

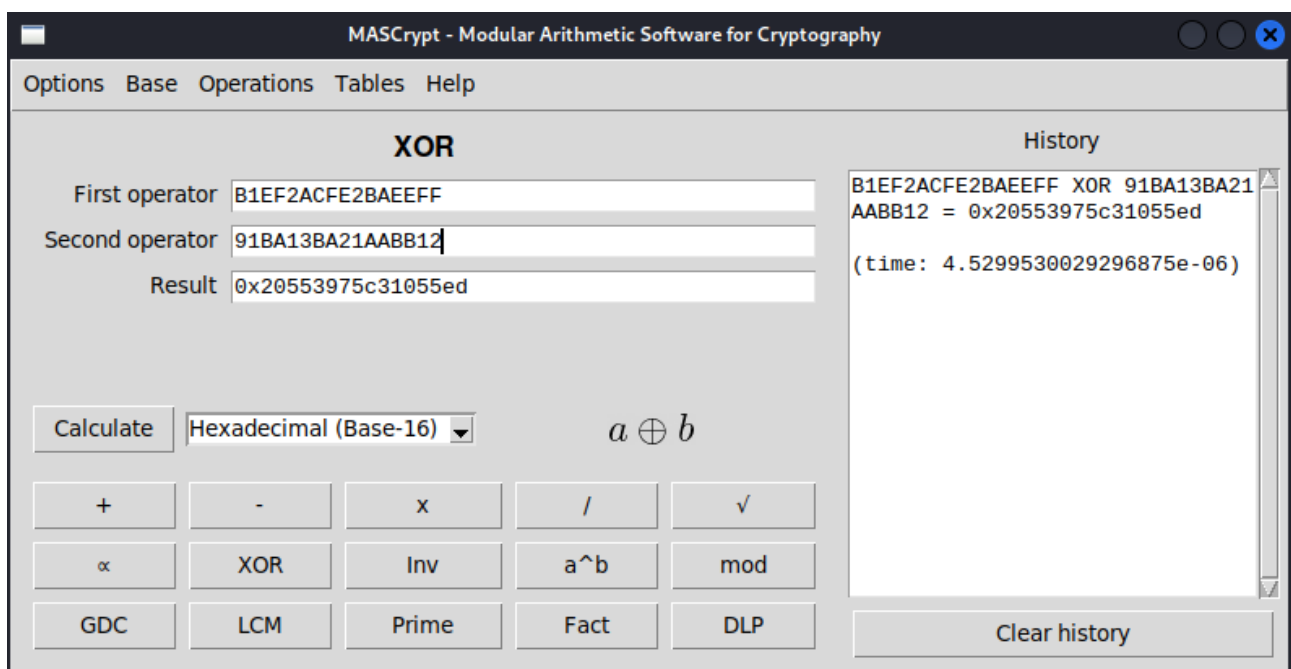
PRACTICA CRIPTOGRAFÍA

Ejercicios:

1. Tenemos un sistema que usa claves de 16 bytes. Por razones de seguridad vamos a proteger la clave de tal forma que ninguna persona tenga acceso directamente a la clave. Por ello, vamos a realizar un proceso de disociación de la misma, en el cuál tendremos, una clave fija en código, la cual, sólo el desarrollador tendrá acceso, y otra parte en un fichero de propiedades que rellenará el Key Manager. La clave final se generará por código, realizando un XOR entre la que se encuentra en el properties y en el código.

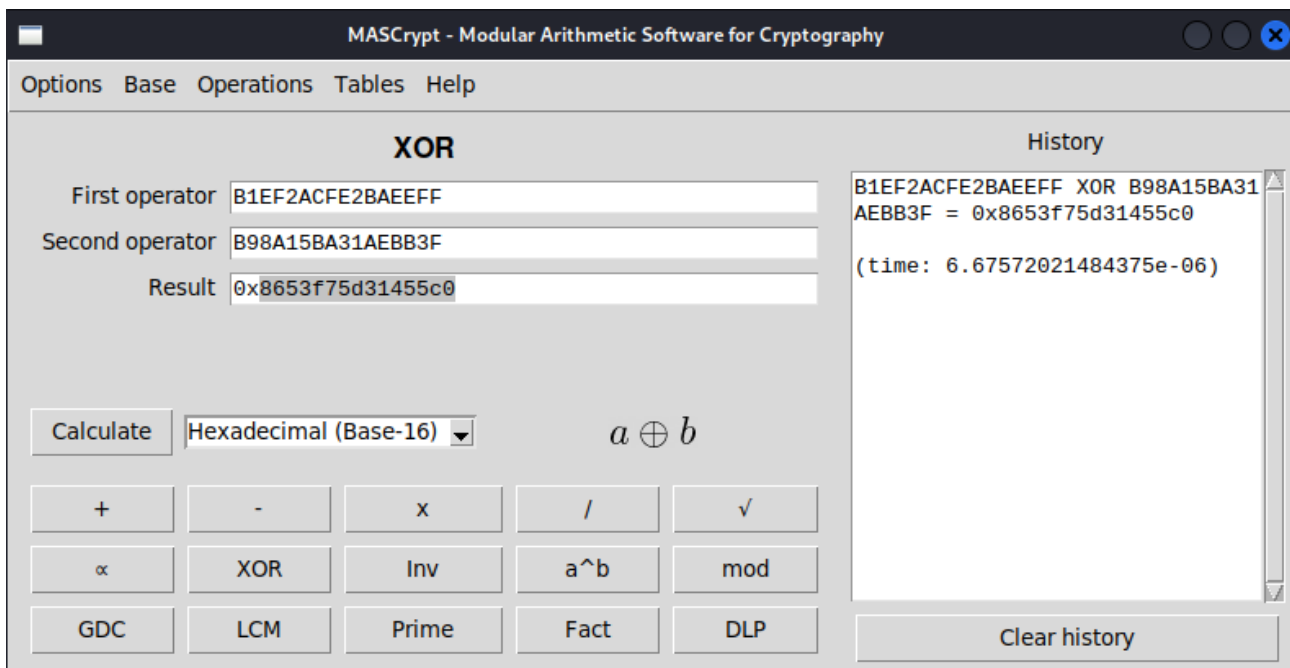
La clave fija en código es **B1EF2ACFE2BAEEFF**, mientras que en desarrollo sabemos que la clave final (en memoria) es **91BA13BA21AABB12**.

¿Qué valor ha puesto el Key Manager en properties para forzar dicha clave final?



La clave fija, recordemos es **B1EF2ACFE2BAEEFF**, mientras que en producción sabemos que la parte dinámica que se modifica en los ficheros de propiedades es **B98A15BA31AEBB3F**.

¿Qué clave será con la que se trabaje en memoria?



2. Dada la clave con etiqueta “cifrado-sim-aes-256” que contiene el keystore. El iv estará compuesto por el hexadecimal correspondiente a ceros binarios (“00”). Se requiere obtener el dato en claro correspondiente al siguiente dato cifrado:

**TQ9SOMKc6aFS9SlxhfK9wT18UXpPCd505Xf5J/5nLI7Of/
o0QKIWXg3nu1RRz4QWElezdrLAD5LO4US
t3aB/i50nvvJbBiG+le1ZhpR84oI=**

Para este caso, se ha usado un AES/CBC/PKCS7. Si lo desciframos, ¿qué obtenemos?

Obtenemos el texto en claro:

Esto es un cifrado en bloque típico. Recuerda, vas por el buen camino.
Animo

¿Qué ocurre si decidimos cambiar el padding a x923 en el descifrado?

Cuando cambiamos el formato padding de pkcs7 a x923 cambiaria en en el hexadecimal en este caso el padding es uno

¿Cuánto padding se ha añadido en el cifrado?

3. Se requiere cifrar el texto “KeepCoding te enseña a codificar y a cifrar”. La clave para ello, tiene la etiqueta en el Keystore “cifrado-sim-chacha20-256”. El nonce “9Yccn/f5nJJhAt2S”. El algoritmo que se debe usar es un Chacha20.

¿Cómo podríamos mejorar de forma sencilla el sistema, de tal forma, que no sólo garanticemos la confidencialidad sino, además, la integridad del mismo?

Para hacerlo mas seguro deberiamos utilizar un nonce random y un chacha20poly por que utiliza un autentificador

Se requiere obtener el dato cifrado, demuestra, tu propuesta por código, así como añadir los datos necesarios para evaluar tu propuesta de mejora.

ejercicio 3

```
from Crypto.Cipher import ChaCha20
from base64 import b64decode, b64encode
from Crypto.Random import get_random_bytes

textoPlano_bytes = bytes('KeepCoding te enseña a codificar y a cifrar', 'UTF-8')
clave =
bytes.fromhex('AF9DF30474898787A45605CCB9B936D33B780D03CABC81719D52383480DC3120'
)
print(clave.hex())
nonce_mensaje = b64decode("9Yccn/f5nJJhAt2S")
cipher = ChaCha20.new(key=clave, nonce=nonce_mensaje)
```

```

texto_cifrado = cipher.encrypt(textoPlano_bytes)
print('Mensaje cifrado en HEX = ', texto_cifrado.hex() )
print('Mensaje cifrado en B64 = ', b64encode(texto_cifrado).decode('utf-8'))

decipher = ChaCha20.new(key=clave, nonce=nonce_mensaje)
plaintext = decipher.decrypt(texto_cifrado)
print('Mensaje en claro = ',plaintext.decode('utf-8'))

# Respuesta
# af9df30474898787a45605ccb9b936d33b780d03cab81719d52383480dc3120
# Mensaje cifrado en HEX =
69ac4ee7c4c552537a00a19bcdf7f0aaed7c9c8f769956a09bce6fadef6c3535f2211c9467067cf5
c4a842ab
# Mensaje cifrado en B64 =
aaxO58TFUln6AKGbyvfwqu18nI92mVagm85vre9sNTXyIRyUZwZ89cSoQqs=
# Mensaje en claro = KeepCoding te enseña a codificar y a cifrar

# Seria mas seguro utilizando un nonce random y con un chacha20poly por que
utiliza un autentificador

```

4. Tenemos el siguiente jwt, cuya clave es “Con KeepCoding aprendemos”.

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c3VhcmlvIjoibG9uIFBlcGl0byBkZSBsb3MgcGFsb3RlcyIsInJvbCI6ImIzTm9ybWFsIiwiaWF0IjoxNjY3OTMzNTMzZm90ZDxp6oixMLXXRP97W4TDTrv0y7B5YjD0U8ixrE

¿Qué algoritmo de firma hemos realizado?
HS256

¿Cuál es el body del jwt?

Usuario: Don Pepito de los palotes
Rol: isNormal
Iat: 1667933533

Un hacker está enviando a nuestro sistema el siguiente jwt:

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c3VhcmlvIjoibG9uIFBlcGl0byBkZSBsb3MgcGFsb3RlcyIsInJvbCI6ImIzQWRtaW4iLCJpYXQiOiJlY2N

jc5MzM1MzN9.krgBkzCBQ5WZ8JnZHuRvmnAZdg4ZMeRNv2CIAODIHRI

¿Qué está intentando realizar?

El hacker estaria queriendo recolectar informacion para escalar privilegios

¿Qué ocurre si intentamos validarlo con pyjwt?

Da fallo en la verificacion de la firma

5. El siguiente hash se corresponde con un SHA3 Keccak del texto “En KeepCoding aprendemos cómo protegernos con criptografía”.

bced1be95fbd85d2ffcce9c85434d79aa26f24ce82fbd4439517ea3f072d56fe

¿Qué tipo de SHA3 hemos generado?

Generamos un SHA3-256 por la longitud del codigo

Y si hacemos un SHA2, y obtenemos el siguiente resultado:

4cec5a9f85dcc5c4c6ccb603d124cf1cdc6dfe836459551a1044f4f2908aa5d63739506f6468833d77c07cfd69c488823b8d858283f1d05877120e8c5351c833

¿Qué hash hemos realizado?

Un SHA2-512

Genera ahora un SHA3 Keccak de 256 bits con el siguiente texto: “En KeepCoding aprendemos cómo protegernos con criptografía.”

¿Qué propiedad destacarías del hash, atendiendo a los resultados anteriores?

Destacaríamos que un mínimo cambio en un byte en este caso el punto del final del texto produciera un cambio total en el hash

6. Calcula el hmac-256 (usando la clave contenida en el Keystore) del siguiente texto:

Siempre existe más de una forma de hacerlo, y más de una solución válida. Se debe evidenciar la respuesta. Cuidado si se usan herramientas fuera de los lenguajes de programación, por las codificaciones es mejor trabajar en hexadecimal.

El HMAC sería

857d5ab916789620f35bcfe6a1a5f4ce98200180cc8549e6ec83f408e8ca0550

7. Trabajamos en una empresa de desarrollo que tiene una aplicación web, la cual requiere un login y trabajar con passwords. Nos preguntan qué mecanismo de almacenamiento de las mismas proponemos.

Tras realizar un análisis, el analista de seguridad propone un hash SHA-1. Su responsable, le indica que es una mala opción. ¿Por qué crees que es una mala opción?

El sha-1 es un error por que se por que ya esta vulnerado ya que se descubrieron varios ataques como los ataques de colision, esto ocurre cuando dos entradas diferentes generan el mismo hash. Esto lo hace mas propenso a ataques mas practicos en menor tiempo

Después de meditarlo, propone almacenarlo con un SHA-256, y su responsable le pregunta si no lo va a fortalecer de alguna forma. ¿Qué se te ocurre?

Añadiríamos un Salt

Parece que el responsable se ha quedado conforme, tras mejorar la propuesta del SHA-256, no obstante, hay margen de mejora. ¿Qué propondrías?

Utilizaríamos un HMAC para combinar el salt con la clave maestra antes de realizar el hash. Esto agrega otra capa de seguridad y autenticación al proceso de derivación de claves.

8. ¿Qué algoritmos usarías?

La mejor manera de conseguir todas las cosas mencionadas es con algoritmos que nos brinden un tag o mac. Con esto podemos garantizar la confidencialidad e integridad de los mensajes.

Un AES/GCM para criptografía simétrica en bloque o un CHACHA20POLY1305 para criptografía en flujo sería lo que se utilizaría

9. Se requiere calcular el KCV de la siguiente clave AES:

**A2CFF885901A5449E9C448BA5B948A8C4EE377152B3F1ACFA0148
FB3A426DB72**

Para lo cual, vamos a requerir el KCV(SHA-256) así como el KCV(AES). El KCV(SHA-256) se corresponderá con los 3 primeros bytes del SHA-256. Mientras que el KCV(AES) se corresponderá con cifrar un texto del tamaño del bloque AES (16 bytes) compuesto con ceros binarios (00), así como un iv igualmente compuesto de ceros binarios. Obviamente, la clave usada será la que queremos obtener su valor de control.

KCV: 5244db

KCV: 5244db

KCV SHA256: db7df2

10. El responsable de Raúl, Pedro, ha enviado este mensaje a RRHH:

Se debe ascender inmediatamente a Raúl. Es necesario mejorarle sus condiciones económicas un 20% para que se quede con nosotros.

Lo acompaña del siguiente fichero de firma PGP (MensajeRespoDeRaulARRHH.txt.sig). Nosotros, que pertenecemos a RRHH vamos al directorio a recuperar la clave para verificarlo. Tendremos los ficheros Pedro-priv.txt y Pedro-publ.txt, con las claves privada y pública.

Las claves de los ficheros de RRHH son RRHH-priv.txt y RRHH-publ.txt que también se tendrán disponibles.

Se requiere verificar la misma, y evidenciar dicha prueba.

Así mismo, se requiere firmar el siguiente mensaje con la clave correspondiente de las anteriores, simulando que eres personal de RRHH.

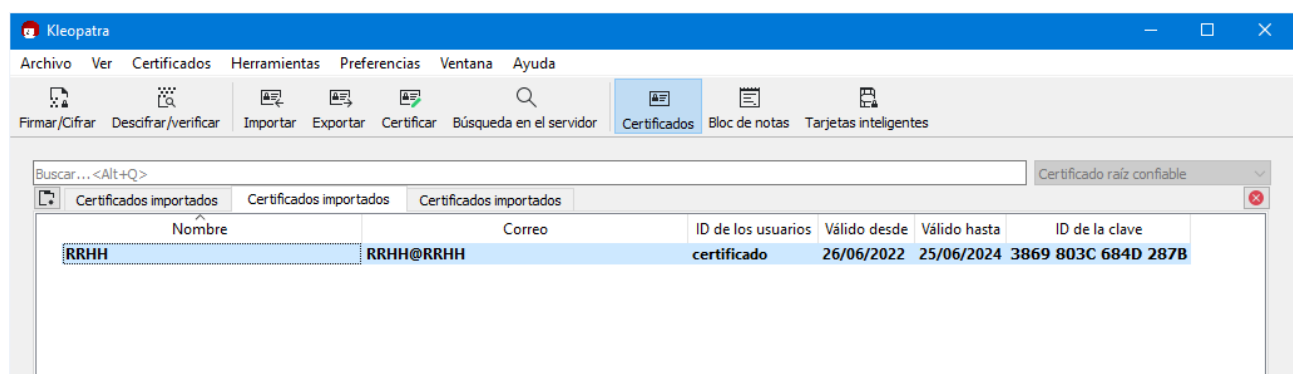
Viendo su perfil en el mercado, hemos decidido ascenderle y mejorarle un 25% su salario. Saludos.

Por último, cifra el siguiente mensaje tanto con la clave pública de RRHH como la de Pedro y adjunta el fichero con la práctica.

Estamos todos de acuerdo, el ascenso será el mes que viene, agosto, si no hay sorpresas.

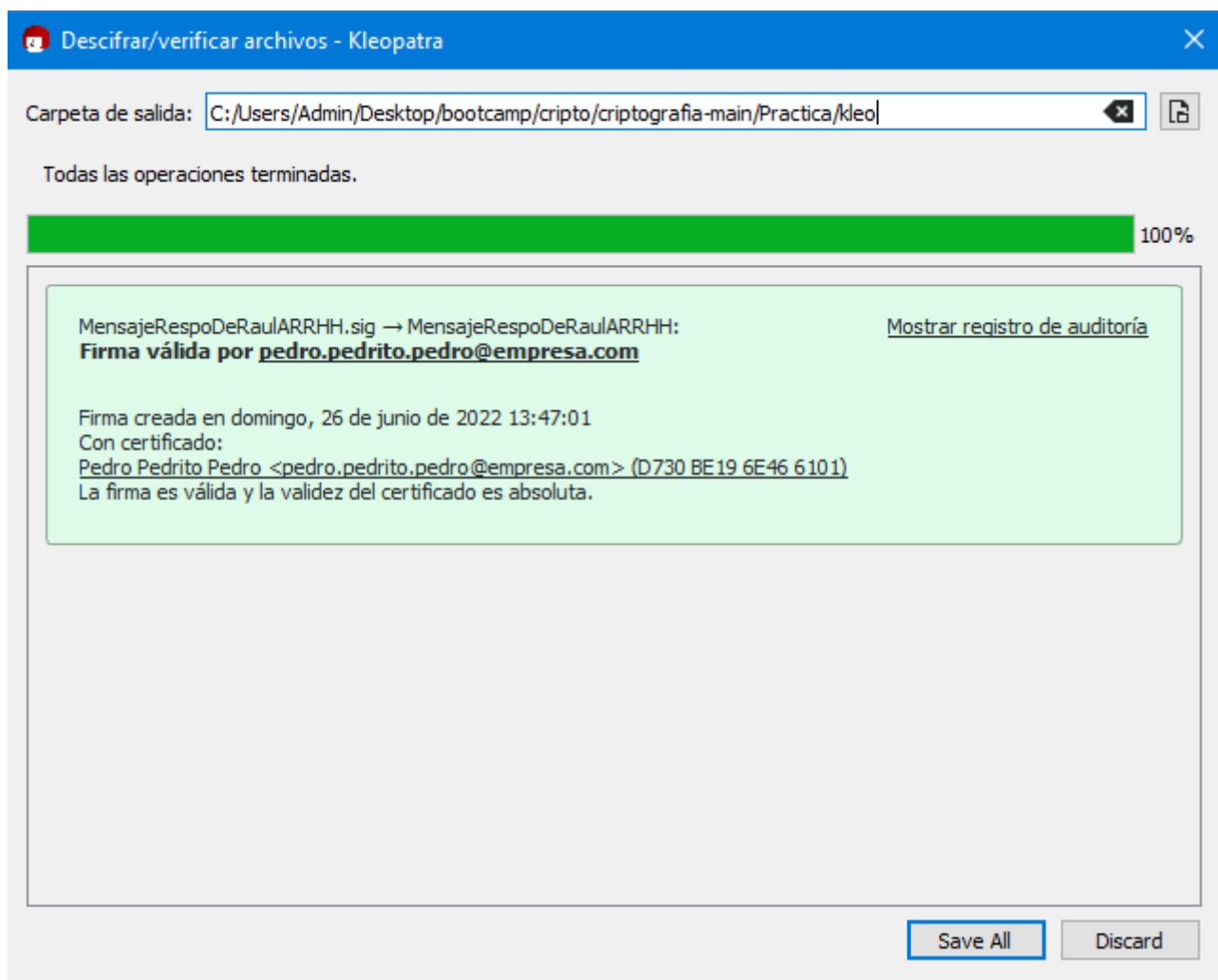
Para realizar este ejercicio hemos utilizado un programa llamado kleopatra para automatizar los procesos

Primero hemos importados las claves publicas y privadas de los archivos del que nos proporciona el ejercicio

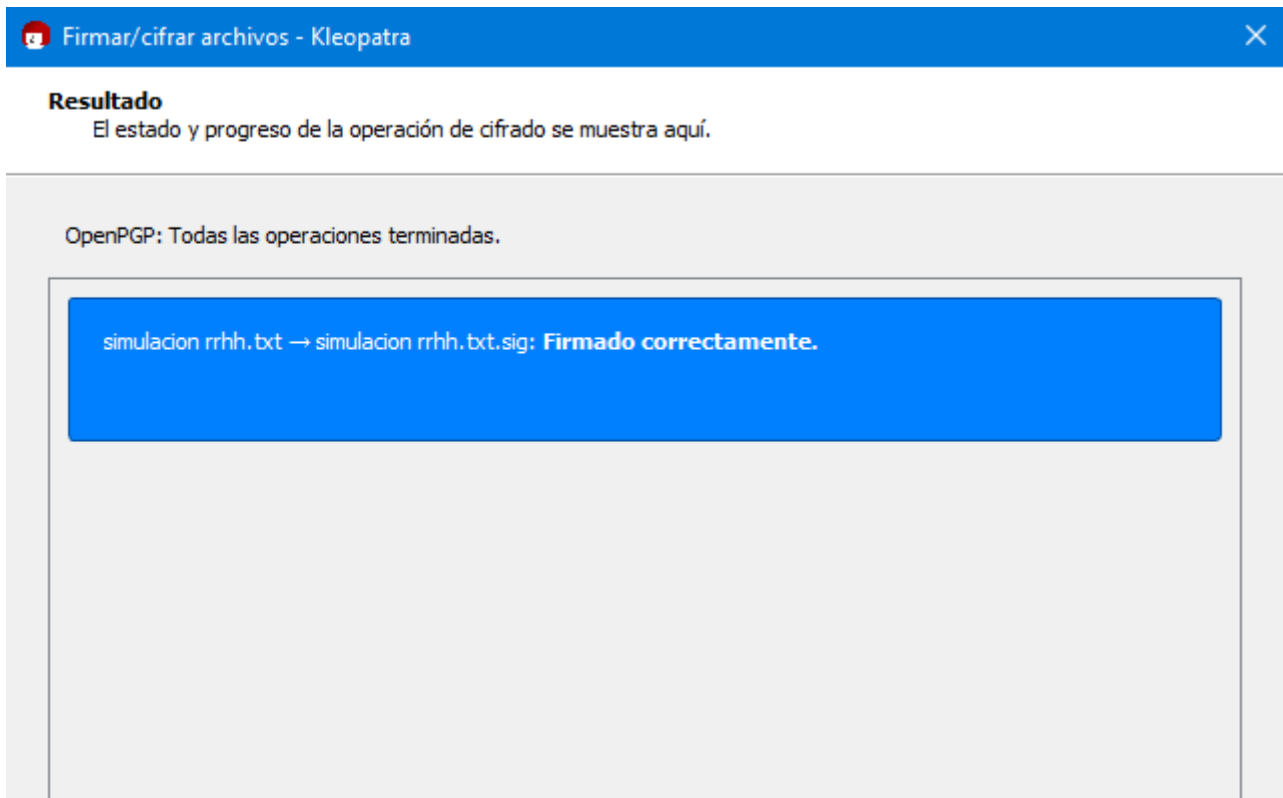




luego hemos descifrado y verificado el mensaje con la clave de rhh



proseguiremos con el siguiente punto del punto del ejercicio.
Ahora firmaremos el siguiente archivo como si fuéramos rrhh



y por ultimo cifraremos el ultimo mensaje con las claves publicas de rrhh y pedro

Firmar/cifrar archivos - Kleopatra


Firmar o cifrar archivos



Probar autenticidad (firmar)

☐ Firmar como: ✓ RRHH <RRHH@RRHH> (certificado, creado: 26/06/2022)

Cifrar

☒ Cifrar para mí: ✓ Pedro Pedrito Pedro <pedro.pedrito.pedro@empresa.com> (certificado, creado: 26/06/2022)




☒ Cifrar para otros: ✓ RRHH <RRHH@RRHH> (certificado, OpenPGP, creado: 26/06/2022) 

 Por favor, introduzca un nombre o dirección de correo... 

☐ Cifrar con contraseña. Cualquier persona con la que comparta la contraseña podrá ver los datos.

Salida

Archivos/carpeta de salida:

 C:/Users/Admin/Desktop/bootcamp/cripto/criptografia-main/Practica/kleo/cifrado para los dos.txt.gpg  

☐ Cifrar o firmar cada archivo por separado.

Cifrar Cancel

Adjunto todos los archivos que pide el ejercicio en la carpeta llamada Kleopatra

11. Nuestra compañía tiene un contrato con una empresa que nos da un servicio de almacenamiento de información de videollamadas. Para lo cual, la misma nos envía la clave simétrica de cada videollamada cifrada usando un RSA-OAEP. El hash que usa el algoritmo interno es un SHA-256.

El texto cifrado es el siguiente:

**b72e6fd48155f565dd2684df3ffa8746d649b11f0ed4637fc4c99d18283b32
e1709b30c96b4a8a20d5dbc639e9d83a53681e6d96f76a0e4c279f0dffa76
a329d04e3d3d4ad629793eb00cc76d10fc00475eb76bfbcb1273303882609
957c4c0ae2c4f5ba670a4126f2f14a9f4b6f41aa2edba01b4bd586624659fc
a82f5b4970186502de8624071be78ccef573d896b8eac86f5d43ca7b10b59
be4acf8f8e0498a455da04f67d3f98b4cd907f27639f4b1df3c50e05d5bf63
768088226e2a9177485c54f72407fdf358fe64479677d8296ad38c6f177ea7
cb74927651cf24b01dee27895d4f05fb5c161957845cd1b5848ed64ed3b03
722b21a526a6e447cb8ee**

Las claves pública y privada las tenemos en los ficheros clave-rsa-oaep-
publ.pem y clave-rsaoaep-priv.pem.

Si has recuperado la clave, vuelve a cifrarla con el mismo algoritmo. ¿Por
qué son diferentes los textos cifrados?

Son diferentes por el padding generado es un valor aleatorio y eso le da
variabilidad en cada cifrado

12. Nos debemos comunicar con una empresa, para lo cual, hemos
decidido usar un algoritmo como el AES/GCM en la comunicación.
Nuestro sistema, usa los siguientes datos en cada comunicación con el
tercero:

**Key:E2CFF885901B3449E9C448BA5B948A8C4EE322152B3F1ACFA
0148FB3A2 6DB74**

Nonce:9Yccn/f5nJJhAt2S

¿Qué estamos haciendo mal?

El nonce debería ser un valor aleatorio

Cifra el siguiente texto:

He descubierto el error y no volveré a hacerlo mal

**Xcu2Jh0PuinOOUMemgE7NMvKKk4Euy2QFJ1h9K/
QTWXiq92dhLum64MHC V9QePv8FiVt**

Usando para ello, la clave, y el nonce indicados. El texto cifrado presentalo en hexadecimal y en base64.

5dcbb6261d0fba29ce39431e9a013b34cbca2a4e04bb2d90149d61f4afd04d65e2abdd9d84bba6eb8307095f5078fbfc16256d

13. Se desea calcular una firma con el algoritmo PKCS#1 v1.5 usando las claves contenidas en los ficheros clave-rsa-oaep-priv y clave-rsa-oaep-publ.pem del mensaje siguiente:

El equipo está preparado para seguir con el proceso, necesitaremos más recursos.

¿Cuál es el valor de la firma en hexadecimal?

Calcula la firma (en hexadecimal) con la curva elíptica ed25519, usando las claves ed25519- priv y ed25519-publ.

14. Necesitamos generar una nueva clave AES, usando para ello una HKDF (HMAC-based Extractand-Expand key derivation function) con un hash SHA-512. La clave maestra requerida se encuentra en el keystore con la etiqueta “cifrado-sim-aes-256”. La clave obtenida dependerá de un identificador de dispositivo, en este caso tendrá el valor en hexadecimal:

e43bb4067cbcfab3bec54437b84bef4623e345682d89de9948fbb0afedc461a3

¿Qué clave se ha obtenido?

Clave Cifrado:

e716754c67614c53bd9bab176022c952a08e56f07744d6c9edb8c934f52e448a

Con este algoritmo podriamos generar mas de una clave pero el ejercicio solo nos pedia 1

15. Nos envían un bloque TR31:

**D0144D0AB00S000042766B9265B2DF93AE6E29B58135B77A2F616C8D515ACDB
E6A5626F79FA7B4071E9EE1423C6D7970FA2B965D18B23922B5B2E5657495E0 3CD857FD37018E111B**

Donde la clave de transporte para desenvolver (unwrap) el bloque es:

A1A101010101010101010101010101010102

¿Con qué algoritmo se ha protegido el bloque de clave?

algoritmo de tipo AES

¿Para qué algoritmo se ha definido la clave?

algoritmo de tipo AES

¿Para qué modo de uso se ha generado?

Se genero la clave para cifrar y descifrar

¿Es exportable? ¿Para qué se puede usar la clave?

Si pero es sensible, exportable solo bajo clave no confiable

¿Qué valor tiene la clave?

c1c1c1c1c1c1c1c1c1c1c1c1c1c1c1c1

