Trabajo Clase de Criptografía y RSA

Criptografía y Seguridad Informática (86.36) - 1°C 2023

Integrantes	**Padrón**
del Mazo, Federico	100029
Galán, Santiago	101248
Pardo, Lucía	99999
Lafroce, Matías	91378
Rodríguez, Florencia	100033

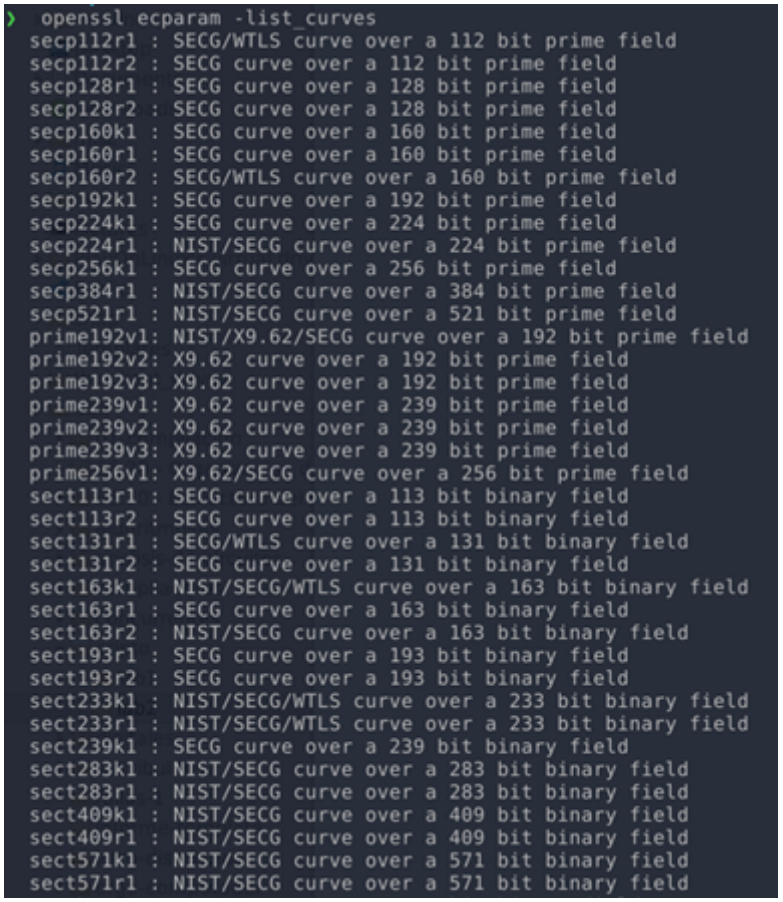
Resolución de ejercicios

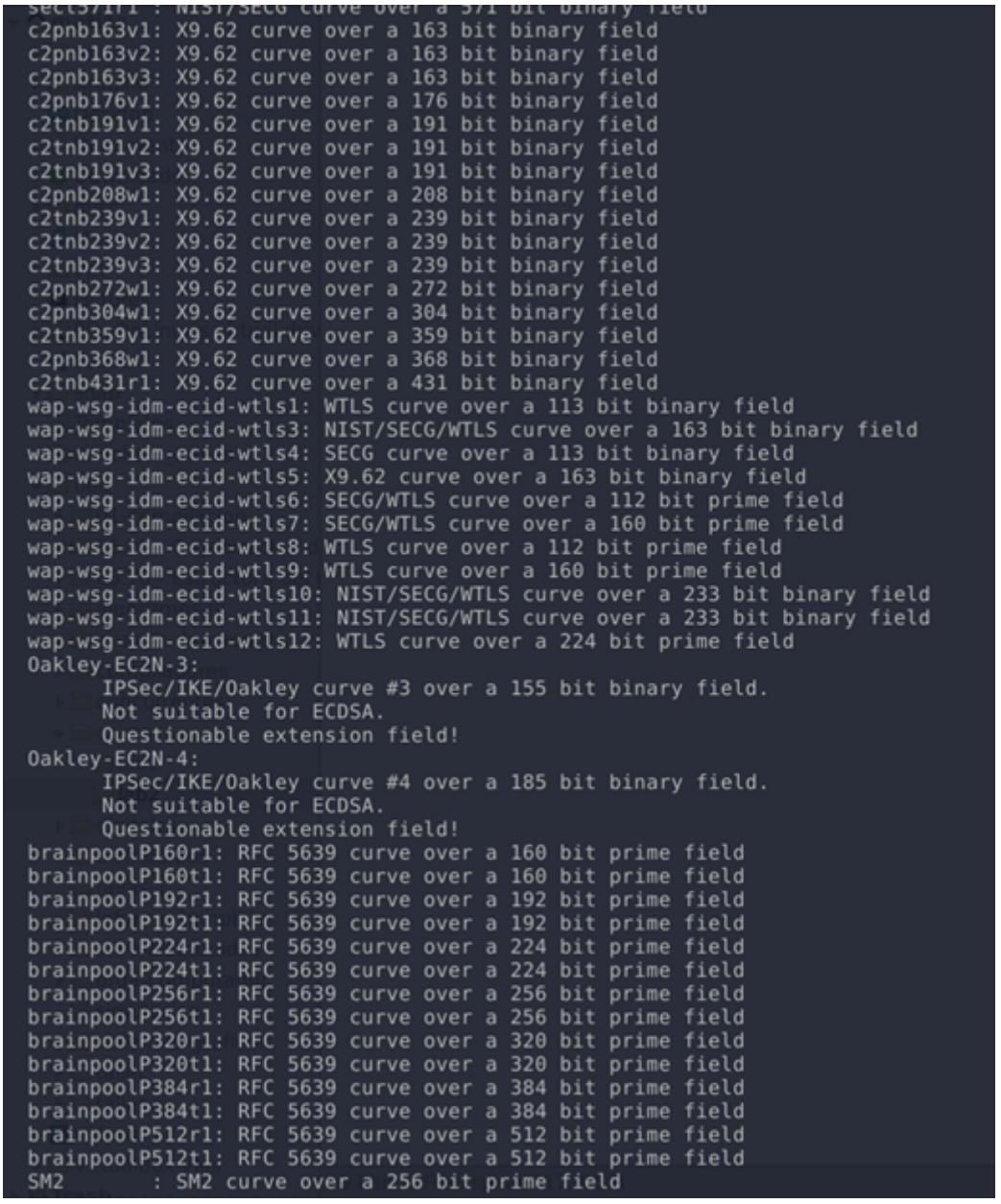
ECCDSA con openSSL

- ***Ejecutar el comando***

openssl ecparam -list_curves

- ***¿Qué muestra?***





- ***¿Cuáles son las curvas que soporta TLS?***

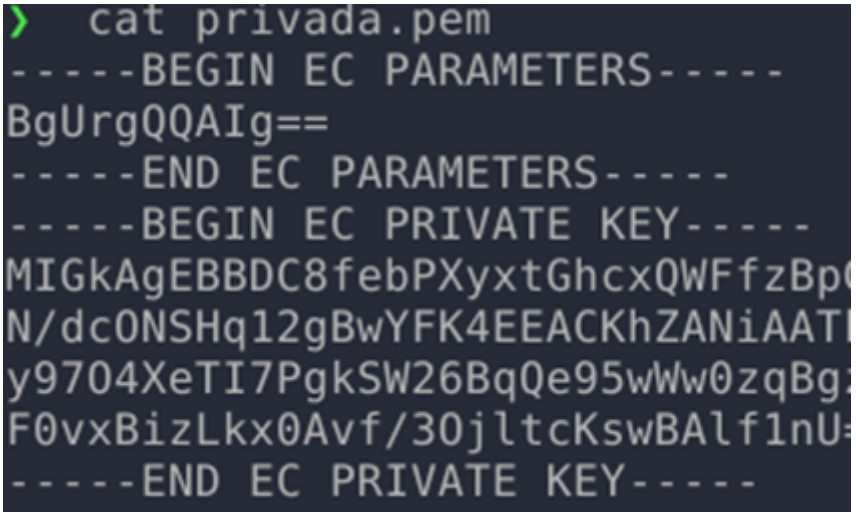
En la documentación se mencionan 2 curvas:

- P-256 nistp256 secp256r1
- P-384 nistp384 secp384r1

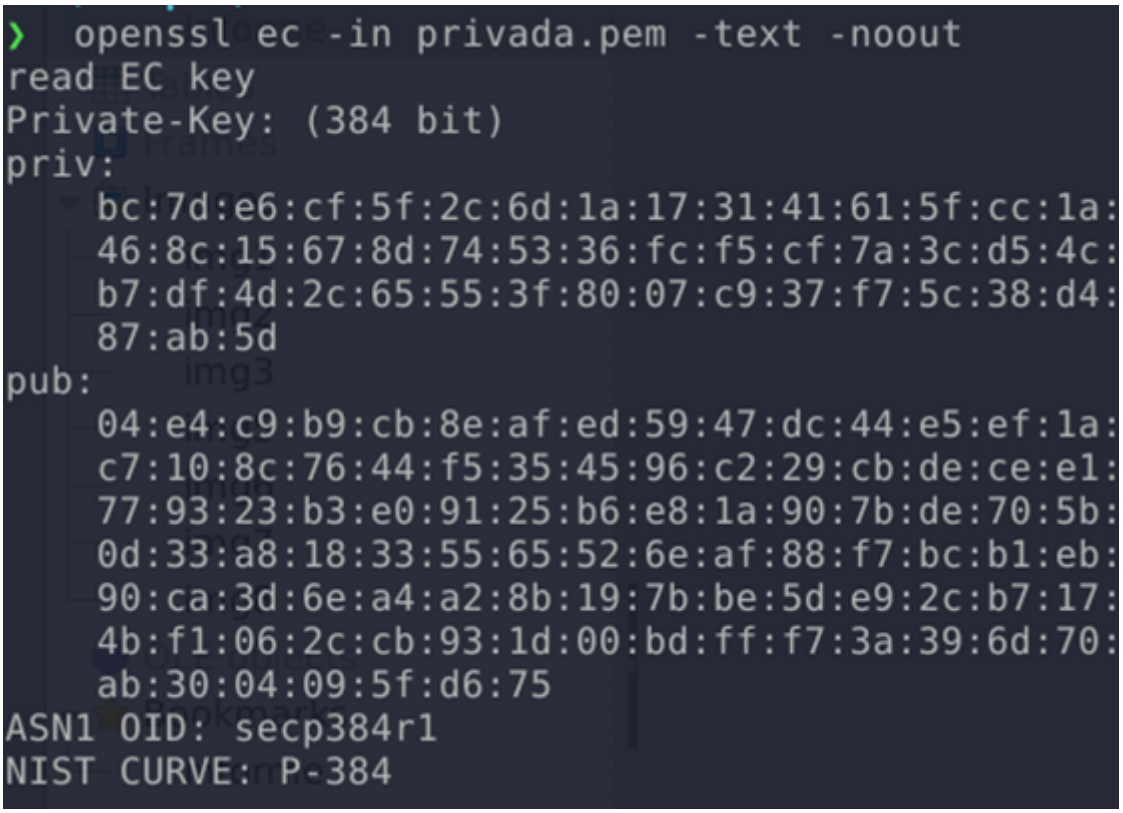
En la lista de curvas soportadas por openSSL vemos que sólo aparece la segunda.

- **Generar las claves pública y privada en la curva secp384r1**

Ejecutamos el comando y el archivo generado contiene:



- **Imprimir las claves pública y privada en formato texto**



- ***Investigar la salida del comando siguiente e indicar que son dichos parámetros***

openssl ecparam -in privada.pem -text -param_enc explicit -noout

El comando openssl ecparam nos permite inspeccionar los parámetros de curvas elípticas. Las opciones que tenemos son:

- -in privada.pem : Especificamos la clave privada de entrada
- -text : Se emite texto plano.
- -param_enc explicit: Muestra los parámetros de la curva. El default es named_curve que sólo muestra el nombre de una curva predefinida. named_curve_explicit muestra ambos datos
- -noout: no mostrar la salida standard con los parámetros codificados

- ***Extraer la clave pública en el archivo publica.pem***

Generamos la firma pública utilizando la opción -pubout

```
❯  openssl ec -in privada.pem -pubout > publica.pem
read EC key
writing EC key

uba/cripto/lab2
❯  cat publica.pem
-----BEGIN PUBLIC KEY-----
MHYwEAYHKoZIzj0CAQYFK4EEACIDYgAE5Mm5y46v7VlH3ETl7xrHEIx2RP
KcvezuF3kyOz4JEltugakHvecFsNM6gYM1VlUm6viPe8seuQyj1upKKLGX
txdL8QYsy5MdAL3/9zo5bXCrMAQJX9Z1
-----END PUBLIC KEY-----
```

- **Firmar el archivo (generar archivo firma)**

Realizamos un digest usando el algoritmo sha256 de nuestro archivo “datos”

```
uba/cripto/lab2
❯  openssl dgst -sha256 -sign privada.pem datos > firma

uba/cripto/lab2
❯  cat firma
0e1T}`
        Y      l  a:jpB v  q gM.=x    0IF   Hc-  W  &:  V  cE
```

- **Pasar a firma base64**

```
uba/cripto/lab2
❯  base64 firma > firmab64

uba/cripto/lab2
❯  cat firmab64
MGUCMQCDqFR9YI/ZC91Z4p/gBPx/EMAR/9Bsg7JhOmpwQr52Ft9//3ERm2dNLj146f2ulg8CMElG
lIGbSGMtrY0PVyCAJjr8zLxWFsUeY0W7WL1CymGcpP3/tHGGycvaB5ArmmKTuA==
```

- **Verificar el archivo datos, modificar, y verificar nuevamente**

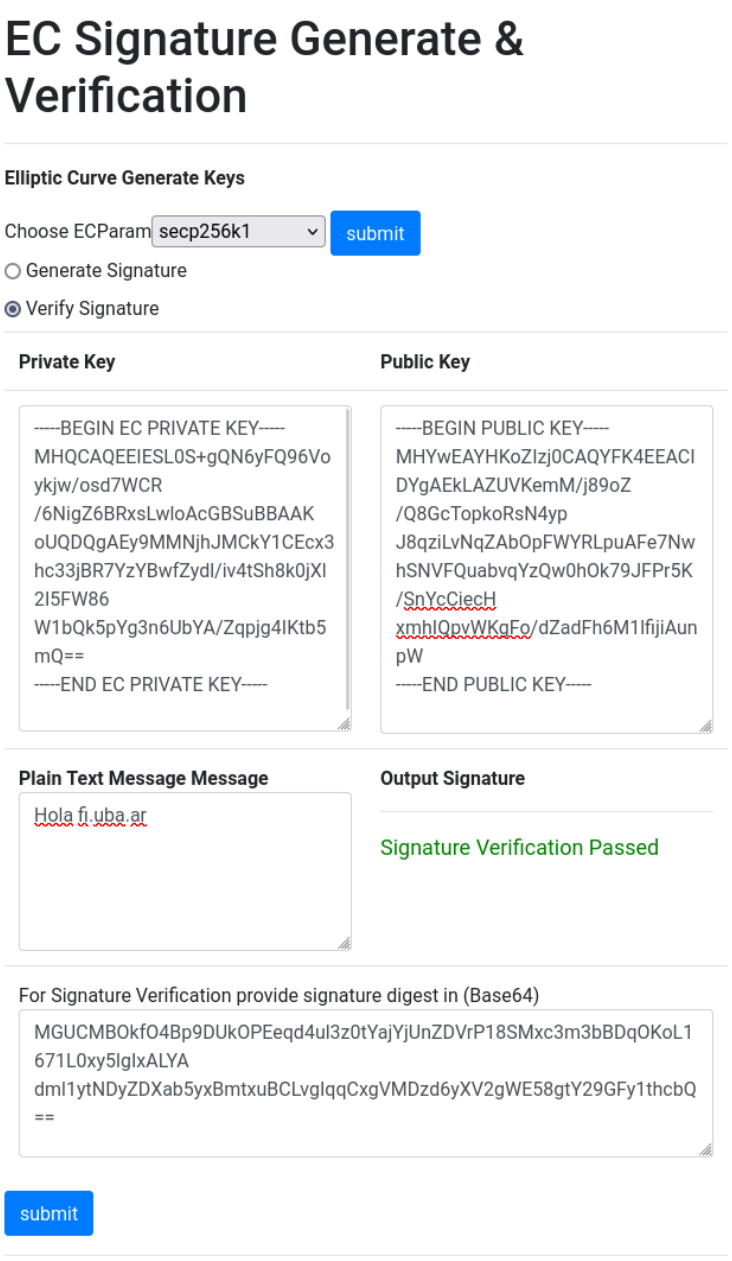
```
uba/cripto/lab2
❯  openssl dgst -sha256 -verify publica.pem -signature firma datos
Verified OK

uba/cripto/lab2
❯  echo " modified" >> datos

uba/cripto/lab2
❯  openssl dgst -sha256 -verify publica.pem -signature firma datos
Verification failure
```

Verificación de la firma por “un tercero”

Para este ejercicio se debe entrar a:

- https://8gwifi.org/ecsignverify.jsp
- Seleccionar “Verify Signature
- Poner la clave pública, el mensaje en texto plano y la firma • Ver que la firma verifica!

Se muestra a continuación el resultado de la acción.



HASH 2

Leer las consideraciones del documento incibe_toma_evidencias_analisis_forense.pdf

¿Cómo se usa un HASH en una pericia? ¿Qué HASH usaría para una pericia informática? (md5, sha1, sha256, etc)

En una pericia informática se obtiene un hash del volcado de la memoria física para garantizar su integridad.
Es importante que este hash no presente colisiones ya que eso deslegitimaría la validez de las pruebas. Es por esto que no se recomienda usar ni md5 ni SHA-1. Una recomendación sería usar SHA-256 o incluso SHA-512