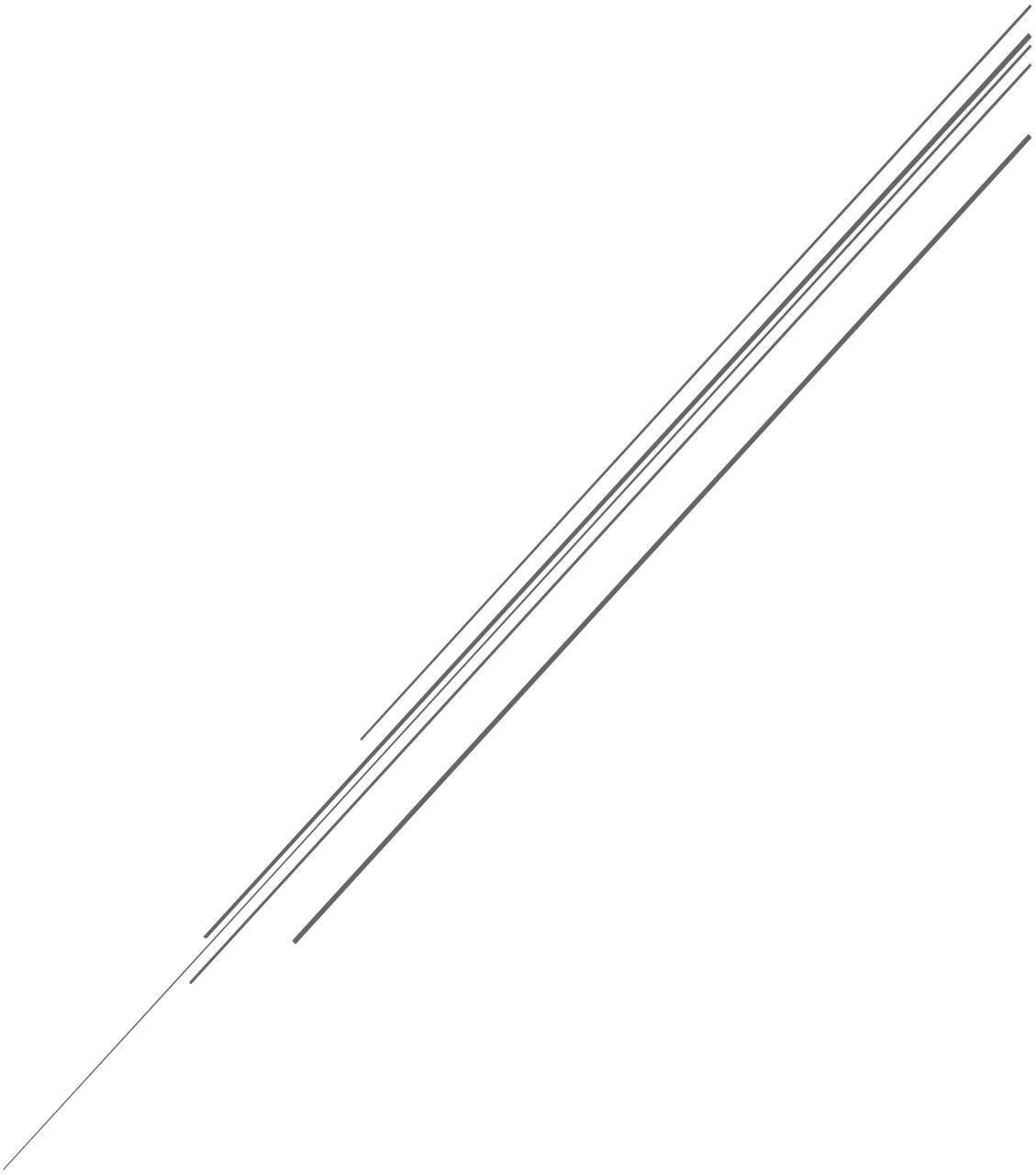
ACTIVIDAD 6: CRIPTOGRAFÍA DE CLAVE PÚBLICA CON OPENSSL

Seguridad Informática



Escuela Técnica Superior de Ingeniería
Grado en Ingeniería Informática

CONTENIDO

1. GENERACIÓN DE CLAVES

a. BREVE INTRODUCCIÓN A RSA

En esta parte de la sesión, se entró en contacto por primera vez con el sistema criptográfico RSA, cuyas iniciales corresponden a los apellidos de los creadores del mismo (Rivers, Shamir, Adleman). Este es uno de los primeros sistemas de cifrado asimétrico utilizados y se sigue utilizando para transmisión segura de datos.

En RSA la asimetría de las claves se basa en la dificultad de la factorización del producto de dos primos muy grandes. Cuando se utiliza RSA, se crea una clave pública, que se basa en dos primos considerables y un valor auxiliar, y una clave privada. La clave pública la puede utilizar cualquier persona para encriptar un mensaje, pero únicamente alguien con conocimiento de la clave privada puede desencriptar fácilmente el mensaje.

Uno de los principios básicos de RSA es que no es excesivamente complejo encontrar tres enteros positivos grandes, *e*, *d*, y *n* tal que, con exponenciación modular, para todo *m*:

$$(m^e)^d \equiv m \ (mod\ n)$$

Para este módulo, incluso conociendo *e* y *n* o *m*, puede ser extremadamente difícil encontrar *d*. El cifrado se realiza de forma tal que un mensaje P se encripta por C sería algo así:

$$C = P^e mod\ n$$

Y se desencripta tal que así:

$$P = C^d mod\ n$$

La clave de encriptación está compuesta por dos enteros (*e*, *n*) y la clave para desencriptar también (*d*, *n*). El valor de *n* es lo que permite iniciar la selección de claves. El valor de *n* debería ser muy grande, siendo el producto de dos primos: *p* y *q*. Estos dos primos, habitualmente, también son muy grandes. Un valor grande de *n* causa que factorizar *p* y *q* sea una tarea compleja, pero el tiempo de encriptado aumenta según el valor de *n* crece.

b. CREACIÓN DE PAR DE CLAVES RSA

Para iniciar la sesión, se han creado un par de claves (privada y pública) RSA de 1024 bits. Se ha utilizado el siguiente comando:

```
openssl genrsa -out parclaves.pem 1024
```

Donde *genrsa* indica que se desea crear una clave pública y una privada RSA, *-out parclaves.pem* indica dónde se desea guardar esta información y *1024* el tamaño de las claves. Para obtener la información de las claves almacenadas en el fichero, se ha ejecutado el comando:

```
openssl rsa -in parclaves.pem -text -noout
```

Se ha obtenido la siguiente información:

```
Private-Key: (1024 bit)
modulus:
00:ac:18:1a:89:b2:52:0c:35:b9:60:88:4a:4a:23:
25:8b:72:cf:31:cd:f6:b1:d7:51:ad:a5:27:d2:5c:
```

```
95:b9:0b:94:d5:53:ec:1d:9b:16:98:45:ce:67:ac:
d0:cd:b3:cb:9c:34:02:bd:01:fa:63:83:c4:3b:2a:
db:a4:28:9c:5b:e6:18:0f:67:6e:6d:ba:d1:b6:09:
33:06:bb:dd:fb:7c:41:a5:f4:dc:58:7a:5f:73:50:
fc:3d:06:07:e2:bf:9d:4e:4e:5f:94:28:70:32:52:
9b:88:6d:9a:b7:30:99:02:a6:68:46:a6:d5:83:3b:
ef:47:44:85:55:de:2b:09:65
publicExponent: 65537 (0x10001)
privateExponent:
19:38:cb:22:e5:2c:9b:37:80:7c:c2:5c:c8:f2:cf:
bc:d9:4d:be:89:e6:1e:f8:64:b3:23:62:6b:b5:40:
08:47:c7:3f:60:b6:59:3c:72:9a:4c:98:cc:9f:0a:
57:2b:83:aa:d8:00:92:bc:e8:7a:8f:44:42:3b:ac:
c7:29:60:88:e5:c2:46:65:d1:73:e1:e5:ba:fb:a9:
8b:d9:5c:aa:82:77:19:d0:2d:d6:1a:9f:a1:98:5f:
66:bc:a2:73:de:8a:4e:9a:8a:79:fb:ce:e6:fa:8a:
05:71:73:85:96:95:46:d4:0d:d1:f8:cd:2c:0e:c7:
84:ed:1b:f5:b1:cd:b2:1d
prime1:
00:e2:a7:45:0c:98:7c:90:f2:ed:f1:fc:50:0a:cb:
9b:02:78:37:c2:6a:db:05:54:59:f2:6b:66:d4:a1:
07:1b:05:e5:7d:40:8d:29:31:71:41:d1:2e:50:4d:
25:e9:60:d1:ee:83:89:f5:3e:58:2f:a8:1b:a4:46:
f1:be:1b:5f:a3
prime2:
00:c2:60:65:12:5e:eb:2f:1a:c0:e9:1d:f0:87:da:
54:58:5f:e7:97:d0:2e:0b:da:14:2c:08:3c:fd:e7:
4f:9c:01:29:f2:6f:5f:1a:83:93:cc:22:03:74:23:
82:22:f9:ad:b6:09:84:69:c1:e5:75:76:0f:7d:92:
4c:9f:b0:e3:57
exponent1:
41:86:c0:64:b2:d5:18:86:d3:19:e9:ab:2e:63:cc:
cd:ba:f9:cb:e0:5e:af:bc:c2:40:5e:a2:9d:08:6c:
e8:78:1c:ef:c8:30:c7:5e:fe:f2:5b:4a:7b:76:c2:
66:25:52:ac:15:25:56:5c:8e:dc:40:4a:b5:84:b9:
31:0f:f2:e7
exponent2:
4d:ff:54:4f:6c:db:3e:c9:a3:83:67:ca:2f:19:83:
22:fb:48:f9:78:46:21:e7:5e:94:d4:b1:74:c9:2e:
fe:d0:d0:be:41:c6:8d:e3:22:99:95:44:81:84:06:
22:76:c2:27:fb:e5:b0:72:67:db:1c:86:d6:c2:b0:
```

10:9c:70:7d **coefficient:** **00:cc:82:8f:d9:6a:43:85:3c:8e:53:00:6e:80:92:** **97:e5:30:a6:7f:59:4d:b6:0b:11:0a:07:cb:d6:00:** **36:8e:15:33:12:f0:e0:97:56:19:76:e6:69:97:c0:** **29:33:6a:6d:63:73:23:7b:46:9a:71:72:4d:e6:69:** **04:c2:69:4b:b4**
Tabla I: información de las claves almacenadas

c. ANÁLISIS DEL PAR DE CLAVES

A continuación, se analizaron los campos más importantes que se obtuvieron a partir del comando utilizado para conseguir información de las claves. Estos campos son *modulus*, *publicExponent*, *privateExponent*, *prime1*, *prime2*.

- *modulus*: es el módulo utilizado por ambas claves: tanto la privada como la clave pública. Sería la *n* utilizada en la fórmula explicada en el apartado *I.a*.
- *publicExponent*: es el exponente público. Sería la *e* en la fórmula explicada en el apartado *I.a*.
- *privateExponent*: es el exponente privado. Sería la *d* en la fórmula explicada en el apartado *I.a*.
- *prime1*: es uno de los números primos que da lugar a *n, p*.
- *prime2*: es el otro primo, cuyo producto con *p* da lugar a n, *q*.

Todos estos campos permiten que se pueda encriptar y desencriptar un mensaje sin necesidad de realizar ninguna acción aparte de acceder a *parclaves.pem*. Es por ello que en este caso, no, no estarían almacenadas de forma segura las claves, ya que si se puede acceder a él, se puede acceder a ambas claves.

d. CREACIÓN DE PAR DE CLAVES CON EL FICHERO RESULTANTE CIFRADO

Para almacenar las claves de forma segura, se puede utilizar un algoritmo de cifrado simétrico cuando se crea el par de claves. En este caso es necesario indicar el algoritmo a utilizar y la contraseña, cuando nos lo pida, se ha utilizado el siguiente comando para ello:

```
openssl genrsa -des3 -out parclaves2.pem 1024
```

Tal y como se puede ver, es similar al primer comando que se utilizó, con la diferencia de que se añade la opción *-des3* que sería la opción de cifrado simétrico para el acceso al fichero con la información del cifrado asimétrico.

openssl rsa -in parclaves2.pem -text -noout **Enter pass phrase for parclaves2.pem:**
Tabla II: solicitud de clave para acceder al fichero de claves

En este caso, antes de poder acceder a la información del fichero, se solicita la contraseña del algoritmo simétrico, tal y como se ve en la *Tabla I*. Una vez se introduce, se vería información similar a la dada en la *Tabla II*.

e. EXTRACCIÓN DE CLAVES PÚBLICAS

Para extraer únicamente la clave pública correspondiente a una clave privada, se usa la opción *-pubout*, en el comando, que indica que la salida contiene sólo la parte pública:

```
openssl rsa -in parclaves.pem -pubout -out publica1.pem
openssl rsa -in parclaves2.pem -pubout -out publica2.pem
```

En el caso del segundo archivo, fue necesario introducir la contraseña para acceder el fichero que contenía la clave pública, ya que está cifrado. Con este comando, es posible compartir un fichero con únicamente la clave pública, lo que hace que sea seguro, ya que quién lo reciba no tiene forma de acceder directamente a la clave privada. Además, es una forma más cómoda de compartirlo.

f. INFORMACIÓN DE LAS DOS CLAVES PÚBLICAS

Ejercicio 1.5: Mostrad la información de las dos claves públicas. Comprobad que no aparece información relativa a la clave privada.

Para revisar el contenido de los archivos se ejecutó el siguiente comando:

```
openssl rsa -in publica1.pem -text -notout
```

Sin embargo, puesto que por defecto se asume que todos los ficheros tienen clave privada y este fichero únicamente cuenta con clave pública, se produjo un error:

```
openssl rsa -in publica1.pem -text -noout
unable to load Private Key
139963808614040:error:0906D06C:PEM routines:PEM_read_bio:no start
line:pem_lib.c:701:Expecting: ANY PRIVATE KEY
```

Tabla III: error al intentar acceder al contenido de los archivos

Para solucionar este problema, fue necesario añadir una opción más en el comando, conforme se informaba de que el fichero de entrada solo contiene claves públicas:

```
openssl rsa -in publica1.pem -text -notout -pubin
```

Así, fue posible obtener información sobre la clave pública:

```
Public-Key: (1024 bit)
Modulus:
00:b0:a9:f6:b8:df:42:9f:a1:b0:28:36:f9:0b:e2:
f1:c5:29:7e:86:c8:d1:8f:c4:34:b0:20:04:0b:b7:
28:d0:98:2d:c2:5b:01:e3:69:fb:ff:cd:7a:2c:03:
19:2a:ec:b2:49:f2:b8:a4:78:84:aa:36:0c:fd:68:
36:49:98:60:21:46:3d:28:86:66:09:74:f7:1e:2f:
8d:0d:13:10:f6:ea:1f:a2:de:92:e2:cb:a7:f2:70:
63:a9:3e:0d:8b:d0:0d:bd:f9:c3:9c:d0:dc:73:e1:
cb:ba:1d:a0:3c:2a:92:d4:2d:2e:59:14:a5:4e:af:
5c:bd:25:e5:e3:5c:ac:2e:fd
Exponent: 65537 (0x10001)
```

Tabla IV: información de la clave pública almacenada

2. CIFRADO Y DESCIFRADO

a. CIFRADO Y DESCIFRADO DE UN TEXTO PEQUEÑO

A partir de una de las claves públicas generadas en el apartado anterior, se ha encriptado un fichero de texto plano de tamaño relativamente pequeño, 128 bytes. Se ha utilizado el siguiente comando para encriptar:

```
openssl rsautl -encrypt -in a.txt -inkey publickey.pem -pubin -out
aencriptada.txt
```

Una vez cifrado, se ha abierto con el fin de comprobar si el encriptado se había realizado apropiadamente, y se pudo observar que el texto original (*fichero a encriptar*) había sido sustituido por lo que se puede ver en la Ilustración I.

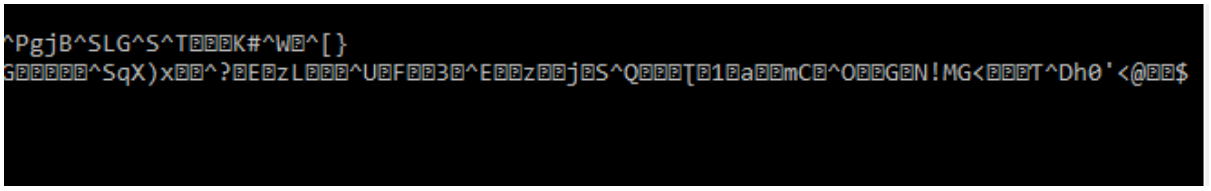


Ilustración 1 Archivo encriptado

A continuación, se decidió desencriptar el archivo, con el siguiente comando:

```
openssl rsautl -decrypt -in aencriptada.txt -inkey parclaves2.pem  -out
adesencriptada.txt
```

En *adesencriptada.txt* se encontraba el mismo texto que se había encriptado a partir de *a.txt*, por lo que se puede decir que el desencriptado fue exitoso.

b. CIFRADO DE UN TEXTO GRANDE

Sabiendo que, para un texto pequeño, la encriptación con algoritmos de cifrado asimétrico funciona apropiadamente, el siguiente paso en esta práctica tenía que ser el intento de cifrar un archivo de un tamaño relativamente grande. Se utilizó un fichero de 1362 bytes, lo que es un poco más de diez veces el tamaño del fichero utilizado en el apartado anterior. Se utilizó el siguiente comando:

```
openssl rsautl -encrypt -in notas.txt -inkey publickey.pem -pubin -out
notascifradas.txt
```

Al ejecutarlo, se obtuvo el error descrito en la *Tabla V*. Este error se debe a que los datos a encriptar son demasiado grandes como para que el módulo utilizado en RSA, explicado en la introducción de la sesión, lo encripte apropiadamente. Por ello, no se puede utilizar cifrado de clave asimétrica para cifrar archivos grandes: para esto ya tenemos los algoritmos de cifrado asimétrico, que han sido estudiados en la práctica 5.

RSA operation error 140026376554136:error:0406D06E:rsa routines:RSA_padding_add_PKCS1_type_2:data too large for key size:rsa_pk1.c:153:
Tabla V: error al intentar encriptar un archivo grande.

c. INTERCAMBIO DE INFORMACIÓN ÚNICAMENTE CON CIFRADO ASIMÉTRICO

A continuación, se realizó un intercambio de información con un compañero de clase. Este intercambio consistió en la petición de la clave pública de otro compañero, para cifrarlo con ella y devolvérselo. También se envió una de las claves públicas generadas previamente, *parclaves2.pem*, y se recibió un fichero encriptado con la clave pública por parte del compañero.

Para el cifrado del archivo a enviar se utilizó el siguiente comando:

```
openssl rsautl -encrypt -in fichero.txt -inkey parclaves2.pem -pubin -
out archivoencriptado.txt
```

Para el descrifrado del archivo recibido se utilizó el siguiente comando:

```
openssl rsautl -decrypt -in corto_rsa_enviar.txt -inkey parclaves2.pem -
out corto_desencriptado.txt
```

Y se obtuvo el siguiente texto descifrado:

Este es un fichero de texto pequeño, que debe ser cifrado.

d. INTERCAMBIO DE INFORMACIÓN CON CIFRADO SIMÉTRICO Y ASIMÉTRICO

Para continuar con la práctica, se cifró un texto más grande con un algoritmo de cifrado simétrico (CBC) y se cifró la contraseña utilizada para el cifrado simétrico con la clave pública del compañero que la envió previamente. También se recibieron los archivos equivalentes por parte del compañero.

Para cifrar simétricamente el texto, se utilizó el siguiente comando:

```
openssl enc -des-cbc -in ejemplo.txt -out fichero_encriptado.txt
```

Para cifrar asimétricamente la contraseña a enviar, se utilizó el siguiente comando:

```
openssl rsautl -encrypt -in contrasinal.txt -inkey publica1.pem -pubin -
out contrasinalencriptado.txt
```

Para descifrar la contraseña recibida, se utilizó el siguiente comando:

```
openssl rsautl -decrypt -in pass_enviar.txt -inkey parclaves2.pem -out
contrasinal_des.txt
```

Con el comando, se obtuvo la contraseña, *hibrido*, y se procedió a desencriptar el cifrado simétrico con el siguiente comando:

```
openssl enc -d -aes-128-cbc -in largo_aes_enviar.txt -out largo_des.txt
```

Tras introducir la contraseña, se desencriptó efectivamente el archivo, que contenía el siguiente texto:

La criptografía asimétrica (en inglés asymmetric key cryptography), también llamada criptografía de clave pública (en inglés public key cryptography) o criptografía de dos claves1(en inglés two-key cryptography), es el método criptográfico que usa un par de claves para el envío de mensajes. Las dos claves pertenecen a la misma persona que ha enviado el mensaje. Una clave es pública y se puede entregar a cualquier persona, la otra clave es privada y el propietario debe guardarla de modo que nadie tenga acceso a ella. Además, los métodos criptográficos garantizan que esa pareja de claves sólo se puede generar una vez, de modo que se puede asumir que no es posible que dos personas hayan obtenido casualmente la misma pareja de claves.

Si el remitente usa la clave pública del destinatario para cifrar el mensaje, una vez cifrado, sólo la clave privada del destinatario podrá descifrar este mensaje, ya que es el único que la conoce. Por tanto se logra la confidencialidad del envío del mensaje, nadie salvo el destinatario puede descifrarlo.

Si el propietario del par de claves usa su clave privada para cifrar el mensaje, cualquiera puede descifrarlo utilizando su clave pública. En este caso se consigue por tanto la identificación y autentificación del remitente, ya que se sabe que sólo pudo haber sido él quien empleó su clave privada (salvo que alguien se la hubiese podido robar). Esta idea es el fundamento de la firma electrónica.

Los 'sistemas de cifrado de clave pública' o 'sistemas de cifrado asimétricos' se inventaron con el fin de evitar por completo el problema del intercambio de claves de los sistemas de cifrado simétricos. Con las claves públicas no es necesario que el remitente y el destinatario se pongan de acuerdo en la clave a emplear. Todo lo que se requiere es que, antes de iniciar la comunicación secreta, el remitente consiga una copia de la clave pública del destinatario. Es más, esa misma clave pública puede ser usada por cualquiera que desee comunicarse con su propietario. Por tanto, se necesitarán sólo n pares de claves por cada n personas que deseen comunicarse entre sí.

3. FIRMAS

a. USO DE CLAVES PARA FIRMAR

Para crear la firma de un documento es necesario cifrar un resumen del mismo utilizando la clave privada. La razón principal para el uso de la clave privada es la necesidad de garantizar la autenticación. Únicamente con la clave privada se puede alcanzar esto, ya que la clave pública es, como bien indica su nombre, pública.

b. VERIFICACIÓN DE UNA FIRMA UTILIZANDO EL RESUMEN DIGITAL

A modo de ejemplo, se realizó la firma de un mensaje. En primer lugar, se obtuvo el resumen de un archivo:

```
openssl dgst -sha1 -binary notas.txt > resumen.txt
```

Se utilizó la opción binary con el fin de poder redirigir el extracto directamente al archivo *resumen.txt* sin necesidad de tener que editarlo a posteriori para utilizarlo como resumen. A continuación, se encriptó el archivo en el que estaba la firma y se desencriptó con los siguientes comandos:

```
openssl rsautl -sign -in resumen.txt -inkey parclaves2.pem -out
firma.txt
```

```
openssl rsautl -verify -in firma.txt -pubin -inkey publickey.pem -out
firmades.txt
```

Finalmente, se compararon los archivos *resumen.txt* y *firmades.txt* con el fin de saber si la firma coincidía una vez desencriptada y, por lo tanto, las operaciones de encriptar y desencriptar en conjunto eran ídempotentes. Se utilizó la herramienta diff para ello:

```
diff resumen.txt firmades.txt
```

Al ejecutarlo, el comando no devolvió nada, por lo que se pudo confirmar que ambos archivos eran iguales y, por lo tanto, las operaciones idempotentes. El hecho de que no se haya modificado, indica que no se realizó ninguna modificación sobre el archivo encriptado.

c. Uso de firma y resumen digital para comprobar la autenticidad de un mensaje

A continuación, se le enviaron tres mensajes firmados a un compañero. Puesto que se trataba de comprobar la autenticidad del mensaje, se editó ligeramente el contenido de mensaje 1, de *Este es el mensaje 1*, se pasó a *Este es el mensaje 1.* . Para la obtención de la firma digital, se realizó el mismo proceso que en el apartado anterior, en primer lugar se obtuvo la firma y a continuación se encriptó. Para ello se utilizaron los siguientes comandos:

```
openssl dgst -sha1 -binary mensaje1.txt > resumen1.txt
openssl dgst -sha1 -binary mensaje2.txt > resumen2.txt
openssl dgst -sha1 -binary mensaje3.txt > resumen3.txt
openssl rsautl -sign -in resumen1.txt -inkey parclaves2.pem -out
firma1.txt
openssl rsautl -sign -in resumen2.txt -inkey parclaves2.pem -out
firma2.txt
openssl rsautl -sign -in resumen3.txt -inkey parclaves2.pem -out
firma3.txt
```

A continuación, se recibió un fichero comprimido por parte del compañero con los tres mensajes y las tres firmas encriptadas. En primer lugar, se descifraron con la clave pública las firmas:

```
openssl rsautl -verify -in mensaje1_firma_sha512_cifrado.txt -pubin -
inkey publica1.pem -out mensaje1_firma_desc.txt
openssl rsautl -verify -in mensaje2_firma_sha512_cifrado.txt -pubin -
inkey publica1.pem -out mensaje2_firma_desc.txt
openssl rsautl -verify -in mensaje3_firma_sha512_cifrado.txt -pubin -
inkey publica1.pem -out mensaje3_firma_desc.txt
```

En segundo lugar, se obtuvieron los resúmenes necesarios a partir de los archivos de los mensajes:

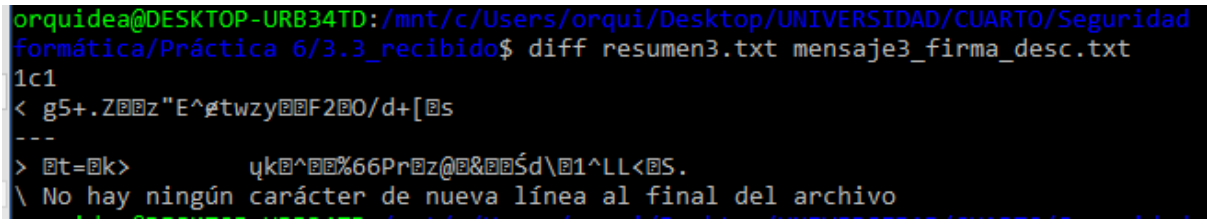
```
openssl dgst -sha512 -binary mensaje1.txt > resumen1.txt
openssl dgst -sha512 -binary mensaje2.txt > resumen2.txt
openssl dgst -sha512 -binary mensaje3.txt > resumen3.txt
```

Finalmente, se utilizó el commando *diff* con los resúmenes digitales asociados a cada uno de los mensajes:

```
diff resumen1.txt mensaje1_firma_desc.txt
```

```
diff resumen2.txt mensaje2_firma_desc.txt
diff resumen3.txt mensaje3_firma_desc.txt
```

Para los resúmenes asociados a *mensaje1.txt* y *mensaje2.txt*, no se detectó ninguna diferencia. Sin embargo, para los resúmenes asociados a *mensaje3.txt*, sí:



Así pues, se puede concluir que el *mensaje3.txt* fue editado después de generar la firma digital.

d. INTERCAMBIO DE INFORMACIÓN

Para finalizar la sesión, se decidió mejorar el proceso de encriptado utilizado en el apartado *2.d* con el fin de poder autentificar al emisor. Así pues, en primer lugar, se encriptó un fichero con el algoritmo de cifrado simétrico CBC:

```
openssl enc -des-cbc -in texto.txt -out fichero_encriptado
```

En segundo lugar, se escribió la contraseña utilizada en un fichero de texto plano y se encriptó asimétricamente con una de las claves públicas generadas previamente:

```
openssl rsautl -encrypt -in contrasinal.txt -inkey publica1.pem -pubin -
out contrasinalencriptado.txt
```

Finalmente, se generó la firma asociada al fichero y se compartió con el compañero:

```
openssl dgst -sha512 -binary tecto.txt > resumen.txt
openssl rsautl -sign -in resumen.txt -inkey parclaves2.pem -out
firma.txt
```

En este momento, el compañero compartió los ficheros asociados a esta parte de la práctica, con el fin de poder desencriptar el fichero recibido y autenticarlo, conforme realmente era suyo. En primer lugar, se desencriptó el fichero contenedor de la contraseña:

```
openssl rsautl -decrypt -in pass_enviar.txt -inkey parclaves2.pem -out
contrasinal_des.txt
```

A desencriptar, se obtuvo la contraseña *mejorado*. A continuación, se desencriptó el texto cifrado simétricamente:

```
openssl enc -d -aes-128-cbc -in largo_aes_enviar.txt -out largo_des.txt
```

El texto descifrado era el mismo que se recibió cuando se realizó el apartado *2.d*. A continuación, se obtuvo el extracto digital de dicho texto y se desencriptó la firma encriptada:

```
openssl dgst -sha512 -binary largo_des.txt > resumen.txt
openssl rsautl -verify -in largo_firma_sha512.txt -pubin -inkey
publica1.pem -out firma.txt
```

Finalmente, se comprobó con *diff* si los resúmenes coincidían:

```
diff firma.txt resumen.txt
```

Y, tal y como era de esperar, sí, ambas firmas coincidían, por lo que se puede afirmar que el mensaje recibido fue correctamente autentificado.

REFERENCIAS

- RSA (Cryptosystem). Wikipedia. 2017. https://en.wikipedia.org/wiki/RSA_(cryptosystem) [Última visita: 23/10/2017]
- *Security in Computing*. 5º edición. C.P. Pfleeger. 2015. C.P. Pfleeger, et al. ISBN: 9780134085043.