

Informe Laboratorio 4

Sección 3

Alumno Felipe Ormazábal
e-mail: felipe.ormazabal1@mail_udp.cl

Octubre de 2025

Índice

1. Descripción de actividades	2
2. Desarrollo de actividades según criterio de rúbrica	3
2.1. Investiga y documenta los tamaños de clave e IV	3
2.2. Solicita datos de entrada desde la terminal	4
2.3. Valida y ajusta la clave según el algoritmo	4
2.4. Implementa el cifrado y descifrado en modo CBC	5
2.5. Compara los resultados con un servicio de cifrado online	6
2.6. Describe la aplicabilidad del cifrado simétrico en la vida real	8
3. Referencias	8

1. Descripción de actividades

Desarrollar un programa en Python utilizando la librería pycrypto para cifrar y descifrar mensajes con los algoritmos DES, AES-256 y 3DES, permitiendo la entrada de la key, vector de inicialización y el texto a cifrar desde la terminal.

Instrucciones:

1. Investigación
 - Investigue y documente el tamaño en bytes de la clave y el vector de inicialización (IV) requeridos para los algoritmos DES, AES-256 y 3DES. Mencione las principales diferencias entre cada algoritmo, sea breve.
2. El programa debe solicitar al usuario los siguientes datos desde la terminal
 - Key correspondiente a cada algoritmo.
 - Vector de Inicialización (IV) para cada algoritmo.
 - Texto a cifrar.
3. Validación y ajuste de la clave
 - Si la clave ingresada es menor que el tamaño necesario para el algoritmo complete los bytes faltantes agregando bytes adicionales generados de manera aleatoria (utiliza `get_random_bytes`).
 - Si la clave ingresada es mayor que el tamaño requerido, trunque la clave a la longitud necesaria.
 - Imprima la clave final utilizada para cada algoritmo después de los ajustes.
4. Cifrado y Descifrado
 - Implemente una función para cada algoritmo de cifrado y descifrado (DES, AES-256, y 3DES). Use el modo CBC para todos los algoritmos.
 - Asegúrese de utilizar el IV proporcionado por el usuario para el proceso de cifrado y descifrado.
 - Imprima tanto el texto cifrado como el texto descifrado.
5. Comparación con un servicio de cifrado online
 - Selecciona uno de los tres algoritmos (DES, AES-256 o 3DES), ingrese el mismo texto, key y vector de inicialización en una página web de cifrado online.
 - Compare los resultados de tu programa con los del servicio online. Valide si el resultado es el mismo y fundamente su respuesta.
6. Aplicabilidad en la vida real

2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

- Describa un caso, situación o problema donde usaría cifrado simétrico. Defina que algoritmo de cifrado simétrico recomendaría justificando su respuesta.
- Suponga que la recomendación que usted entrego no fue bien percibida por su contraparte y le pide implementar hashes en vez de cifrado simétrico. Argumente cuál sería su respuesta frente a dicha solicitud.

2. Desarrollo de actividades según criterio de rúbrica

2.1. Investiga y documenta los tamaños de clave e IV

Algoritmo	Tamaño de Clave (Bits)	Tamaño de vector(IV) (Bits)
DES	56 + 8 (64)	64
3DES	128 o 192	64
AES-256	256 bits	128 bits

- DES: Usa una clave de solo 56 bits, es lento y ya esta obsoleto.
- 3DES: Usa una clave de 128 o 196 bits lo que lo hace un poco mas seguro, es mucho mas lento en comparación a DES debido a que aplica DES 3 veces, no esta obsoleto, pero no es recomendado.
- AES-256: Con una clave de 256 bits es mas seguros que los dos mencionados anteriormente, donde se hace uso de permutación, sustitución y mezcla, siendo también mas rápido

2.2. Solicitud datos de entrada desde la terminal

```
--- DES ---
Ingrese Key para DES: hola
Ingrese IV para DES: hola
Ingrese el texto a cifrar: hola
Key final (hex): 686f6c618fedaaa6
Texto cifrado (hex): fcc1ebaca252f6e8
Texto descifrado: hola

--- 3DES ---
Ingrese Key para 3DES: chao
Ingrese IV para 3DES: chao
Ingrese el texto a cifrar: chao
Key final (hex): 6368616fb49a3fed34de58ca605a8309bc2eb18885337d15
Texto cifrado (hex): 69ed4bcabdfad293
Texto descifrado: chao

--- AES-256 ---
Ingrese Key para AES-256: messi
Ingrese IV para AES-256: messi
Ingrese el texto a cifrar: messi
Key final (hex): 6d657373697c2db9845ae89f70471cfef55d5ca5c473408e0399f88e94cd2d8a
Texto cifrado (hex): b602da1c03a9a87c9a87946d07027da4
Texto descifrado: messi
```

Figura 1: Datos de entrada desde terminal

Como se puede ver en la figura 1 se piden las llaves, el vector de inicialización y el texto a cifrar para cada algoritmo en la terminal.

2.3. Valida y ajusta la clave según el algoritmo

Para ajustar la clave segun el algoritmo se llama a una función creada llamada `adjust_input` la cual se puede ver en la figura 2 donde dependiendo del algoritmo se le pasa el tamaño de clave que debería de tener y obviamente la clave ingresada, donde si es mas chica se rellena con bytes aleatorios y si es mas larga esta truncada.

```

6  def adjust_input(data_bytes, expected_size):
7      len_data = len(data_bytes)
8
9      if len_data == expected_size:
10         return data_bytes
11
12     if len_data < expected_size:
13         padding_needed = expected_size - len_data
14         random_padding = get_random_bytes(padding_needed)
15         return data_bytes + random_padding
16
17     if len_data > expected_size:
18         return data_bytes[:expected_size]
19

```

Figura 2: Funcion adjust_input

2.4. Implementa el cifrado y descifrado en modo CBC

Para el cifrado y descifrado con CBC se usa la función de la biblioteca de *pycryptodome* donde solo le pasamos la clase del algoritmo y ya hace el cifrado o descifrado correspondiente, esto se puede ver en la figura 3

```

38  cipher_encrypt = cipher_class.new(final_key, cipher_class.MODE_CBC, final_iv)
39  padded_text = pad(plaintext_bytes, block_size)
40  ciphertext = cipher_encrypt.encrypt(padded_text)
41  print(f"Texto cifrado (hex): {ciphertext.hex()}")
42
43
44  cipher_decrypt = cipher_class.new(final_key, cipher_class.MODE_CBC, final_iv)
45  decrypted_padded_text = cipher_decrypt.decrypt(ciphertext)
46  decrypted_text = unpad(decrypted_padded_text, block_size)
47

```

Figura 3: Cifrado y descifrado en modo cbc

Finalmente para correr el código en el main se llama a la función que tiene todo con cada algoritmo y sus especificaciones las cuales serian el tamaño de las llaves y vectores de inicialización en bytes.

```
def main():
    cifrar_descifrar_demo("DES", DES, 8, 8)
    cifrar_descifrar_demo("3DES", DES3, 24, 8)
    cifrar_descifrar_demo("AES-256", AES, 32, 16)
```

Figura 4: Main

2.5. Compara los resultados con un servicio de cifrado online

Para este espacio se uso la pagina <https://anycript.com/crypto/des> para cifrar online, donde se puede ver el resultado en la figura 5

The figure shows a screenshot of a web-based DES encryption tool. The interface is titled "DES Encryption". It has several input fields and settings:

- Encryption Text:** A text input field containing "hola".
- Secret Key:** A text input field containing "hola".
- Encryption Mode:** A dropdown menu with options "CBC" and "ECB", where "CBC" is selected.
- IV (optional):** A text input field containing "hola".
- Output format:** A dropdown menu with options "Base64" and "HEX", where "HEX" is selected.

On the right side, there is a large text area labeled "Encrypted Text" which contains the value "e4abfde6cb8b5fb".

Figura 5: Encriptación online

y el resultado del codigo hecho es el siguiente: figura 6

```
Ingrese Key para DES: hola
Ingrese IV para DES: hola
Ingrese el texto a cifrar: hola
Key final (hex): 686f6c611032e62b
Texto cifrado (hex): 56e9fa67688aa0d7
Texto descifrado: hola
```

Figura 6: Encriptación por código

Como se puede observar no dan el mismo resultado, esto es posible dado a que usé una llave y una IV de un tamaño mas chico del necesario haciendo que en el caso del código creado este lo rellena con bytes aleatorios, sin embargo la encriptación de la pagina online puede estar usando otro tipo de padding haciendo que de un texto cifrado distinto en cada algoritmo.

Para comprobar esto un poco mas voy a probar nuevamente el mismo texto a cifrar, pero ahora con una llave y IV de largo 8, los resultados en cada cifrado se pueden ver en las siguientes figuras 7 y 8

DES Encryption

Encryption Text	Encrypted Text
hola	78b4982ca2ca4509
Secret Key	
12345678	
Encryption Mode	
CBC	ECB
IV (optional)	
12345678	
Output format	
Base64	HEX

Figura 7: Encriptación online

```
--- DES ---
Ingrese Key para DES: 12345678
Ingrese IV para DES: 12345678
Ingrese el texto a cifrar: hola
Key final (hex): 3132333435363738
Texto cifrado (hex): 78b4982ca2ca4509
Texto descifrado: hola
```

Figura 8: Encriptación por código

Ahora como se puede ver en las imágenes el texto cifrado concuerda en ambos.

2.6. Describe la aplicabilidad del cifrado simétrico en la vida real

El cifrado simétrico en la vida real se puede usar mayormente en la protección de datos, generalmente de contraseñas un ejemplo de esto seria usar openssl y encriptar la contraseña con AES-256, igualmente esto va para cualquier tipo de mensajes que tienen que ser enviados a un destinatario sin que se sepa entremedio lo que se esta enviando

Conclusiones y comentarios

En este laboratorio se aprendió a crear, usar y ver el funcionamiento interno de algoritmos de cifrado simétricos en modo CBC, lo que puede venir muy bien para un futuro cuando se tenga que trabajar con encriptación de datos en la vida real.

3. Referencias

- <https://crypto.stackexchange.com/questions/76845/does-the-initialization-vector-in-des-have-to-be-8-bytes-long>
- <https://crypto.stackexchange.com/questions/24664/key-sizes-supported-by-3des>
- <https://stackoverflow.com/questions/31132162/what-size-of-initialization-vector-needed-for-aes-256-encryption-in-java>
- Repositorio: <https://github.com/Xxfelipe89xX/lab04-Cripto>