Seguridad y Protección de Sistemas Informáticos Práctica 3

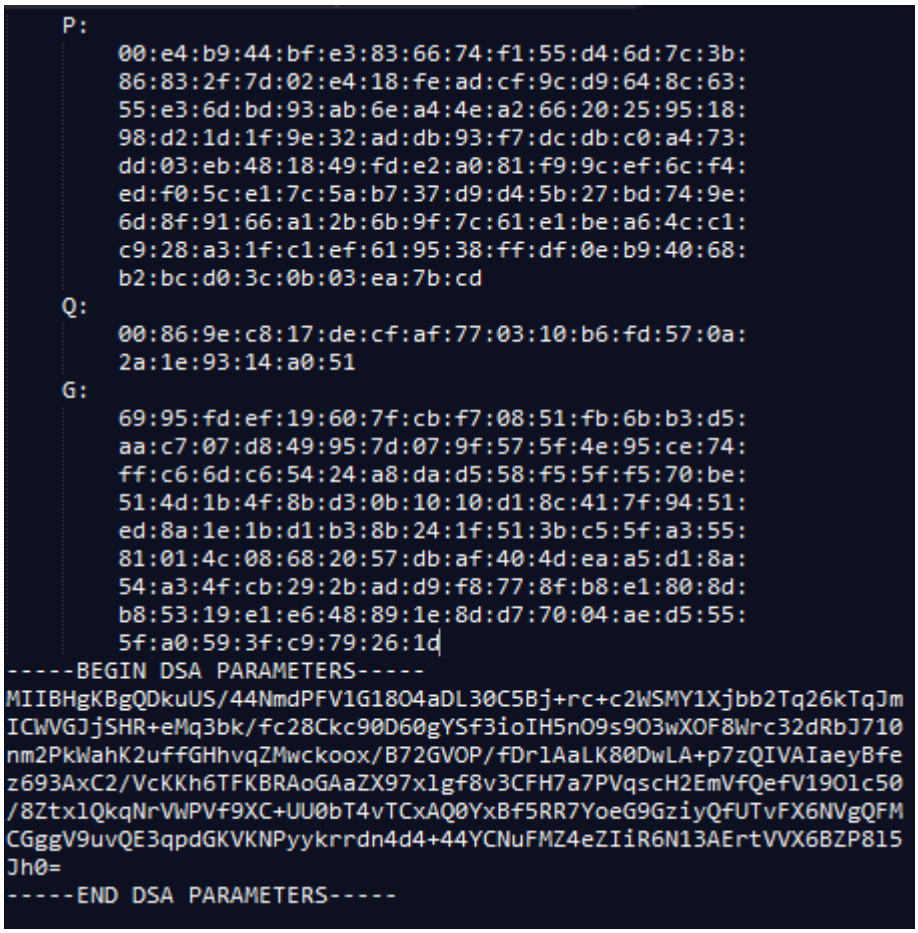
Miguel Morales Castillo

1. Generad un archivo sharedDSA.pem que contenga los parámetros. Mostrad los valores.

   Para crear los parámetros para DSA usamos el siguiente comando.

   dsaparam -text -out "C:\Users\Miguemc\Desktop\sharedDSA.pem" 1024

   Con este comando , pasándole el tamaño (1024 bits en este caso) genera p, q y g , a partir de los cuales se genera la clave que contiene el par de claves pública/privada. Estos son los valores generados.



2. Generad dos parejas de claves para los parámetros anteriores. La claves se almacenarán en los archivos nombreDSAkey.pem y apellidoDSAkey.pem. No es necesario protegerlas por contraseña.

   Para hacer esta tarea usaremos el siguiente comando dos veces usando miguelDSA.pem y castilloDSA.pem cada vez para crear las dos parejas de claves:

   gendsa -out "C:\Users\Miguemc\Desktop\(miguelDSA.pem|castilloDSA.pem)" "C:\Users\Miguemc\Desktop\sharedDSA.pem"

3. “Extraed" la clave privada contenida en el archivo nombreDSAkey.pem a otro archivo que tenga por nombre nombreDSApriv.pem. Este archivo deberá estar protegido por contraseña. Mostrad sus valores. Lo mismo para el archivo apellidoDSAkey.pem.

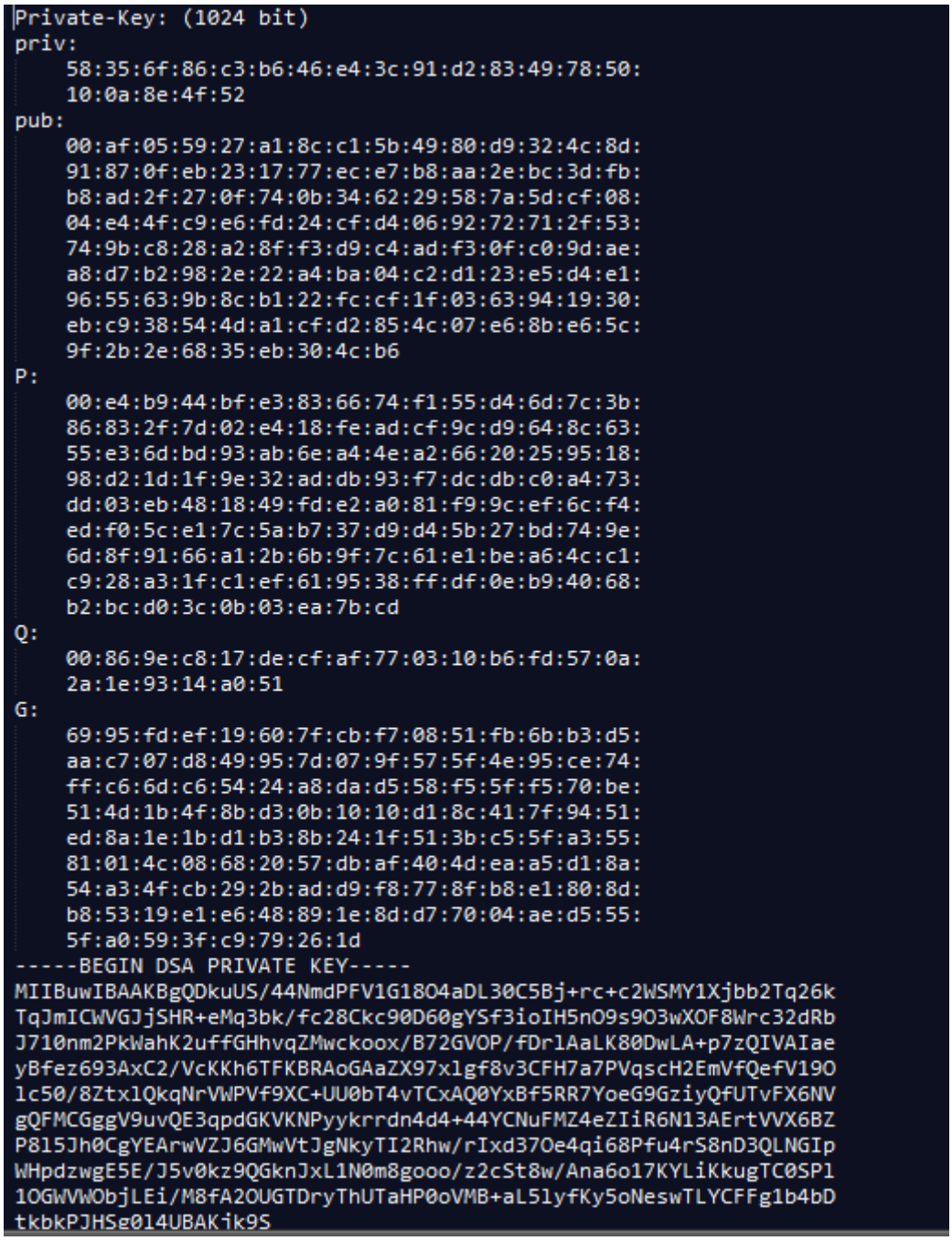
   Para la realización de esta tarea deberemos usar el siguiente comando dos veces, una para miguelDSA.pem y otra para castilloDSA.pem:

   dsa -aes128 -in "C:\Users\Miguemc\Desktop\(miguelDSA.pem|castilloDSA.pem)" -out "C:\Users\Miguemc\Desktop\(miguelDSApriv.pem|castilloDSApriv.pem)"

   La contraseña usada para cifrar con AES128 es la usada en prácticas anteriores “0123456789”

   Los valores de miguelDSApriv.pem y castilloDSApriv.pem respectivamente son:

```
Private-Key: (1024 bit)
priv:
    25:40:92:2f:31:7d:7f:07:27:25:6f:65:b4:0b:aa:
    af:7b:f3:59:34
pub:
    03:52:91:81:10:9a:df:0c:1a:c2:6d:25:0b:46:c4:
    eb:69:1d:ac:96:e6:34:ed:47:3b:5a:e5:7d:9e:d9:
    8f:b1:9e:2a:73:fe:67:9f:40:ff:ba:f6:18:32:bc:
    74:6f:5b:04:28:46:4b:22:76:45:78:b3:74:b8:f7:
    d8:31:6d:3d:3d:82:3f:7c:2a:7e:15:85:f3:a7:6e:
    62:d7:76:c4:d6:a0:a4:83:35:cd:06:ea:ed:81:53:
    44:e0:ae:9a:fe:6e:0c:ec:d9:2e:fc:13:d1:01:bd:
    8a:6d:dc:f4:eb:97:aa:6c:36:b0:bb:98:bf:0a:e9:
    7c:35:81:87:1d:0a:e3:cb
P:
    00:e4:b9:44:bf:e3:83:66:74:f1:55:d4:6d:7c:3b:
    86:83:2f:7d:02:e4:18:fe:ad:cf:9c:d9:64:8c:63:
    55:e3:6d:bd:93:ab:6e:a4:4e:a2:66:20:25:95:18:
    98:d2:1d:1f:9e:32:ad:db:93:f7:dc:db:c0:a4:73:
    dd:03:eb:48:18:49:fd:e2:a0:81:f9:9c:ef:6c:f4:
    ed:f0:5c:e1:7c:5a:b7:37:d9:d4:5b:27:bd:74:9e:
    6d:8f:91:66:a1:2b:6b:9f:7c:61:e1:be:a6:4c:c1:
    c9:28:a3:1f:c1:ef:61:95:38:ff:df:0e:b9:40:68:
    b2:bc:d0:3c:0b:03:ea:7b:cd
Q:
    00:86:9e:c8:17:de:cf:af:77:03:10:b6:fd:57:0a:
    2a:1e:93:14:a0:51
G:
    69:95:fd:ef:19:60:7f:cb:f7:08:51:fb:6b:b3:d5:
    aa:c7:07:d8:49:95:7d:07:9f:57:5f:4e:95:ce:74:
    ff:c6:6d:c6:54:24:a8:da:d5:58:f5:5f:f5:70:be:
    51:4d:1b:4f:8b:d3:0b:10:10:d1:8c:41:7f:94:51:
    ed:8a:1e:1b:d1:b3:8b:24:1f:51:3b:c5:5f:a3:55:
    81:01:4c:08:68:20:57:db:af:40:4d:ea:a5:d1:8a:
    54:a3:4f:cb:29:2b:ad:d9:f8:77:8f:b8:e1:80:8d:
    b8:53:19:e1:e6:48:89:1e:8d:d7:70:04:ae:d5:55:
    5f:a0:59:3f:c9:79:26:1d
-----BEGIN DSA PRIVATE KEY-----
MIIBugIBAAKBgQDkuUS/44NmdPFV1G18O4aDL30C5Bj+rc+c2WSMY1Xjbb2Tq26k
TqJmICWVGJjSHR+eMq3bk/fc28Ckc90D60gYSf3ioIH5nO9s9O3wXOF8Wrc32dRb
J710nm2PkWahK2uffGHhvqZMwckoox/B72GVOP/fDrlAaLK80DwLA+p7zQIVAIae
yBfez693AxC2/VcKKh6TFKBRAoGAaZX97xlgf8v3CFH7a7PVqscH2EmVfQefV19O
lc50/8ZtxlQkqNrVWPVf9XC+UU0bT4vTCxAQ0YxBf5RR7YoeG9GziyQfUTvFX6NV
gQFMCGggV9uvQE3qpdGKVKNPyykrrdn4d4+44YCNuFMZ4eZIiR6N13AErtVVX6BZ
P815Jh0CgYADUpGBEJrfDBrCbSULRsTraR2sluY07Uc7WuV9ntmPsZ4qc/5nn0D/
uvYYMrx0b1sEKEZLInZFeLN0uPfYMW09PYI/fCp+FYXzp25i13bE1qCkgzXNBurt
gVNE4K6a/m4M7Nku/BPRAb2Kbdz065eqbDawu5i/Cul8NYGHHQrjywIUJUCSLzF9
fwcnJW9ltAuar3vzWTO=
```



```
Private-Key: (1024 bit)
priv:
    58:35:6f:86:c3:b6:46:e4:3c:91:d2:83:49:78:50:
    10:0a:8e:4f:52
pub:
    00:af:05:59:27:a1:8c:c1:5b:49:80:d9:32:4c:8d:
    91:87:0f:eb:23:17:77:ec:e7:b8:aa:2e:bc:3d:fb:
    b8:ad:2f:27:0f:74:0b:34:62:29:58:7a:5d:cf:08:
    04:e4:4f:c9:e6:fd:24:cf:d4:06:92:72:71:2f:53:
    74:9b:c8:28:a2:8f:f3:d9:c4:ad:f3:0f:c0:9d:ae:
    a8:d7:b2:98:2e:22:a4:ba:04:c2:d1:23:e5:d4:e1:
    96:55:63:9b:8c:b1:22:fc:cf:1f:03:63:94:19:30:
    eb:c9:38:54:4d:a1:cf:d2:85:4c:07:e6:8b:e6:5c:
    9f:2b:2e:68:35:eb:30:4c:b6
P:
    00:e4:b9:44:bf:e3:83:66:74:f1:55:d4:6d:7c:3b:
    86:83:2f:7d:02:e4:18:fe:ad:cf:9c:d9:64:8c:63:
    55:e3:6d:bd:93:ab:6e:a4:4e:a2:66:20:25:95:18:
    98:d2:1d:1f:9e:32:ad:db:93:f7:dc:db:c0:a4:73:
    dd:03:eb:48:18:49:fd:e2:a0:81:f9:9c:ef:6c:f4:
    ed:f0:5c:e1:7c:5a:b7:37:d9:d4:5b:27:bd:74:9e:
    6d:8f:91:66:a1:2b:6b:9f:7c:61:e1:be:a6:4c:c1:
    c9:28:a3:1f:c1:ef:61:95:38:ff:df:0e:b9:40:68:
    b2:bc:d0:3c:0b:03:ea:7b:cd
Q:
    00:86:9e:c8:17:de:cf:af:77:03:10:b6:fd:57:0a:
    2a:1e:93:14:a0:51
G:
    69:95:fd:ef:19:60:7f:cb:f7:08:51:fb:6b:b3:d5:
    aa:c7:07:d8:49:95:7d:07:9f:57:5f:4e:95:ce:74:
    ff:c6:6d:c6:54:24:a8:da:d5:58:f5:5f:f5:70:be:
    51:4d:1b:4f:8b:d3:0b:10:10:d1:8c:41:7f:94:51:
    ed:8a:1e:1b:d1:b3:8b:24:1f:51:3b:c5:5f:a3:55:
    81:01:4c:08:68:20:57:db:af:40:4d:ea:a5:d1:8a:
    54:a3:4f:cb:29:2b:ad:d9:f8:77:8f:b8:e1:80:8d:
    b8:53:19:e1:e6:48:89:1e:8d:d7:70:04:ae:d5:55:
    5f:a0:59:3f:c9:79:26:1d
-----BEGIN DSA PRIVATE KEY-----
MIIBuwIBAAKBgQDkuUS/44NmdPFV1G18O4aDL30C5Bj+rc+c2WSMY1Xjbb2Tq26k
TqJmICWVGJjSHR+eMq3bk/fc28Ckc90D60gYSf3ioIH5nO9s9O3wXOF8Wrc32dRb
J710nm2PkWahK2uffGHhvqZMwckoox/B72GVOP/fDrlAaLK80DwLA+p7zQIVAIae
yBfez693AxC2/VcKKh6TFKBRAoGAaZX97xlgf8v3CFH7a7PVqscH2EmVfQefV19O
lc50/8ZtxlQkqNrVWPVf9XC+UU0bT4vTCxAQ0YxBf5RR7YoeG9GziyQfUTvFX6NV
gQFMCGggV9uvQE3qpdGKVKNPyykrrdn4d4+44YCNuFMZ4eZIiR6N13AErtVVX6BZ
P8l5Jh0CgYEArwVZJ6GMwVtJgNkyTI2Rhw/rIxd37Oe4qi68Pfu4rS8nD3QLNGIp
WHpdzwgE5E/J5v0kz9QGknJxL1N0m8gooo/z2cSt8w/Ana6o17KYLiKkugTC0SP1
1OGWVWObjLEi/M8fA2OUGTDryThUTaHP0oVMB+aL5lyfKy5oNeswTLYCFFg1b4bD
tkbkPJHSg0l4UBAKjk9S
```

4. Extraed en nombreDSApub.pem la clave pública contenida en el archivo nombreDSAkey.pem. De nuevo nombreDSApub.pem no debe estar cifrado ni protegido. Mostrad sus valores. Lo mismo para el archivo apellidoDSAkey.pem.

   Repetimos el mismo proceso que en el apartado 3 pero añadimos el flag –pubout para indicar que queremos la clave pública:

dsa -in "C:\Users\Miguemc\Desktop\(miguelDSA.pem|castilloDSA.pem)" -pubout -out "C:\Users\Miguemc\Desktop\(miguelDSApub.pem|castilloDSApub.pem)"

Los valores de miguelDSApub.pem y castilloDSApub.pem respectivamente son las siguientes:

```
pub:
    03:52:91:81:10:9a:df:0c:1a:c2:6d:25:0b:46:c4:
    eb:69:1d:ac:96:e6:34:ed:47:3b:5a:e5:7d:9e:d9:
    8f:b1:9e:2a:73:fe:67:9f:40:ff:ba:f6:18:32:bc:
    74:6f:5b:04:28:46:4b:22:76:45:78:b3:74:b8:f7:
    d8:31:6d:3d:3d:82:3f:7c:2a:7e:15:85:f3:a7:6e:
    62:d7:76:c4:d6:a0:a4:83:35:cd:06:ea:ed:81:53:
    44:e0:ae:9a:fe:6e:0c:ec:d9:2e:fc:13:d1:01:bd:
    8a:6d:dc:f4:eb:97:aa:6c:36:b0:bb:98:bf:0a:e9:
    7c:35:81:87:1d:0a:e3:cb
P:
    00:e4:b9:44:bf:e3:83:66:74:f1:55:d4:6d:7c:3b:
    86:83:2f:7d:02:e4:18:fe:ad:cf:9c:d9:64:8c:63:
    55:e3:6d:bd:93:ab:6e:a4:4e:a2:66:20:25:95:18:
    98:d2:1d:1f:9e:32:ad:db:93:f7:dc:db:c0:a4:73:
    dd:03:eb:48:18:49:fd:e2:a0:81:f9:9c:ef:6c:f4:
    ed:f0:5c:e1:7c:5a:b7:37:d9:d4:5b:27:bd:74:9e:
    6d:8f:91:66:a1:2b:6b:9f:7c:61:e1:be:a6:4c:c1:
    c9:28:a3:1f:c1:ef:61:95:38:ff:df:0e:b9:40:68:
    b2:bc:d0:3c:0b:03:ea:7b:cd
Q:
    00:86:9e:c8:17:de:cf:af:77:03:10:b6:fd:57:0a:
    2a:1e:93:14:a0:51
G:
    69:95:fd:ef:19:60:7f:cb:f7:08:51:fb:6b:b3:d5:
    aa:c7:07:d8:49:95:7d:07:9f:57:5f:4e:95:ce:74:
    ff:c6:6d:c6:54:24:a8:da:d5:58:f5:5f:f5:70:be:
    51:4d:1b:4f:8b:d3:0b:10:10:d1:8c:41:7f:94:51:
    ed:8a:1e:1b:d1:b3:8b:24:1f:51:3b:c5:5f:a3:55:
    81:01:4c:08:68:20:57:db:af:40:4d:ea:a5:d1:8a:
    54:a3:4f:cb:29:2b:ad:d9:f8:77:8f:b8:e1:80:8d:
    b8:53:19:e1:e6:48:89:1e:8d:d7:70:04:ae:d5:55:
    5f:a0:59:3f:c9:79:26:1d
-----BEGIN PUBLIC KEY-----
MIIBtjCCASsGByqGSM44BAEwggEeAoGBAOS5RL/jg2Z08VXUbXw7hoMvfQLkGP6t
z5zZZIxjVeNtvZOrbqROomYgJZUYmNIdH54yrduT99zbwKRz3QPrSBhJ/eKggfmc
72z07fBc4XxatzfZ1FsnvXSebY+RZqEra598YeG+pkzBySijH8HvYZU4/98OuUBo
srzQPAsD6nvNAhUAhp7IF97Pr3cDELb9VwoqHpMUoFECgYBplf3vGWB/y/cIUftr
s9WqxwfYSZV9B59XX06VznT/xm3GVCSo2tVY9V/1cL5RTRtPi9MLEBDRjEF/lFHt
ih4b0bOLJB9RO8Vfo1WBAUwIaCBX269ATeql0YpUo0/LKSut2fh3j7jhgI24Uxnh
5kiJHo3XcASu1VVfoFk/yXkmHQOBhAACgYADUpGBEJrfDBrCbSULRsTraR2sluY0
7Uc7WuV9ntmPsZ4qc/5nn0D/uvYYMrx0b1sEKEZLInZFeLN0uPfYMW09PYI/fCp+
FYXzp25i13bE1qCkgzXNBurtgVNE4K6a/m4M7Nku/BPRAb2Kbdz065eqbDawu5i/
Cul8NYGHHQrjyw==
-----END PUBLIC KEY-----
```

```
pub:
    00:af:05:59:27:a1:8c:c1:5b:49:80:d9:32:4c:8d:
    91:87:0f:eb:23:17:77:ec:e7:b8:aa:2e:bc:3d:fb:
    b8:ad:2f:27:0f:74:0b:34:62:29:58:7a:5d:cf:08:
    04:e4:4f:c9:e6:fd:24:cf:d4:06:92:72:71:2f:53:
    74:9b:c8:28:a2:8f:f3:d9:c4:ad:f3:0f:c0:9d:ae:
    a8:d7:b2:98:2e:22:a4:ba:04:c2:d1:23:e5:d4:e1:
    96:55:63:9b:8c:b1:22:fc:cf:1f:03:63:94:19:30:
    eb:c9:38:54:4d:a1:cf:d2:85:4c:07:e6:8b:e6:5c:
    9f:2b:2e:68:35:eb:30:4c:b6
P:
    00:e4:b9:44:bf:e3:83:66:74:f1:55:d4:6d:7c:3b:
    86:83:2f:7d:02:e4:18:fe:ad:cf:9c:d9:64:8c:63:
    55:e3:6d:bd:93:ab:6e:a4:4e:a2:66:20:25:95:18:
    98:d2:1d:1f:9e:32:ad:db:93:f7:dc:db:c0:a4:73:
    dd:03:eb:48:18:49:fd:e2:a0:81:f9:9c:ef:6c:f4:
    ed:f0:5c:e1:7c:5a:b7:37:d9:d4:5b:27:bd:74:9e:
    6d:8f:91:66:a1:2b:6b:9f:7c:61:e1:be:a6:4c:c1:
    c9:28:a3:1f:c1:ef:61:95:38:ff:df:0e:b9:40:68:
    b2:bc:d0:3c:0b:03:ea:7b:cd
Q:
    00:86:9e:c8:17:de:cf:af:77:03:10:b6:fd:57:0a:
    2a:1e:93:14:a0:51
G:
    69:95:fd:ef:19:60:7f:cb:f7:08:51:fb:6b:b3:d5:
    aa:c7:07:d8:49:95:7d:07:9f:57:5f:4e:95:ce:74:
    ff:c6:6d:c6:54:24:a8:da:d5:58:f5:5f:f5:70:be:
    51:4d:1b:4f:8b:d3:0b:10:10:d1:8c:41:7f:94:51:
    ed:8a:1e:1b:d1:b3:8b:24:1f:51:3b:c5:5f:a3:55:
    81:01:4c:08:68:20:57:db:af:40:4d:ea:a5:d1:8a:
    54:a3:4f:cb:29:2b:ad:d9:f8:77:8f:b8:e1:80:8d:
    b8:53:19:e1:e6:48:89:1e:8d:d7:70:04:ae:d5:55:
    5f:a0:59:3f:c9:79:26:1d
-----BEGIN PUBLIC KEY-----
MIIBtzCCASsGByqGSM44BAEwggEeAoGBAOS5RL/jg2Z08VXUbXw7hoMvfQLkGP6t
z5zZZIxjVeNtvZOrbqROomYgJZUYmNIdH54yrduT99zbwKRz3QPrSBhJ/eKggfmc
72z07fBc4XxatzfZ1FsnvXSebY+RZqEra598YeG+pkzBySijH8HvYZU4/98OuUBo
srzQPAsD6nvNAhUAhp7IF97Pr3cDELb9VwoqHpMUoFECgYBplf3vGWB/y/cIUftr
s9WqxwfYSZV9B59XX06VznT/xm3GVCSo2tVY9V/1cL5RTRtPi9MLEBDRjEF/1FHt
ih4b0bOLJB9RO8Vfo1WBAUwIaCBX269ATeql0YpUo0/LKSut2fh3j7jhgI24Uxnh
5kiJHo3XcASu1VVfoFk/yXkmHQOBhQACgYEArwVZJ6GMwVtJgNkyTI2Rhw/rIxd3
7Oe4qi68Pfu4rS8nD3QLNGIpWHpdzwgE5E/J5v0kz9QGknJxL1N0m8gooo/z2cSt
8w/Ana6o17KYLiKkugTC0SP11OGWVWObjLEi/M8fA2OUGTDryThUTaHP0oVMB+aL
5lyfKy5oNeswTLY=
-----END PUBLIC KEY-----
```

5. Calculad el valor hash del archivo con la clave pública nombreDSApub.pem usando sha384 con salida hexadecimal con bloques de dos caracteres separados por dos puntos. Mostrad los valores por salida estándar y guardadlo en nombreDSApub.sha384.

   Para hacer esto y mostrar la salida por pantalla será necesario usar el siguiente comando:

   dgst -sha384 -hex -c "C:\Users\Miguemc\Desktop\miguelDSApub.pem"

   Cuya salida es la siguiente:

```
OpenSSL> dgst -sha384 -hex -c "C:\Users\Miguemc\Desktop\miguelDSApub.pem"
SHA384(C:\Users\Miguemc\Desktop\miguelDSApub.pem)= 90:31:68:85:6b:00:2b:92:
60:e7:67:ec:da:b6:ef:30:53:a7:5c:ed:18:29:1e:10:dc:f0:b1:d9:84:7a:90:c0:8c:
22:b0:4f:68:2c:e9:90:33:33:65:ea:4d:6a:ff:83
OpenSSL>
```

En cambio si lo que queremos es que la salida se guarde en el archivo especificado, entonces el comando a usar es el siguiente:

dgst -sha384 -hex -c -out "C:\Users\Miguemc\Desktop\miguelDSApub.sha384" "C:\Users\Miguemc\Desktop\miguelDSApub.pem"

Cuya salida es la misma, pero en el archivo especificado después de –out.

6. Calculad el valor hash del archivo con la clave pública apellidoDSApub.pem usando una función hash de 160 bits con salida binaria. Guardad el hash en apellidoDSApub.[algoritmo] y mostrad su contenido.

   Para esta tarea, como función hash de 160 bits hemos escogido SHA1 la cual usada en el siguiente comando hace justo lo que buscamos.

   dgst -sha1 -binary -r -out "C:\Users\Miguemc\Desktop\castilloDSApub.sha1" "C:\Users\Miguemc\Desktop\castilloDSApub.pem"

   El contenido de la salida es el siguiente:

```
9568 3fc5 00e5 601a 3b59 d8cd 15d7 5a54
4e96 032b
```

7. Generad el valor HMAC del archivo sharedDSA.pem con clave '12345' mostrándolo por pantalla.

   Para esta tarea emplearemos el siguiente comando, cuya salida es mostrada por pantalla:

```
OpenSSL> dgst -hmac 12345 "C:\Users\Miguemc\Desktop\sharedD
SA.pem"
HMAC-SHA256(C:\Users\Miguemc\Desktop\sharedDSA.pem)= 16e3e4
745c4121aa43790700eb16fe83c9244297a340f274ebd305580acb7ed4
OpenSSL>
```

8. Simulad una ejecución completa del protocolo Estación a Estación. Para ello emplearemos como claves para firma/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. Por ejemplo, si mi clave privada está en javierECpriv.pem y la clave pública del otro usuario está en lobilloECpub.pem, el comando para generar la clave derivada será
$> openssl pkeyutl -inkey javierECpriv.pem -peerkey lobilloECpub.pem -derive –out key.bin
El algoritmo simétrico a utilizar en el protocolo estación a estación será AES-128 en modo CFB8.

El protocolo Estación a Estación es una mejora del algoritmo de Diffie-Hellman, buscando resolver su debilidad frente a ataques de "hombre en el medio", los participantes en la conversación son en este caso Miguel y Castillo, los pasos a seguir son los siguientes:

1- Miguel calcula la clave derivada k usando su clave privada (miguelECpriv.pem) y la pública de Castillo (castilloECpub.pem) usando el comando:

   pkeyutl -inkey "C:\Users\Miguemc\Desktop\miguelECpriv.pem" -peerkey "C:\Users\Miguemc\Desktop\castilloECpub.pem" -derive -out "C:\Users\Miguemc\Desktop\key.bin"

2- Miguel firma con su clave privada de DSA (miguelDSApriv.pem) la pareja de claves públicas de Miguel y Castillo (miguelcastillo.pubkeys) usando el siguiente comando:

   dgst -binary -sha384 -sign C:\Users\Miguemc\Desktop\miguelDSApriv.pem" -out "C:\Users\Miguemc\Desktop\miguel.sign" "C:\Users\Miguemc\Desktop\miguelcastillo.pubkeys"

3- Miguel cifra la firma con AES128 en modo CFB8 usando como clave k con el siguiente comando:

   enc -aes-128-cfb8 -iv 0 -kfile "C:\Users\Miguemc\Desktop\key.bin" -in "C:\Users\Miguemc\Desktop\miguel.sign" -out "C:\Users\Miguemc\Desktop\miguel.enc"

4- Miguel envía la firma cifrada (miguel.enc) a Castillo, y este calcula la clave k con su clave privada (castilloECpriv.pem) y la pública de Miguel (miguelECpub.pem) y la usa para desencriptar la firma con los siguientes comandos:

   pkeyutl -inkey "C:\Users\Miguemc\Desktop\castilloECpriv.pem" -peerkey "C:\Users\Miguemc\Desktop\miguelECpub.pem" -derive -out "C:\Users\Miguemc\Desktop\key2.bin"

   enc -d -aes-128-cfb8 -iv 0 -kfile "C:\Users\Miguemc\Desktop\key2.bin" -in "C:\Users\Miguemc\Desktop\miguel.enc" -out "C:\Users\Miguemc\Desktop\miguel.dec"

5- Ahora Castillo, que ya tiene la firma desencriptada (miguel.dec) procede a verificar la firma usando la clave pública de Miguel (miguelDSApub.pem) usando el siguiente comando:

dgst -sha384 -verify "C:\Users\Miguemc\Desktop\miguelDSApub.pem" -signature "C:\Users\Miguemc\Desktop\miguel.sign" "C:\Users\Miguemc\Desktop\miguelcastillo.pubkeys"

Que si el que ha enviado el mensaje ha sido realmente Miguel, entonces la salida debe ser la siguiente:

```
OpenSSL> dgst -sha384 -verify "C:\Users\Miguemc\Desktop\miguelDSApub.pem" -signature "C:\Users
\Miguemc\Desktop\miguel.sign" "C:\Users\Miguemc\Desktop\miguelcastillo.pubkeys"
Verified OK
```

6- Ahora que Castillo sabe que Miguel es quien dice ser, él procede a hacer lo mismo que ha hecho Miguel, como ya tiene k, firma las claves públicas de ambos, y encripta la firma con AES128 en modo CFB8 usando como clave k y se lo envía a Miguel.

dgst -binary -sha384 -sign "C:\Users\Miguemc\Desktop\castilloDSApriv.pem" -out "C:\Users\Miguemc\Desktop\castillo.sign" "C:\Users\Miguemc\Desktop\miguelcastillo.pubkeys"

enc -aes-128-cfb8 -iv 0 -kfile "C:\Users\Miguemc\Desktop\key2.bin" -in "C:\Users\Miguemc\Desktop\castillo.sign" -out "C:\Users\Miguemc\Desktop\castillo.enc"

7- Miguel usa su clave k para desencriptar la firma y proceder a la verificación.

enc -d -aes-128-cfb8 -iv 0 -kfile "C:\Users\Miguemc\Desktop\key.bin" -in "C:\Users\Miguemc\Desktop\castillo.enc" -out "C:\Users\Miguemc\Desktop\castillo.dec"

dgst -sha384 -verify "C:\Users\Miguemc\Desktop\castilloDSApub.pem" -signature "C:\Users\Miguemc\Desktop\castillo.sign" "C:\Users\Miguemc\Desktop\miguelcastillo.pubkeys"

Que si Castillo es quien dice ser debe producir la siguiente salida:

```
OpenSSL> dgst -sha384 -verify "C:\Users\Miguemc\Desktop\castilloDSApub.pem" -signature "C:\Use
rs\Miguemc\Desktop\castillo.sign" "C:\Users\Miguemc\Desktop\miguelcastillo.pubkeys"
Verified OK
```

8- Ahora que ambos saben que la otra persona es quien dice ser, pueden cambiar mensajes de forma segura encriptándolos con AES128 en modo CFB8 y usando como clave k.