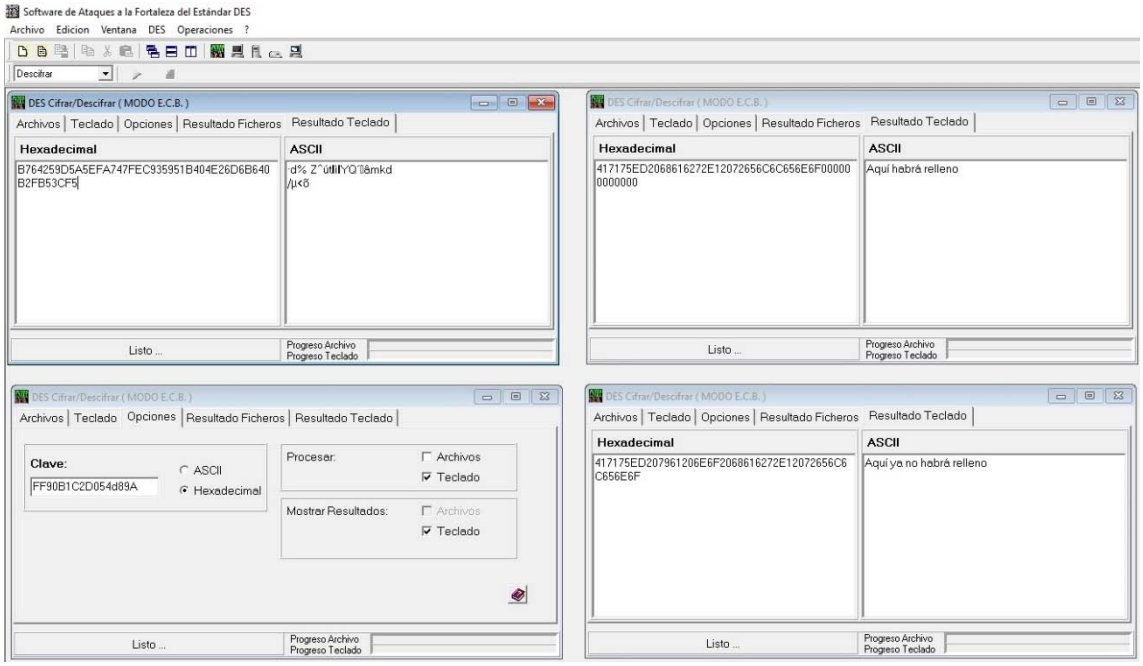
Ejercicio 1)

1.1. Cifra el texto M1 = Aquí habrá relleno
Con la clave K = 0x FF90B1C2D054d89A
1.2. Descifra el criptograma y observa el relleno.

Ejercicio 2)

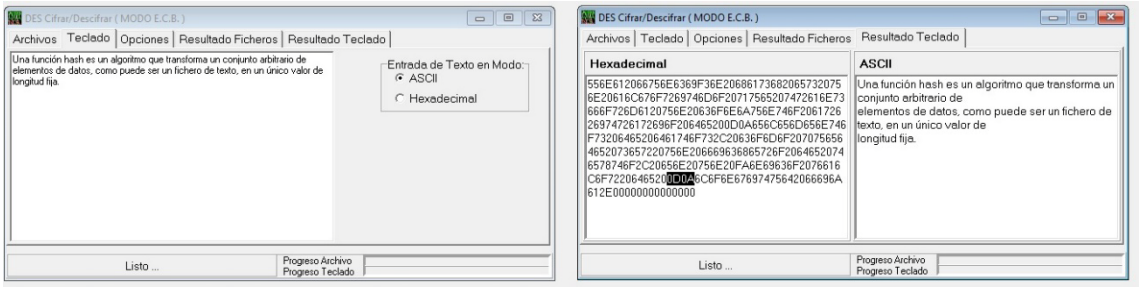
2.1. Cifra el texto M2 = Aquí ya no habrá relleno
Con la clave K = 0x FF90B1C2D054d89A
2.2. Descifra el criptograma y observa el relleno.
2.3. Saca conclusiones de lo visto.

Comprueba tu trabajo:



Cifrado y descifrado con safeDES: apartados 1 y 2.

Observación 1. Con safeDES es recomendable introducir el criptograma en hexadecimal, ya que el programa a propósito no guarda valores ASCII no imprimibles del criptograma, aunque lógicamente sí conserve esos bytes en el formato hexadecimal.
Si para descifrar introduces el criptograma en ASCII, comprueba que sólo puedes descifrar bien C2 pero no C1. Puedes comprobar mejor este efecto con este otro mensaje M3:
M3 = Una función hash es un algoritmo que transforma un conjunto arbitrario de elementos de datos, como puede ser un fichero de texto, en un único valor de longitud fija.
Vas a descifrar correctamente sólo si el criptograma lo introduces en hexadecimal. En cambio, si introduces el criptograma en ASCII sólo se descifrarán correctamente algunos bloques.
Observación 2. En el descifrado de texto, safeDES lo hace por líneas, lo cual no es del todo correcto ya que el texto descifrado tendrá estos dos caracteres ASCII añadidos 0D 0A y que significan 0D = CR = Retorno de carro y 0A = LF = Salto de línea.
Observa esos caracteres en la figura, justo después de la palabra "de ", al final del texto.



Observación del error en safeDES al cifrar por líneas en caja de texto.

No obstante, safeDES sí descifrará correctamente archivos. Compruébalo cifrando un archivo TXT o Word con varias líneas de texto y luego descifrando el criptograma.
Nota. Próximamente safeDES se actualizará como socDES, un nuevo software para prácticas con el algoritmo DES, realizado en Java, con mejores prestaciones y que incluye los modos de cifra ECB, CBC y CTR, además de un seguimiento (log) de la cifra.

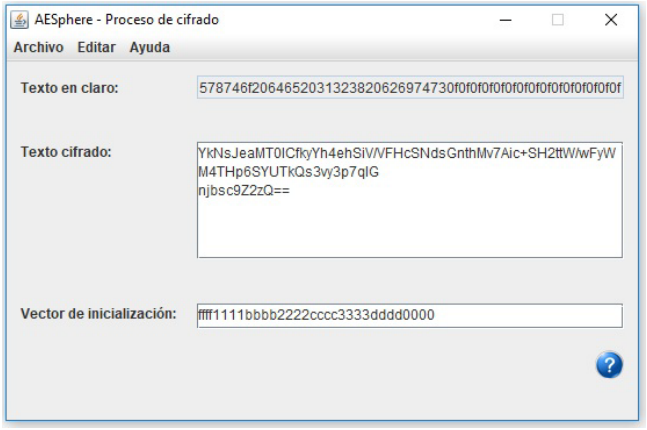
II. AES 128 modo CBC

Ejercicio 3)

3.1. Cifra de forma Directa el texto M1 con salida en Base 64
M1 = La cifra con AES usa bloques de texto de 128 bits
K = 0x 11223344556677889900AABBCCDDEEFF
IV = 0x FFFF1111BBBB2222CCCC3333DDDD0000

3.2. Observa el relleno que se indica en el texto en claro.

Comprueba tu trabajo:



Rellenos en AES.

Ejercicio 4)

4.1. Con las mismas claves K e IV, cifra de forma Directa M2 (se ha quitado la "s" en bits) de forma que el texto en claro tiene 48 caracteres = 384 bits, exactamente 3 bloques:
M2 = La cifra con AES usa bloques de texto de 128 bit
K = 0x 11223344556677889900AABBCCDDEEFF
IV = 0x FFFF1111BBBB2222CCCC3333DDDD0000

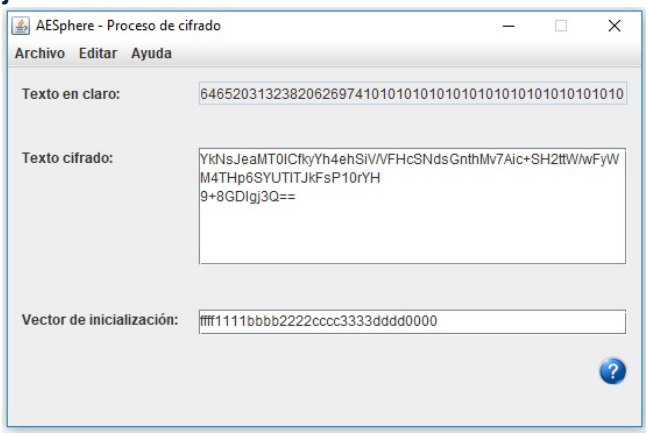
4.2. Observa el relleno que se indica ahora en el texto en claro.

4.3. ¿Por qué, a pesar de que la cifra es en modo CBC, excepto el último bloque del criptograma, los demás bloques son iguales en C1 y C2?

4.4. ¿Cuántos bloques se han cifrado en M1 y cuántos bloques han cifrado en M2?

4.5. ¿Por qué si M1 y M2 tienen diferentes tamaños, en C1 y C2 se observa en el código Base 64 dos signos =?

Comprueba tu trabajo:



Relleno en AES de un bloque completo.

III. AES 128 modo ECB

Ejercicio 5)

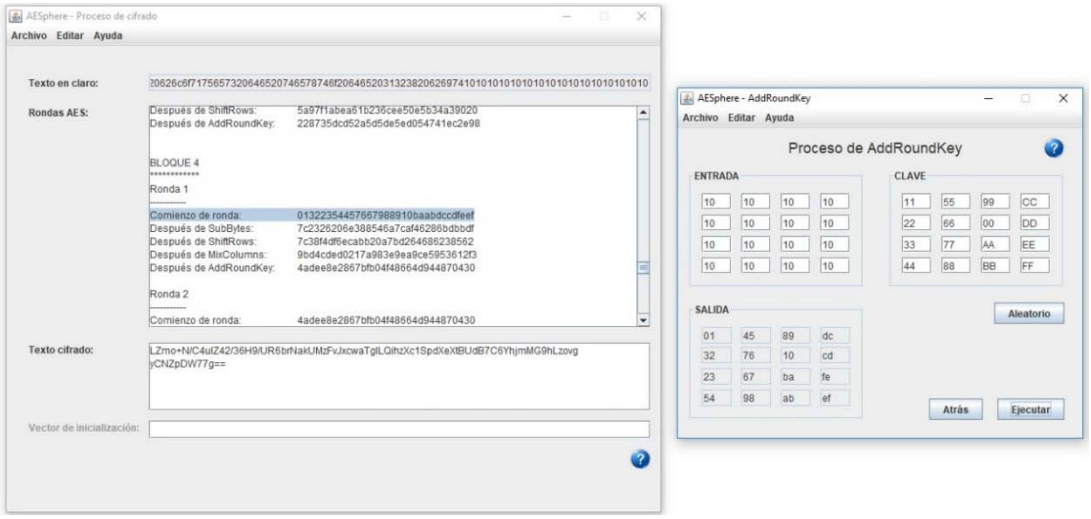
5.1. Cifra el texto anterior M2 en modo ECB pero ahora Paso a Paso:
M2 = La cifra con AES usa bloques de texto de 128 bit
K = 0x 11223344556677889900AABBCCDDEEFF

5.2. Observa que el relleno que se indica en el texto es el mismo que en el ejercicio 4.

5.3. Como el último bloque es todo relleno, comprueba que la entrada a la primera vuelta de ese 4to y último bloque de cifra después de AddRoundKey es: "*Comienzo de ronda: 01322354457667988910baabdccdfeef*". Para ello usa la opción Operaciones de AESPhere y calcula el AddRoundKey de estas dos matrices:
Matriz de estado: 0x 10101010101010101010101010101010
Clave k_0: 0x 11223344556677889900AABBCCDDEEFF

5.4. Según lo observado, ¿el relleno también se cifra?

Comprueba tu trabajo:



Cifra con AESPhere Paso a Paso y función AddRoundKey.