Seguridad y protección en Sistemas Informáticos

Práctica 2

Para generar una clave RSA del tamaño requerido (764) he utilizado:

```
openssl genrsa -out nombreRSAkey.pem 768
```

y para extraer la clave pública contenida en el archivo generado con AES-128:

```
openssl rsa -in nombreRSAkey.pem outform PEM -pbout -out nombreRSApub.pem
```

Y la privada:

```
openssl rsa -aes128 -in nombreRSAkey.pem -out nombreRSApriv.pem -outform PEM
```

Al intentar cifrar nuestro archivo binario, nos da un error, puesto que el tamaño de la clave es insuficiente. Para solucionar esto, diseñamos un sistema híbrido.

Para completar los requisitos de la práctica con más rapidez, he diseñado un programa en bash que realiza las tareas pedidas, llamado sistema.sh:

```
#!/bin/bash

echo "Bienvenido al sistema de encriptacion para SPSI."
read -p"Seleccione modo de operacion: " sim
read -p"Mensaje a cifrar: " men

echo $men > cifrar.txt

read -p"Elija un nombre para su texto (con extension): " nombre
echo Archivo $nombre creado

read -p"Elija un nombre para su texto de salida (con extension): " nombre2
echo Archivo $nombre2 creado

touch $nombre
touch $nombre2

num=${#men}

linea1="openssl rand -hex $num "
liena2="echo $sim"

$linea1 > $nombre
$linea2 >> $nombre

touch cifrar.txt
touch mensajecifrado.txt

read -p"Seleccione la clave publica :" pub
read -p"Seleccione la clave privada :" priv

line=$(head -n 1 $nombre)

openssl rsautl -encrypt -inkey $pub -pubin -in $nombre -out $nombre2
openssl rsautl -decrypt -inkey $priv -in $nombre2 -out dec.bin
openssl $sim -in cifrar.txt -out mensajecifrado.txt -pass pass:$line

rm -rf cifrar.txt
```

Como podemos observar, el programa pregunta por un mensaje, un sistema de cifrado de clave privada (*como por ejemplo, des o aes*), y un archivos de input y output. Calcula el tamaño del mensaje y genera una clave hexadeciman en base a esa información, y acto seguido guarda el contenido en la primera línea del archivo que hemos puesto como input.

Después guarda en la segunda línea el tipo de cifrado elegido al inicio. Se selecciona una clave pública, otra privada y encripta y decripta usando los archivos indicados, dejando en dec.bin el resultado desencriptado final.

Por otro lado, cifra el mensaje usando como contraseña la primera línea del primer archivo, es decir, la clave aleatoria hexadecimal. Con este programa podemos **hacer todo a la vez**: La encriptación de mensajes y el cifrado/descifrado de archivos de cualquier extensión que indiquemos.

Para la parte de curvas elípticas, a la hora de consultar las curvas disponibles en openssl utilizamos:

```
openssl ecparam -list_curves
```

Que nos muestra y explica las distintas opciones. Cuando seleccionamos una (por ejemplo secp256k1), escribimos:

```
openssl ecparam -name secp256k1 -out secp256k1.pem
```

Lo que genera un archivo de parámetros para claves EC. Acto seguido, para crear el par de claves en base a estos parámetros, escribimos:

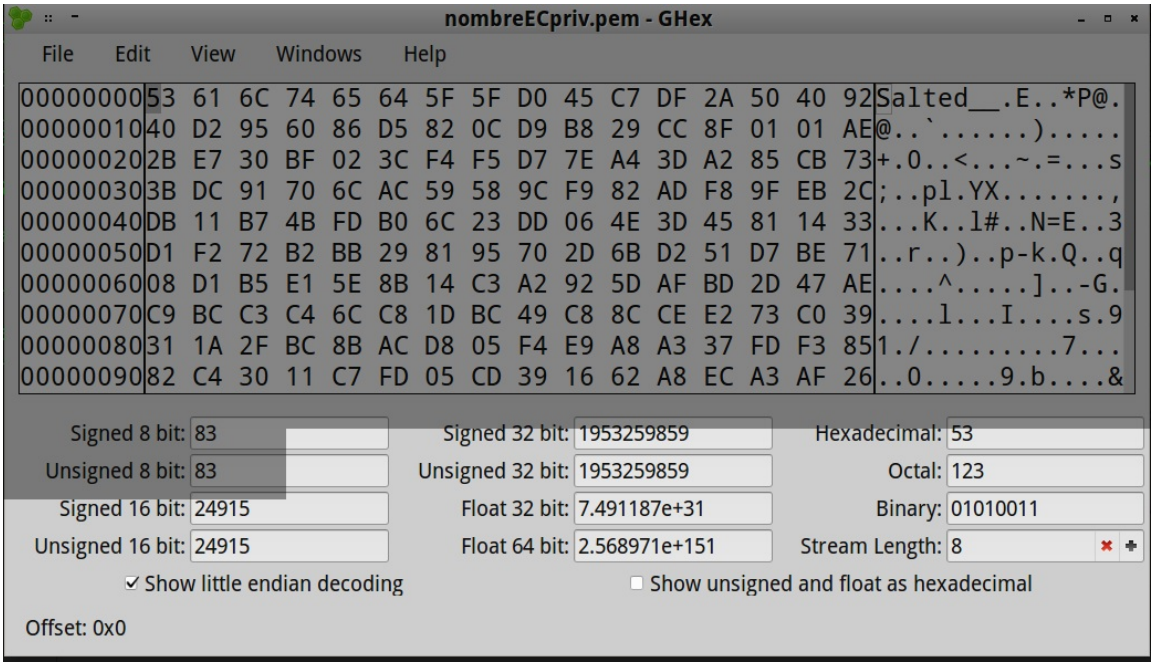
```
openssl ecparam -in secp256k1.pem -genkey -noout -out secp256k1-key.pem
```

O sin parámetros con :

```
openssl ecparam -name secp256k1 -genkey -noout -out nombreECkey.pem
```

Para guardar la clave privada en un archivo nombreECpriv.pem usando des3 y como passphrase "*howallocassionsdoinformagainstme*" usaremos:

```
openssl des3 -salt -k howallocassionsdoinformagainstme < nombreECkey.pem > nombreECpriv.pem
```



Finalmente para extraer la clave pública:

```
openssl exparam -in nombreECkey.pem outform PEM -pbout -out nombreECpub.pem
```