

PRACTICA 3 SPSI

Protocolos Criptograficos

Antonio Manuel Rodríguez Martos

1. (0,5 puntos) Generad un archivo sharedDSA.pem que contenga los parametros. Mostrad los valores.

Para generar y manipular los parámetros asociados (primos p , q y el generador g) usaremos el comando `dsaparam` pasandole el archivo destino con `-out` y a continuación el número de bits de los primos.

```
antonio@antonio-DELL:~$ openssl dsaparam -out sharedDSA.pem 901
Generating DSA parameters, 901 bit long prime
This could take some time
```

Mostraremos resultado pasando el archivo de entrada con `-in` y la opción `-text`.

```
antonio@antonio-DELL:~$ openssl dsaparam -in sharedDSA.pem -text
P:
00:e1:34:e8:42:00:5f:3e:e2:57:63:0f:69:1c:bf:
95:48:1a:34:78:9a:84:4c:73:fd:ea:fa:fc:6c:c5:
c9:f6:c9:99:dc:6f:1c:93:84:93:e4:38:d9:00:82:
d6:eb:79:9b:ae:ab:6e:a3:63:a7:59:77:49:82:82:
9c:23:b2:78:03:5c:9f:06:32:77:2f:3f:7c:5a:ab:
07:32:04:9e:b5:b3:95:c8:34:89:ac:af:54:ed:c4:
c6:87:c6:79:75:0d:ab:11:71:d6:b9:df:b6:06:9a:
83:34:62:8a:d3:b9:d4:ec:62:59:19:70:43:4a:41:
a5
Q:
00:8b:0f:c2:ab:38:e2:a8:d0:db:f2:00:73:5b:43:
d8:75:c2:7d:12:51
G:
00:a3:7e:74:32:e4:5a:fa:a2:0a:ee:00:c4:dc:9d:
38:c4:5a:4b:5b:62:97:a7:6b:56:48:72:0b:f4:f4:
18:32:c7:e8:6e:08:b8:ce:29:b4:b7:5b:01:99:1e:
aa:be:e6:0e:29:9e:15:cd:9c:f7:df:17:90:68:56:
a4:b8:49:6f:b2:37:69:5a:29:c0:98:cb:bc:ee:a1:
19:8d:f0:21:ae:34:58:32:de:55:4e:e9:6c:2a:27:
9c:ec:f9:cc:68:4b:e0:82:61:58:f3:29:1d:6b:8c:
ac:bd:03:b9:91:86:3c:6e:07:79:6a:36:97:a1:80:
44
-----BEGIN DSA PARAMETERS-----
MIIBDQJ5A0E06EIXz7iV2MPaRy/lUGaNHiahExz/er6/GzFyfbJmdxvHJOEk+Q4
2QCC1ut5m66rbqNjp1l3SYKCnC0yeANcnwYydy8/fFqrBzIEnrWzlcg0iayvV03E
xofGeXUNqxFx1rnftgaagzRiit0510xiWRLwQ0pBpQIVAIspwqs44qjQ2/IAC1tD
2HXCfRJRAnkAo350MuRa+qIK7gDE3J04xFpLW2KXp2tWSHIL9PQYMsfobgi4zim0
t1sBmR6qvUyOKZ4VzZz33xeQaFakuElvsjdpWinAmMu87qEZjfAhrjRYMt5VTuls
Kiec7PnMaEvggmFY8ykda4ysvQ05kYY8bgd5ajaXoYBE
-----END DSA PARAMETERS-----
```

2. (0,5 puntos) Generad dos parejas de claves para los parametros anteriores. La claves se almacenaran en los archivos <nombre>DSAkey.pem y <apellido> DSAkey.pem
No es necesario protegerlas por contraseña.

Para la generación de claves DSA a partir de un archo que contenga los parametros anteriores utilizaremos gendsa.

Le pasaremos dicho archivo y generaremos las claves el el archivo que pongamos como destino con la opción -out.

Lo haremos tanto para “antonio” como para “rodriguez”.

```
antonio@antonio-DELL:~$ openssl gendsa sharedDSA.pem -out antonioDSAkey.pem
Generating DSA key, 960 bits
antonio@antonio-DELL:~$ openssl gendsa sharedDSA.pem -out rodriguezDSAkey.pem
Generating DSA key, 960 bits
```

Mostraremos resultados de nombre y apellidos pasando el archivo de entrada con -in y la opción -text.

```
antonio@antonio-DELL:~$ openssl dsa -in antonioDSAkey.pem -text
read DSA key
Private-Key: (960 bit)
priv:
    57:93:33:fc:f2:05:34:f8:4f:c4:3c:da:9f:bb:97:
    a4:3e:7f:6d:8a
pub:
    0a:07:be:f6:12:87:2c:85:0e:61:95:f6:ea:f0:f9:
    43:9f:c1:93:d5:64:46:d3:75:18:72:a4:f3:4e:fd:
    2c:e8:9e:a6:d5:00:6e:11:43:9a:b2:3e:41:34:2c:
    76:8d:37:8d:5c:a6:d3:22:dc:42:26:8e:07:e3:88:
    59:65:10:84:7a:5a:db:11:b0:6d:81:93:90:6b:89:
    0b:da:98:09:28:16:f8:bb:34:81:56:c8:e7:7b:1e:
    1f:c7:84:84:56:28:d5:8f:a7:64:91:16:ec:36:75:
    69:08:cd:21:55:68:bd:f9:bb:cb:59:bb:4c:ce:de
P:
    00:e1:34:e8:42:00:5f:3e:e2:57:63:0f:69:1c:bf:
    95:48:1a:34:78:9a:84:4c:73:fd:ea:fa:fc:6c:c5:
    c9:f6:c9:99:dc:6f:1c:93:84:93:e4:38:d9:00:82:
    d6:eb:79:9b:ae:ab:6e:a3:63:a7:59:77:49:82:82:
    9c:23:b2:78:03:5c:9f:06:32:77:2f:3f:7c:5a:ab:
    07:32:04:9e:b5:b3:95:c8:34:89:ac:af:54:ed:c4:
    c6:87:c6:79:75:0d:ab:11:71:d6:b9:df:b6:06:9a:
    83:34:62:8a:d3:b9:d4:ec:62:59:19:70:43:4a:41:
    a5
Q:
    00:8b:0f:c2:ab:38:e2:a8:d0:db:f2:00:73:5b:43:
    d8:75:c2:7d:12:51
G:
    00:a3:7e:74:32:e4:5a:fa:a2:0a:ee:00:c4:dc:9d:
    38:c4:5a:4b:5b:62:97:a7:6b:56:48:72:0b:f4:f4:
    18:32:c7:e8:6e:08:b8:ce:29:b4:b7:5b:01:99:1e:
    aa:be:e6:0e:29:9e:15:cd:9c:f7:df:17:90:68:56:
    a4:b8:49:6f:b2:37:69:5a:29:c0:98:cb:bc:ee:a1:
    19:8d:f0:21:ae:34:58:32:de:55:4e:e9:6c:2a:27:
    9c:ec:f9:cc:68:4b:e0:82:61:58:f3:29:1d:6b:8c:
    ac:bd:03:b9:91:86:3c:6e:07:79:6a:36:97:a1:80:
    44
```

```
antonio@antonio-DELL:~$ openssl dsa -in rodriguezDSAkey.pem -text
read DSA key
Private-Key: (960 bit)
priv:
    5b:60:ef:60:a8:15:cc:18:70:32:03:ac:72:d6:96:
    99:a5:73:1f:71
pub:
    29:56:8c:1c:46:33:b5:f0:ff:59:d7:d3:ae:51:72:
    cd:8f:eb:ea:3d:4f:1c:a9:1c:35:c8:52:81:e3:02:
    80:5d:5b:2f:ab:c1:95:83:38:c8:e4:77:a1:89:05:
    48:b4:71:3d:5d:fc:55:19:c8:24:15:2a:c8:b0:4d:
    85:eb:05:e2:ee:69:9a:01:40:1c:b9:24:35:f1:2e:
    e1:58:73:fb:a8:8f:9f:3a:fa:c7:04:8e:d5:64:2c:
    ab:cb:a4:98:62:3b:d6:fb:62:50:81:0a:48:61:ba:
    7a:88:6a:a3:80:0f:23:fc:f4:1a:43:fe:bc:c4:da
P:
    00:e1:34:e8:42:00:5f:3e:e2:57:63:0f:69:1c:bf:
    95:48:1a:34:78:9a:84:4c:73:fd:ea:fa:fc:6c:c5:
    c9:f6:c9:99:dc:6f:1c:93:84:93:e4:38:d9:00:82:
    d6:eb:79:9b:ae:ab:6e:a3:63:a7:59:77:49:82:82:
    9c:23:b2:78:03:5c:9f:06:32:77:2f:3f:7c:5a:ab:
    07:32:04:9e:b5:b3:95:c8:34:89:ac:af:54:ed:c4:
    c6:87:c6:79:75:0d:ab:11:71:d6:b9:df:b6:06:9a:
    83:34:62:8a:d3:b9:d4:ec:62:59:19:70:43:4a:41:
    a5
Q:
    00:8b:0f:c2:ab:38:e2:a8:d0:db:f2:00:73:5b:43:
    d8:75:c2:7d:12:51
G:
    00:a3:7e:74:32:e4:5a:fa:a2:0a:ee:00:c4:dc:9d:
    38:c4:5a:4b:5b:62:97:a7:6b:56:48:72:0b:f4:f4:
    18:32:c7:e8:6e:08:b8:ce:29:b4:b7:5b:01:99:1e:
    aa:be:e6:0e:29:9e:15:cd:9c:f7:df:17:90:68:56:
    a4:b8:49:6f:b2:37:69:5a:29:c0:98:cb:bc:ee:a1:
    19:8d:f0:21:ae:34:58:32:de:55:4e:e9:6c:2a:27:
    9c:ec:f9:cc:68:4b:e0:82:61:58:f3:29:1d:6b:8c:
    ac:bd:03:b9:91:86:3c:6e:07:79:6a:36:97:a1:80:
```

3. (0,5 puntos) “Extraed” la clave privada contenida en el archivo nombreDSAkey.pem a otro archivo que tenga por nombre nombreDSApriv.pem. Este archivo debera estar protegido por contraseña. Mostrad sus valores. Haced lo mismo para el archivo apellidoDSAkey.pem

Extraeremos la clave privada usando la opción dsa especificando el archivo de entrada, el cual creamos en el apartado anterior y del que la obtendremos con la opción -in, el archivo destino donde irá la clave con -out y el algoritmo de cifrado que utilizaremos; que como dice el manual podremos usar aes, camelia, des3 o idea, en este caso utilizaremos AES-128 que ya hemos usado en prácticas anteriores.

Lo haremos tanto para “antonio” como para “rodriguez”.

La contraseña introducida será 0123456789.

```
antonio@antonio-DELL:~$ openssl dsa -in antonioDSAkey.pem -out antonioDSApriv.pem -aes128
read DSA key
writing DSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
antonio@antonio-DELL:~$
antonio@antonio-DELL:~$ openssl dsa -in rodriguezDSAkey.pem -out rodriguezDSApriv.pem -aes128
read DSA key
writing DSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

Como vemos las claves privadas están cifradas.

```
antonio@antonio-DELL:~$ cat rodriguezDSApriv.pem
-----BEGIN DSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,0420C52E6B23F17437B0FFA44E9BE5D8

w1+Ahn9ta0VlZQGQPLQLJxAcQ9JaF/uh4wGAtkbaqb2CBDFfbvaRFVRwMdEKr3+h
FGP98oxo1/RoNov2ZBv0HpnXPTJmjGXzptMGgFqBi/akLtjQ7xfUcw7yrsPfiXm5
6/lHkM/SJZPyrjn2+I1f6oQ0Kh0UxXbgze81wAGLWBVZ1IGM1qZUEt8vcEUHs0q1
LR02gJ0a9TvGEwltDa2vPw6fHH52XtHdJhVnt0BVfOYHWTh7VIf+ANVRoCwa+1JG
76wekim//U7IKJyVnIAfNKCTD1i5f9r1DX0Ir6sLGaX04r9gZl9qyDc8t5e2bL57
9boQM6cktoHvVvSFNDjg7s23HP1k6aAI3pk09nI+eddw0HAoMPRrsR1qGE/HBeCys
3qFo3BkQrK7Ki7tNpmWLHUex2fIgljVNvYdL2Y8joREp+OiktL70JSY7tLaphfbl
3cC2eClrxILrms46wFbZaoYiqd4rpPSEvg7x6id90bm/NhQii5o6mxVdBizcA/FV
Tlhjfbv0nbFG8LRSe0WLuUxD+q5wt0jrNYsrVvZUjxQDt2dpbp2AlneaHYWYQ2V
-----END DSA PRIVATE KEY-----
antonio@antonio-DELL:~$ cat antonioDSApriv.pem
-----BEGIN DSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,09D062362D33A42A54060BE320C51A0E

7fQ0h7e5mLN1M2YfG8TAgCv2v+Z9Mg5sazK+d6UJpQR7MMvrTwr7kBAJU3qVv/p/
z9CCmYXfKqY6KTr6k9tJINaEkyL3jzulXyWCNXtoc2bLvS/rC3J8StUY3mCl03L9
NFVMVVBfsaU03IpXdBxVvxa3ApyIWXnLJ+gqz9ZgUok8sBEkRV1s8Vkjjaq0ZXek
vkeYKdunn55ERzswceuJDihiZSiVYrfm+zH1d3zRrA07DyBCJrHj6j5JNwr4PucR
ZxTPJfgDGdQrJ1HoEBVr+xhnpvh6RnEtTiXzxQaszvz0IiJG7ciNQJnLFz5fClGw
CpEAvGdnVrp1jhx8EUxwfaILIoH3gMugj0ccCNgIrb43iI8I6/r54bfK9byivnNL
JfCd3C0yNxnVU26WRKsByVht5kffihPzubKEKG+DYUDAKKRH5iDiZibmf/a9PQsK
xu9VBEusDyLpBs8B39F5ePvB9DQVibH6L1sTufzo50JG1+WjuQRMH6dTWsBjpty+
Z3LEdmztVuvTpvDoNaFBp3Handc10cDhKmwC4R4rHgSJvLLYPCDjPt1xuKgARpQA
-----END DSA PRIVATE KEY-----
```


4. (0,5 puntos) Extraer en <nombre>DSAPub.pem la clave pública contenida en el archivo <nombre>DSAkey.pem. De nuevo <nombre>DSAPub.pem no debe estar cifrado ni protegido. Mostrad sus valores. Lo mismo para el archivo <apellido>DSAkey.pem.

Extraeremos la clave pública usando la opción dsa especificando el archivo de entrada, el cual creamos en el apartado 2 y del que la obtendremos con la opción -in y -pubout, además del archivo destino donde irá la clave con -out. La clave pública no debe estar cifrada.

Lo haremos tanto para “antonio” como para “rodriguez”.

```
antonio@antonio-DELL:~$ openssl dsa -in antonioDSAkey.pem -out antonioDSAPub.pem -pubout
read DSA key
writing DSA key
antonio@antonio-DELL:~$ openssl dsa -in rodriguezDSAkey.pem -out rodriguezDSAPub.pem -pubout
read DSA key
writing DSA key
```

Mostraremos resultados de nombre y apellidos pasando el archivo de entrada con -in y las opciones -text y -pubin. Como vemos son las mismas que ya sacamos por pantalla en el apartado 2.

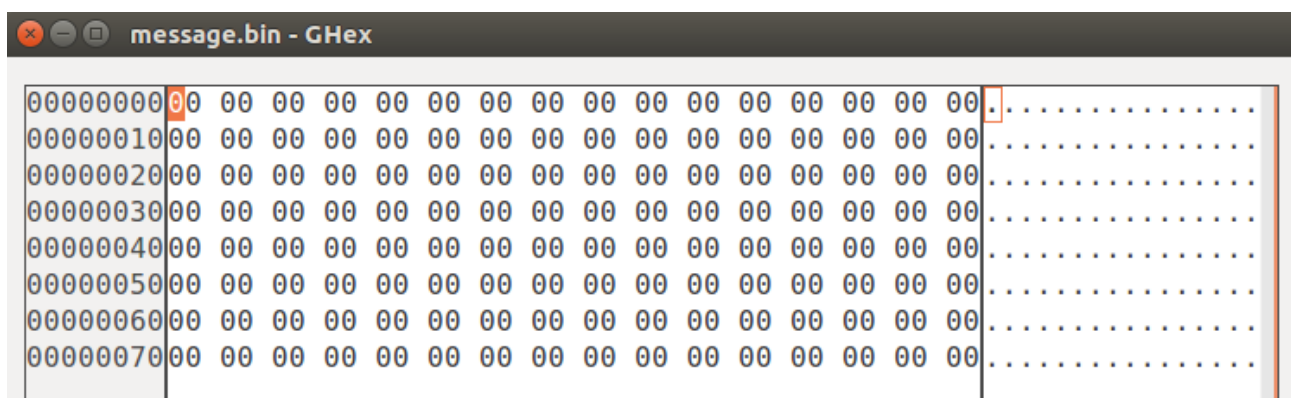
```
antonio@antonio-DELL:~$ openssl dsa -in antonioDSAPub.pem -text -pubin
read DSA key
pub:
 0a:07:be:f6:12:87:2c:85:0e:61:95:f6:ea:f0:f9:
 43:9f:c1:93:d5:64:46:d3:75:18:72:a4:f3:4e:fd:
 2c:e8:9e:a6:d5:00:6e:11:43:9a:b2:3e:41:34:2c:
 76:8d:37:8d:5c:a6:d3:22:dc:42:26:8e:07:e3:88:
 59:65:10:84:7a:5a:db:11:b0:6d:81:93:90:6b:89:
 0b:da:98:09:28:16:f8:bb:34:81:56:c8:e7:7b:1e:
 1f:c7:84:84:56:28:d5:8f:a7:64:91:16:ec:36:75:
 69:08:cd:21:55:68:bd:f9:bb:cb:59:bb:4c:ce:de
P:
 00:e1:34:e8:42:00:5f:3e:e2:57:63:0f:69:1c:bf:
 95:48:1a:34:78:9a:84:4c:73:fd:ea:fa:fc:6c:c5:
 c9:f6:c9:99:dc:6f:1c:93:84:93:e4:38:d9:00:82:
 d6:eb:79:9b:ae:ab:6e:a3:63:a7:59:77:49:82:82:
 9c:23:b2:78:03:5c:9f:06:32:77:2f:3f:7c:5a:ab:
 07:32:04:9e:b5:b3:95:c8:34:89:ac:af:54:ed:c4:
 c6:87:c6:79:75:0d:ab:11:71:d6:b9:df:b6:06:9a:
 83:34:62:8a:d3:b9:d4:ec:62:59:19:70:43:4a:41:
 a5
Q:
 00:8b:0f:c2:ab:38:e2:a8:d0:db:f2:00:73:5b:43:
 d8:75:c2:7d:12:51
G:
 00:a3:7e:74:32:e4:5a:fa:a2:0a:ee:00:c4:dc:9d:
 38:c4:5a:4b:5b:62:97:a7:6b:56:48:72:0b:f4:f4:
 18:32:c7:e8:6e:08:b8:ce:29:b4:b7:5b:01:99:1e:
 aa:be:e6:0e:29:9e:15:cd:9c:f7:df:17:90:68:56:
 a4:b8:49:6f:b2:37:69:5a:29:c0:98:cb:bc:ee:a1:
 19:8d:f0:21:ae:34:58:32:de:55:4e:e9:6c:2a:27:
 9c:ec:f9:cc:68:4b:e0:82:61:58:f3:29:1d:6b:8c:
 ac:bd:03:b9:91:86:3c:6e:07:79:6a:36:97:a1:80:
 44
```

```

antonio@antonio-DELL:~$ openssl dsa -in rodriguezDSApub.pem -text -pubin
read DSA key
pub:
 29:56:8c:1c:46:33:b5:f0:ff:59:d7:d3:ae:51:72:
 cd:8f:eb:ea:3d:4f:1c:a9:1c:35:c8:52:81:e3:02:
 80:5d:5b:2f:ab:c1:95:83:38:c8:e4:77:a1:89:05:
 48:b4:71:3d:5d:fc:55:19:c8:24:15:2a:c8:b0:4d:
 85:eb:05:e2:ee:69:9a:01:40:1c:b9:24:35:f1:2e:
 e1:58:73:fb:a8:8f:9f:3a:fa:c7:04:8e:d5:64:2c:
 ab:cb:a4:98:62:3b:d6:fb:62:50:81:0a:48:61:ba:
 7a:88:6a:a3:80:0f:23:fc:f4:1a:43:fe:bc:c4:da
P:
 00:e1:34:e8:42:00:5f:3e:e2:57:63:0f:69:1c:bf:
 95:48:1a:34:78:9a:84:4c:73:fd:ea:fa:fc:6c:c5:
 c9:f6:c9:99:dc:6f:1c:93:84:93:e4:38:d9:00:82:
 d6:eb:79:9b:ae:ab:6e:a3:63:a7:59:77:49:82:82:
 9c:23:b2:78:03:5c:9f:06:32:77:2f:3f:7c:5a:ab:
 07:32:04:9e:b5:b3:95:c8:34:89:ac:af:54:ed:c4:
 c6:87:c6:79:75:0d:ab:11:71:d6:b9:df:b6:06:9a:
 83:34:62:8a:d3:b9:d4:ec:62:59:19:70:43:4a:41:
 a5
Q:
 00:8b:0f:c2:ab:38:e2:a8:d0:db:f2:00:73:5b:43:
 d8:75:c2:7d:12:51
G:
 00:a3:7e:74:32:e4:5a:fa:a2:0a:ee:00:c4:dc:9d:
 38:c4:5a:4b:5b:62:97:a7:6b:56:48:72:0b:f4:f4:
 18:32:c7:e8:6e:08:b8:ce:29:b4:b7:5b:01:99:1e:
 aa:be:e6:0e:29:9e:15:cd:9c:f7:df:17:90:68:56:
 a4:b8:49:6f:b2:37:69:5a:29:c0:98:cb:bc:ee:a1:
 19:8d:f0:21:ae:34:58:32:de:55:4e:e9:6c:2a:27:
 9c:ec:f9:cc:68:4b:e0:82:61:58:f3:29:1d:6b:8c:
 ac:bd:03:b9:91:86:3c:6e:07:79:6a:36:97:a1:80:
 44

```

5. Coged un archivo cualquiera cualquiera, que actuar# a como entrada, con al menos 128 bytes. En adelanteme referir#e a #el como message, pero pod#eis llamarlo como os parezca

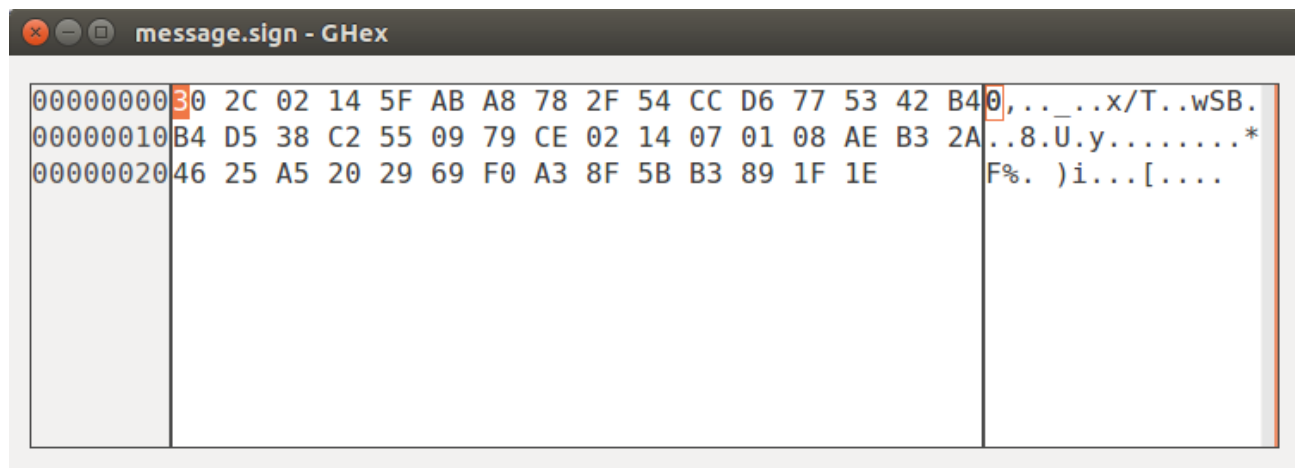


6. (0,5 puntos) Firmad directamente el archivo message empleando el comando openssl pkeyutl sin calcular valores hash, la firma deber# a almacenarse en un archivo llamado, por ejemplo, message.sign. Mostrad el archivo con la firma.

Para firmar con pkeyutl usaremos la opción -sign pasandole la clave privada con la opción -inkey (en este caso como no lo dice por ejemplo utilizaremos la clave privada de “antonio”), le pasaremos el archivo de entrada el cual firmaremos con la opción -in y el archivo donde se almacenará la firma con al opción -out.

A continuación introduciremos la contraseña de la clave privada.

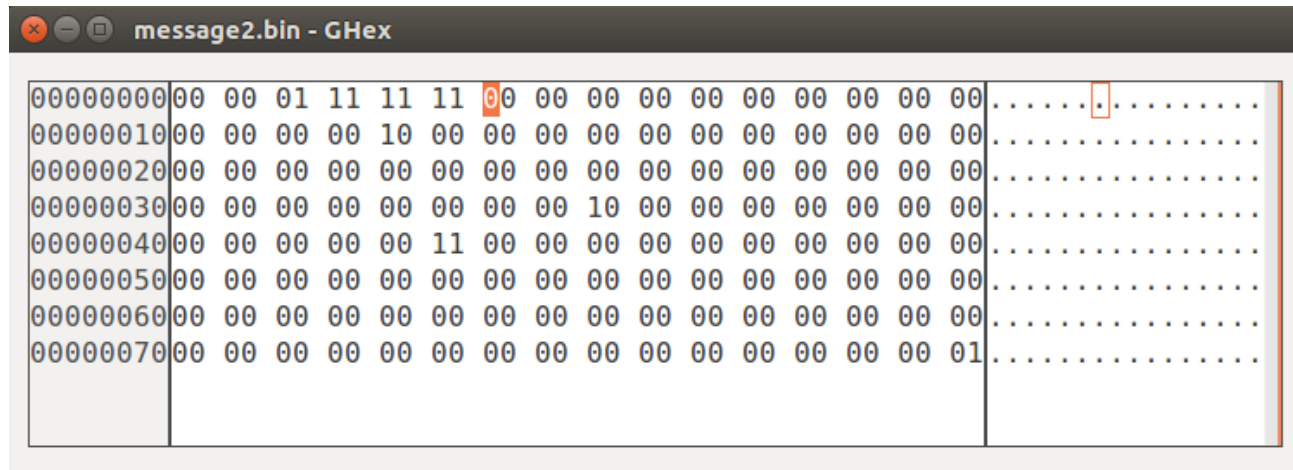
```
antonio@antonio-DELL:~$ openssl pkeyutl -sign -inkey antonioDSPriv.pem -in message.bin -out message.sign
Enter pass phrase for antonioDSPriv.pem:
```



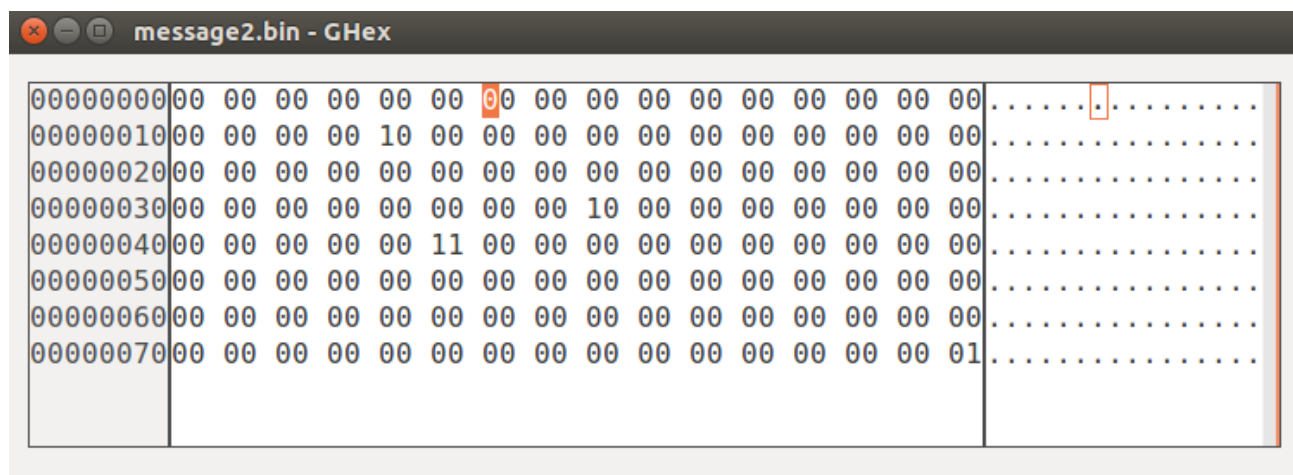
7. (1 punto) Construid un archivo message2 diferente de message tal que la verificación de la firma message.sign sea correcta con respecto al archivo message2.

La modificación del archivo de message no podrá ser en los primeros 20 bytes ya que no se cumpliría la comprobación de la verificación de la firma.

Esta no funcionaría:



Esta si:



Para hacer la verificación con pkeyutl usaremos la opción -verify pasandole la clave publica en este caso como utilizamos la clave de antonio obviamente le pasaremos esa, con la opcion -pubin e -inkey. A continuación el archivo de entrada que será message o message2 con -in y después el archivo de la firma con la opcion -sigfile.

```
antonio@antonio-DELL:~$ openssl pkeyutl -verify -pubin -inkey antonioDSAPub.pem -in message.bin -sigfile message.sign
Signature Verified Successfully
```

```
antonio@antonio-DELL:~$ openssl pkeyutl -verify -pubin -inkey antonioDSAPub.pem -in message2.bin -sigfile message.sign
Signature Verified Successfully
```

Como vemos con las indicaciones de los bytes seguidas la verificación ha sido validada.

8. (0,5 puntos) Calculad el valor hash del archivo con la clave p#ublica nombreDSApub.pem usando sha384 con salida hexadecimal con bloques de dos caracteres separados por dos puntos. Mostrad los valores por salida est#andar y guardadlo en nombreDSApub.sha384.

Para generar unos valores únicos con respecto a un archivo usaremos la funcion hash

Para trabajar con ello utilizaremos dgst pasandole la función, en este caso -sha384 (salida de 384 bits).

Para que la salida sea en hexadecimal solo tendremos que poner a continuación -hex y el archivo destino con -out o -c, por último el archivo donde esta la clave publica del que se calculará el hash.

```
antonio@antonio-DELL:~$ openssl dgst -sha384 -hex -out antonioDSApub.sha384 -c antonioDSApub.pem
```

```
antonio@antonio-DELL:~$ cat antonioDSApub.sha384
SHA384(antonioDSApub.pem)= e1:6b:44:a1:bf:32:82:b4:88:db:91:fc:55:fa:9c:c7:59:08:2f:b2:
11:d0:98:00:81:c7:01:fb:ed:80:7b:d1:b7:17:c8:28:83:19:27:bd:21:3a:af:f4:b4:e0:a7:68
```

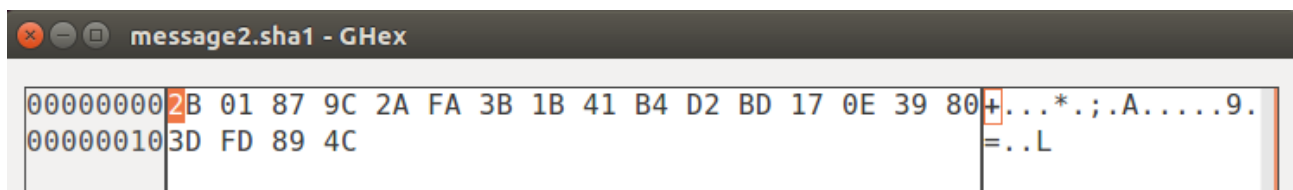
9. (0,5 puntos) Calculad el valor hash de message2 usando una funci#on hash de 160 bits con salida binaria. Guardad el hash en message2.<algoritmo> y mostrad su contenido.

Para buscar las funciones hash disponibles podremos utilizar el siguiente comando:

```
antonio@antonio-DELL:~$ openssl list --digest-commands
```

Como hicimos antes utilizaremos dgst, en este caso usaremos la funcion -sha1 (salida de 160 bits) y salida binaria con la opción -binary.

```
antonio@antonio-DELL:~$ openssl dgst -sha1 -binary -out message2.sha1 message2.bin
```



```
message2.sha1 - GHex
00000000 2B 01 87 9C 2A FA 3B 1B 41 B4 D2 BD 17 0E 39 80 +...*.;.A.....9.
00000010 3D FD 89 4C =..L
```

10. (0,5 puntos) Firmad el archivo message2 mediante el comando openssl dgst y la funci#on hash del punto anterior. La firma deber#a almacenarse en un archivo llamado, por ejemplo, message2.sign

En este caso para firmar con dgst usaremos la opción -sign pasandole la clave privada (seguimos como antonio por lo que utilizaremos la clave privada de “antonio”).

Utilizaremos la función hash anterior (sha1) y le pasaremos el archivo donde se almacenará la firma con al opción -out y despues el archivo de entrada el cual firmaremos.

A continuación introduciremos la contraseña de la clave privada.

```
antonio@antonio-DELL:~$ openssl dgst -sha1 -sign antonioDSPriv.pem -out message
2.sign message2.bin
Enter pass phrase for antonioDSPriv.pem:
```

11. (1 punto) Vericad la firma message2.sign con los archivos message y message2 empleando el comando openssl dgst.

Para hacer la verificación con dgst pondremos la función hash utilizada -sha1 y usaremos la opción -verify pasandole la clave pública en este caso como utilizamos la clave de antonio obviamente le pasaremos esa, con la opción -pubin e -inkey. A continuacion el archivo de la firma con la opción -signature y justo despues el archivo de entrada que será mesagge o message2.

```
antonio@antonio-DELL:~$ openssl dgst -sha1 -verify antonioDSAPub.pem -signature
message2.sign message.bin
Verification Failure
antonio@antonio-DELL:~$ openssl dgst -sha1 -verify antonioDSAPub.pem -signature
message2.sign message2.bin
Verified OK
```

Como vemos la verificación de message2 es correcta pero la de message no es correcta como corresponde ya que ulizamos el hash que solo está asociado a message2.

12. (0,5 puntos) Vericad que message2.sign es una firma correcta para message2 pero empleando el comando openssl pkeyutl

Utilizaremos los comandos del punto 7 con los mismos archivos anteriores salvo que para indicar la función utilizaremos la opción -digest:sha1

```
antonio@antonio-DELL:~$ openssl pkeyutl -verify -pubin -inkey antonioDSAPub.pem
-in message2.sha1 -sigfile message2.sign -pkeyopt digest:sha1
Signature Verified Successfully
```

13. (0,5 puntos) Generad el valor HMAC del archivo sharedDSA.pem con clave '12345' mostr#andolo por pantalla.

El valor HMAC que sirve para la autenticación de mensajes basados en clave y hash podremos obtenerlo con la opción de dgst -hmac y a continuacion la clave para generarlo y el archivo salida.

```
antonio@antonio-DELL:~$ openssl dgst -hmac 12345 sharedDSA.pem
HMAC-SHA1(sharedDSA.pem)= 9c7c3fa686c4b9d4202900ed6555c773fec29773
```

14. (3 puntos) Simulad una ejecución completa del protocolo Estación a Estación. Para ello emplearemos como claves para rma/verificación las generadas en esta práctica, y para el protocolo DH emplearemos las claves asociadas a curvas elípticas de la práctica anterior junto con las de otro usuario simulado que deberéis generar nuevamente. El algoritmo simétrico a utilizar en el protocolo estación a estación será AES-128 en modo CFB8.

Crearemos las claves como hicimos en la práctica anterior con EC.

Lo haremos tanto para antonio como para rodriguez.

```
antonio@antonio-DELL:~$ openssl ecparam -in stdECparam.pem -genkey -out antonioEKey.pem -noout
antonio@antonio-DELL:~$ openssl ec -in antonioEKey.pem -out antonioECpriv.pem -aes-256-cbc
read EC key
writing EC key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
antonio@antonio-DELL:~$ openssl ec -in antonioEKey.pem -out antonioECpub.pem -pubout
read EC key
writing EC key
```

```
antonio@antonio-DELL:~$ openssl ecparam -in stdECparam.pem -genkey -out rodriguezEKey.pem -noout
antonio@antonio-DELL:~$ openssl ec -in rodriguezEKey.pem -out rodriguezECpriv.pem -aes-256-cbc
read EC key
writing EC key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
antonio@antonio-DELL:~$ openssl ec -in rodriguezEKey.pem -out rodriguezECpub.pem -pubout
read EC key
writing EC key
```

Suponemos que ambos comparten claves públicas y obtienen la key idéntica en ambos con su clave privada.

Por ejemplo antonio haría:

```
antonio@antonio-DELL:~$ openssl pkeyutl -derive -inkey antonioECpriv.pem -peerkey rodriguezECpub.pem -out Key.bin
Enter pass phrase for antonioECpriv.pem:
```

Rodriguez juntaría las dos claves públicas en pubJuntas.

```
antonio@antonio-DELL:~$ cat antonioECpub.pem rodriguezECpub.pem > pubJuntas
```

Rodriguez firmaría el archivo anterior de las claves públicas con su privada, el cual llamaremos rodriguezFirma.

```
antonio@antonio-DELL:~$ openssl dgst -sign rodriguezDSPriv.pem -out rodriguezFirma pubJuntas
Enter pass phrase for rodriguezDSPriv.pem:
```

Rodriguez ahora encriptaría el archivo firmado con aes 128 cfb8 como dice el enunciado.

```
antonio@antonio-DELL:~$ openssl enc -aes-128-cfb8 -in rodriguezFirma -out encRodriguezFirma -kfile Key.bin
```

Después se lo enviaría a antonio.

Antonio Juntaría las claves

```
antonio@antonio-DELL:~$ cat antonioECpub.pem rodriguezECpub.pem > pubJuntas_2
```

Antonio descifraría el archivo firmado con la key.

```
antonio@antonio-DELL:~$ openssl enc -aes-128-cfb8 -d -in encRodriguezFirma -out desencRodriguezFirma -kfile Key.bin
```

Antonio verificaría la firma del archivo con el desencryptado correspondiente a la firma de que Rodriguez envio.

```
antonio@antonio-DELL:~$ openssl dgst -verify rodriguezDSAPub.pem -signature desencRodriguezFirma pubJuntas_2
Verified OK
```

Ahora Antonio haría los mismos pasos que Rodriguez hizo e enviaría a Rodriguez.

```
antonio@antonio-DELL:~$ cat rodriguezECpub.pem antonioECpub.pem > pubJuntas_3
```

```
antonio@antonio-DELL:~$ openssl dgst -sign antonioDSApriv.pem -out antonioFirma pubJuntas_3
Enter pass phrase for antonioDSApriv.pem:
```

```
antonio@antonio-DELL:~$ openssl enc -aes-128-cfb8 -in antonioFirma -out encAntonioFirma -kfile Key.bin
```

Rodriguez despues de recibir el encriptado de la firma juntaria las claves publicas.

```
antonio@antonio-DELL:~$ cat rodriguezECpub.pem antonioECpub.pem > pubJuntas_4
```

Rodriguez con el encriptado de la firma de Antonio lo descifraría con la key.

```
antonio@antonio-DELL:~$ openssl enc -aes-128-cfb8 -d -in encAntonioFirma -out desencAntonioFirma -kfile Key.bin
```

Y verificaría con las claves juntas de él y Antonio.

```
antonio@antonio-DELL:~$ openssl dgst -verify antonioDSAPub.pem -signature desencAntonioFirma pubJuntas_4
Verified OK
```


Biografía

[1.] Manual de OpenSSL.

<https://www.openssl.org/docs/>