Ejercicios Criptografía Simétrica

Planteamos dos ejercicios para practicar la criptografía simétrica.

El primero será trabajar con la librería `hashlib` de Python para encriptar y luego usarlo para validar los textos encriptados.

El segundo usará la librería `cryptography` de Python para enviar y recibir mensajes encriptados.

Ejercicio 1 – Técnicas Hash

Dada la librería `hashlib`, realiza un sistema para controlar el acceso por usuario y contraseña. Se va a utilizar el algoritmo SHA256 para encriptar y se deben hacer las siguientes acciones:

- Crear un archivo de texto que contendrá los usuarios y las contraseñas, estás codificadas con SHA256. El archivo se llamara `passwd.in`

```
psanchis:1c42de962e046aea204f48be623ed4a300b6d71841755868be6ed5589e02cc5f
acalabuig:c9e8eb8098f36dba98f291da13f1fb6bc6ba7d6fa874794c669d749fe869aab6
jpolit:d8c352877624b34d8f0e5b0410921d97403460d9927f84aae28eba2eaf523120
```

- Para codificar las contraseñas se ha usado un programa de encriptación sha256. Y se han guardado en el fichero de contraseñas con el formato que se ve.
- Crea un programa servidor que permita validar usuarios.
  - Cuando se conecte un cliente creará un hilo.
  - El servidor solicitará al cliente un nombre de usuario.
  - Cuando reciba la contestación del cliente, solicitara una contraseña.
  - Cuando reciba la contraseña comprobará en el fichero `passwd.in` si el usuario y la contraseña son correctas.
  - El servidor contestará con un error si no coicide la contraseña o con un "Acceso concedido" si la contraseña es correcta.

Ejercicio 2 – Criptografía AES

Dada la librería cryptography, realizad la instalación en vuestro entorno de trabajo para poder hacer uso de ella.

```
$ pip install cryptography
```

Usaremos la clase `Fernet` de la librería `cryptography`. Ésta implementa el cifrado simétrico AES en el modo cifrado de bloques (CBC) y una función de autenticación de mensajes (HMAC)

Ejemplo de uso de la clase Fernet:

```
from cryptography.fernet import Fernet

# Generación de la clave de cifrado
key = Fernet.generate_key()
print("Clave:", key)

# Creación de un objeto Fernet con la clave
cipher_suite = Fernet(key)

# Texto para ser cifrado
text = "Mensaje secreto".encode()

# Cifrado del texto
cipher_text = cipher_suite.encrypt(text)
print("Texto cifrado:", cipher_text)

# Descifrado del texto
plain_text = cipher_suite.decrypt(cipher_text)
print("Texto descifrado:", plain_text.decode())
```

Crea un programa que permita encriptar y desencriptar usando una clave simétrica

Parámetro de entrada:

-e (encode) encripta una cadena de caracteres en texto plano

-d (decode): desencripta según el parámetro.

En primer lugar, se deberá generar una clave para codificar y decodificar (que debe ser la misma). Esta clave se debe almacenar en un fichero para poder consultarla tanto para encriptar un texto como para desencriptar.

El programa deberá comprobar que existe un fichero con clave para no generar otra. Si esta comprobación no existe, se podría dar el caso de que cree una clave para codificar y después no podamos descodificar un mensaje o a la inversa.