

Criptosistemas Asimétricos

Práctica 2



Índice

[1. Generación y manipulación de claves con RSA 3](#_Toc526937943)

[a. Generación clave privada 3](#_Toc526937944)

[Comando 3](#_Toc526937945)

[Parámetros 3](#_Toc526937946)

[Texto plano 4](#_Toc526937947)

[b. Cifrando la clave privada 4](#_Toc526937948)

[Comando 4](#_Toc526937949)

[Parámetros 5](#_Toc526937950)

[Texto plano 6](#_Toc526937951)

[c. Generar la pública 6](#_Toc526937952)

[Comando 6](#_Toc526937953)

[Texto plano 6](#_Toc526937954)

[d. Cifrando con la clave pública 6](#_Toc526937955)

[Comando 6](#_Toc526937956)

[Error 6](#_Toc526937957)

[2. Cifrado híbrido (Simétrico y asimétrico) 7](#_Toc526937958)

[a. Generando el Archivo sessionkey 7](#_Toc526937959)

[Comando 7](#_Toc526937960)

[Texto plano 7](#_Toc526937961)

[b. Cifrado del archivo a enviar con la clave simétrica almacenada en el sessionkey 7](#_Toc526937962)

[Comando 7](#_Toc526937963)

[Texto plano 7](#_Toc526937964)

[c. Cifrando el archivo sesiónkey con la clave pública del receptor 7](#_Toc526937965)

[Comando 7](#_Toc526937966)

[Texto plano 8](#_Toc526937967)

[d. Descifrando el sessinkey con la clave privada del receptor 8](#_Toc526937968)

[Comando 8](#_Toc526937969)

[Texto Plano 8](#_Toc526937970)

[e. Descifrando el archivo recibido con la clave simétrica almacenada en el sessionkey 8](#_Toc526937971)

[Comando 8](#_Toc526937972)

[Texto Plano 8](#_Toc526937973)

[f. Conclusiones 8](#_Toc526937974)

[3. Curvas elípticas 9](#_Toc526937975)

[a. Encontrar una de las curvas del profesor y sacar los parámetros de esta 9](#_Toc526937976)

[Comando 9](#_Toc526937977)

[Texto plano 10](#_Toc526937978)

[b. Generar la clave 10](#_Toc526937979)

[Comandos 10](#_Toc526937980)

[Texto plano 10](#_Toc526937981)

[c. Extraer la clave privada cifrada 10](#_Toc526937982)

[Comandos 10](#_Toc526937983)

[Texto plano 11](#_Toc526937984)

[d. Extraer la pública 11](#_Toc526937985)

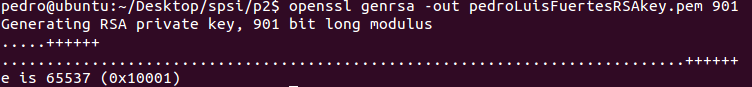
[Comandos 11](#_Toc526937986)

[Texto plano 11](#_Toc526937987)

# Generación y manipulación de claves con RSA

## Generación clave privada

### Comando

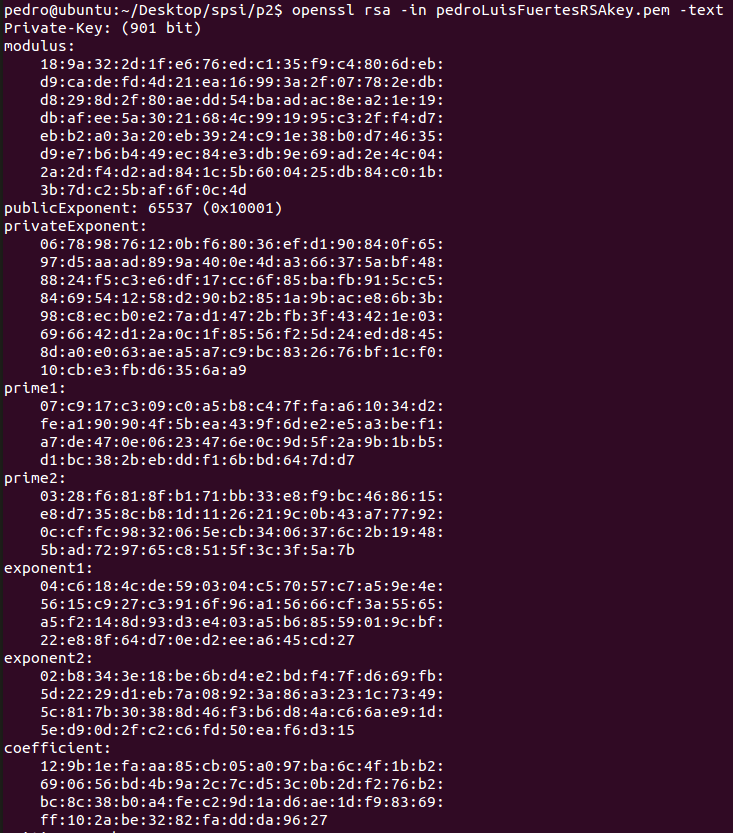


Genrsa: Indica que se va a generar una clave

-out: Es el nombre de salida del archivo

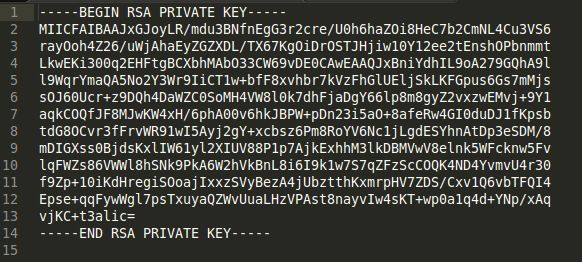
901: Es el número de bits de la clave que se va a generar

### Parámetros



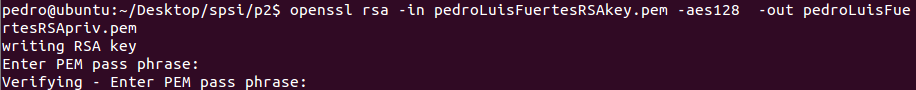


### Texto plano



## Cifrando la clave privada

### Comando

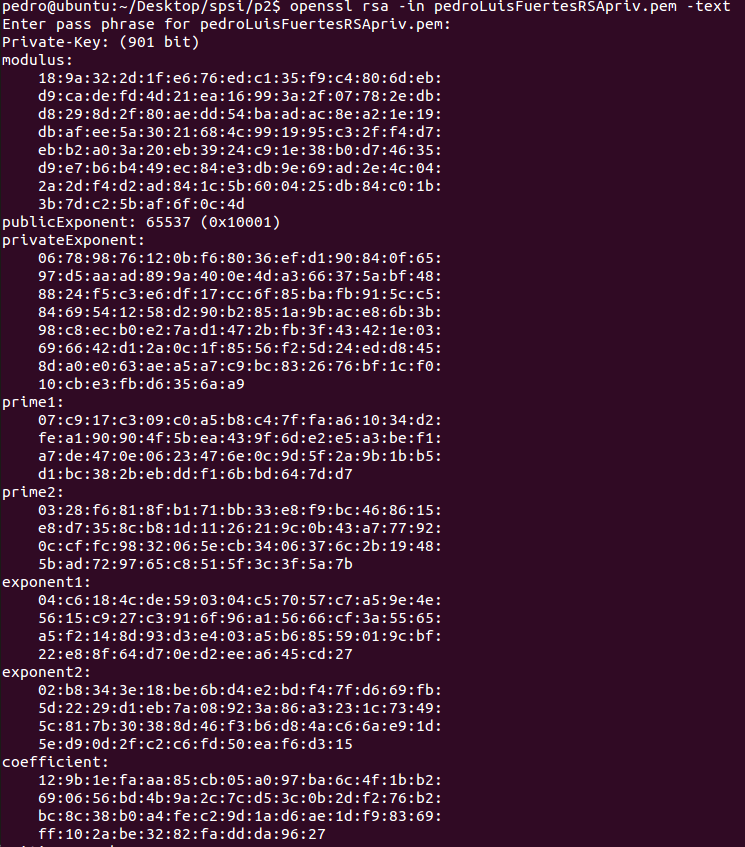


-in: El archivo que se va a cifrar

-aes128: El cifrado que se va a usar

-out: el nombre de salida del archivo ya cifrado

### Parámetros





### Texto plano



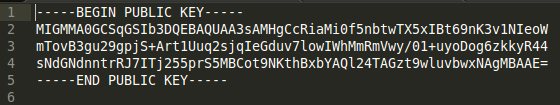
## Generar la pública

### Comando



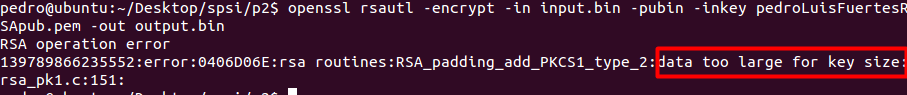
-pubout: Indica que se quiere extraer la clave pública del archivo de entrada

### Texto plano



## Cifrando con la clave pública

### Comando



rsautl: Indica que se va a llevar acciones con una clave.

-encrypt: Indica que se quiere cifrar

-in: El archivo que se quiere cifrar

-pubin: Indica que se va a hacer con clave pública

-inkey: Se le pasa dónde está la clave que se va a usar

-out: El nombre del fichero una vez cifrado.

### Error

Como se puede ver el archivo que se intenta cifrar es demasiado largo.

Esto es debido a que la clave pública está pensada para cifrar archivos pequeños, como claves de sesión, o claves pequeñas de unos 128 o 256 bits, mientras que el archivo que se intenta cifrar es de 1024 bits.

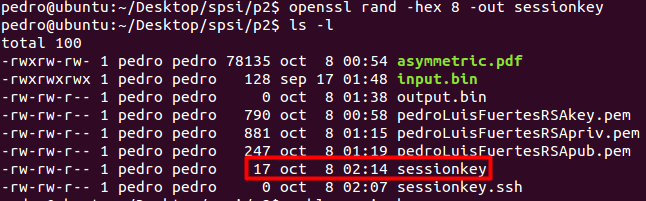
Además, el cifrado asimétrico es realmente ineficiente, por lo que intentar cifrar y descifrar grandes cantidades de datos tomaría demasiado tiempo por lo que no es viable.

Lo que se hace es usar un cifrado simétrico, mucho más eficiente, y cifrar de manera asimétrica la contraseña del cifrado simétrico.

# Cifrado híbrido (Simétrico y asimétrico)

## Generando el Archivo sessionkey

### Comando



Openssl rand: Para generar un archivo aleatorio

-hex: para que el formato sea hexadecmal

8: para decir que sea de 128 bits (8 caracteres x 2 bytes/carácter hexadecimal x 8bit/byte = 128 bits)

-out: indica el nombre del fichero de salida

### Texto plano



\*El método de cifrado se ha añadido a mano

## Cifrado del archivo a enviar con la clave simétrica almacenada en el sessionkey

### Comando



Openssl enc -aes-128-ecb: El tipo de cifrado a usar

-pass file:sessionkey: indica que se va a cifrar con contraseña, y que esta se encuentra en un archivo y el nombre del archivo donde se encuentra

### Texto plano



Como se puede ver el output no tiene nada que ver con el input, además, se usa salted ya que se usa una contraseña.

## Cifrando el archivo sesiónkey con la clave pública del receptor

### Comando



Openssl rsautl: Para decir que se va a usar una de las opciones de RSA

-encrypt: Para decir que se va a encriptar

-in: el archivo de entrada que se quiere cifrar, sessinkey en este caso.

-pubin: indica que se va a cifrar con una clave pública

-inkey: archivo donde se encuentra la clave pública.

-out: nombre del fichero de salida

### Texto plano



Como se puede ver, el sessionkey queda cifrado

## Descifrando el sessinkey con la clave privada del receptor

### Comando



-decrypt: Para indicar que se va a descifrar.

-inkey: es la clave privada del receptor

-in: seria el archivo sessionkey cifrado

-out: daría el archivo descifrado

\*Notar que ya no se usa la opción – pubin, ya que se usa la clave privada.

### Texto Plano



Como se puede ver, se descifra correctamente

## Descifrando el archivo recibido con la clave simétrica almacenada en el sessionkey

### Comando

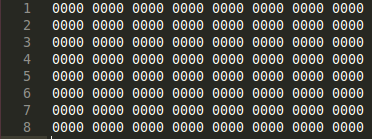


-d: indica que se quiere descifrar

-in: seria el archivo cifrado

-out: daría el archivo descifrado, input\_ecib.bin

### Texto Plano



## Conclusiones

Como se puede ver, esto permite transmitir datos de manera segura por redes no seguras.

Además, evitamos la limitación que imponen los sistemas asimétricos para archivos de gran tamaño, ya que estos se cifran con sistemas simétricos, con una contraseña de sesión, cifrando únicamente de manera asimétrica la contraseña y el tipo de cifrado que se ha usado.

Como se cifra con la pública del receptor, el único capaz de descifrar la contraseña de sesión será el propio receptor y por tanto el único capaz de descifrar todos los archivos.

# Curvas elípticas

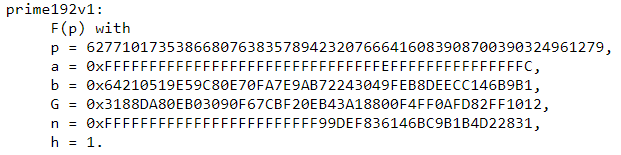
## Encontrar una de las curvas del profesor y sacar los parámetros de esta



La curva de las diapositivas P-192 cuyos parámetros son:

* P = 6277101735386680763835789423207666416083908700390324961279
* Β = 0x 64210519 e59c80e7 0fa7e9ab 72243049 feb8deec c146b9b1

Se corresponde con la curva prime192v1:



### Comando



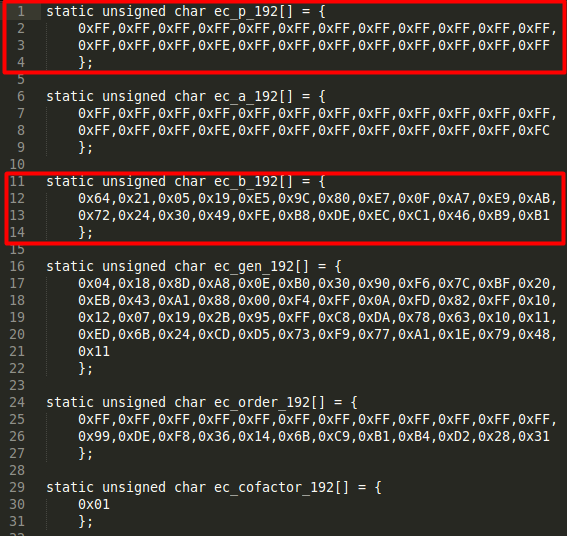
Oppenssl ecparam: Indica que se van a usar parámetros de curvas elípticas

-name: especifica el nombre de la curva

-C: genera los parámetros de la curva

-out: fichero donde se van a almacenar los parámetros

### Texto plano



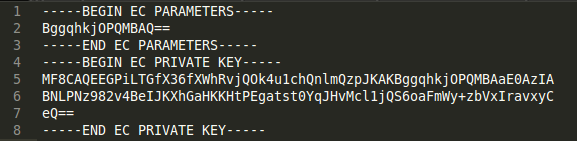
## Generar la clave

### Comandos



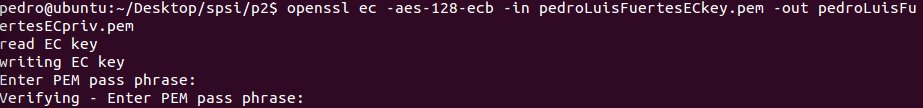
-genkey: genera la clave

### Texto plano



## Extraer la clave privada cifrada

### Comandos



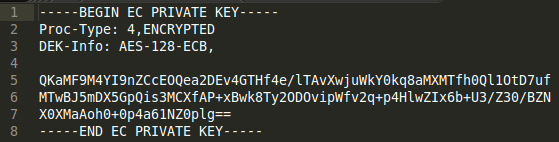
Openssl ec: indica que se va a trabajar con curvas elípticas

-aes-128-ecb: tipo de cifrado

-in: archivo donde está la clave

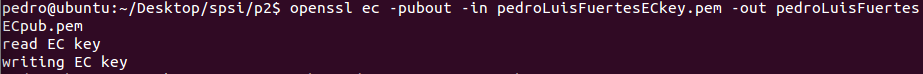
-out: el archivo con la clave privada cifrada

### Texto plano



## Extraer la pública

### Comandos



Openssl ec: indica que se va a trabajar con curvas elípticas

-pubout: para indicar que quieres obtener la clave pública

-in: archivo donde está la clave

-out: el archivo con la clave pública

### Texto plano

