

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?**

```
> openssl ecparam -list_curves
secp112r1 : SECG/WTLS curve over a 112 bit prime field
secp112r2 : SECG curve over a 112 bit prime field
secp128r1 : SECG curve over a 128 bit prime field
secp128r2 : SECG curve over a 128 bit prime field
secp160k1 : SECG curve over a 160 bit prime field
secp160r1 : SECG curve over a 160 bit prime field
secp160r2 : SECG/WTLS curve over a 160 bit prime field
secp192k1 : SECG curve over a 192 bit prime field
secp224k1 : SECG curve over a 224 bit prime field
secp224r1 : NIST/SECG curve over a 224 bit prime field
secp256k1 : SECG curve over a 256 bit prime field
secp384r1 : NIST/SECG curve over a 384 bit prime field
secp521r1 : NIST/SECG curve over a 521 bit prime field
prime192v1: NIST/X9.62/SECG curve over a 192 bit prime field
prime192v2: X9.62 curve over a 192 bit prime field
prime192v3: X9.62 curve over a 192 bit prime field
prime239v1: X9.62 curve over a 239 bit prime field
prime239v2: X9.62 curve over a 239 bit prime field
prime239v3: X9.62 curve over a 239 bit prime field
prime256v1: X9.62/SECG curve over a 256 bit prime field
sect113r1 : SECG curve over a 113 bit binary field
sect113r2 : SECG curve over a 113 bit binary field
sect131r1 : SECG/WTLS curve over a 131 bit binary field
sect131r2 : SECG curve over a 131 bit binary field
sect163k1 : NIST/SECG/WTLS curve over a 163 bit binary field
sect163r1 : SECG curve over a 163 bit binary field
sect163r2 : NIST/SECG curve over a 163 bit binary field
sect193r1 : SECG curve over a 193 bit binary field
sect193r2 : SECG curve over a 193 bit binary field
sect233k1 : NIST/SECG/WTLS curve over a 233 bit binary field
sect233r1 : NIST/SECG/WTLS curve over a 233 bit binary field
sect239k1 : SECG curve over a 239 bit binary field
sect283k1 : NIST/SECG curve over a 283 bit binary field
sect283r1 : NIST/SECG curve over a 283 bit binary field
sect409k1 : NIST/SECG curve over a 409 bit binary field
sect409r1 : NIST/SECG curve over a 409 bit binary field
sect571k1 : NIST/SECG curve over a 571 bit binary field
sect571r1 : NIST/SECG curve over a 571 bit binary field
```

```

secp371r1: NIST/SEC2 curve over a 371 bit binary field
c2pnb163v1: X9.62 curve over a 163 bit binary field
c2pnb163v2: X9.62 curve over a 163 bit binary field
c2pnb163v3: X9.62 curve over a 163 bit binary field
c2pnb176v1: X9.62 curve over a 176 bit binary field
c2tnb191v1: X9.62 curve over a 191 bit binary field
c2tnb191v2: X9.62 curve over a 191 bit binary field
c2tnb191v3: X9.62 curve over a 191 bit binary field
c2pnb208w1: X9.62 curve over a 208 bit binary field
c2tnb239v1: X9.62 curve over a 239 bit binary field
c2tnb239v2: X9.62 curve over a 239 bit binary field
c2tnb239v3: X9.62 curve over a 239 bit binary field
c2pnb272w1: X9.62 curve over a 272 bit binary field
c2pnb304w1: X9.62 curve over a 304 bit binary field
c2tnb359v1: X9.62 curve over a 359 bit binary field
c2pnb368w1: X9.62 curve over a 368 bit binary field
c2tnb431r1: X9.62 curve over a 431 bit binary field
wap-wsg-idm-ecid-wtls1: WTLS curve over a 113 bit binary field
wap-wsg-idm-ecid-wtls3: NIST/SEC2/WTLS curve over a 163 bit binary field
wap-wsg-idm-ecid-wtls4: SEC2 curve over a 113 bit binary field
wap-wsg-idm-ecid-wtls5: X9.62 curve over a 163 bit binary field
wap-wsg-idm-ecid-wtls6: SEC2/WTLS curve over a 112 bit prime field
wap-wsg-idm-ecid-wtls7: SEC2/WTLS curve over a 160 bit prime field
wap-wsg-idm-ecid-wtls8: WTLS curve over a 112 bit prime field
wap-wsg-idm-ecid-wtls9: WTLS curve over a 160 bit prime field
wap-wsg-idm-ecid-wtls10: NIST/SEC2/WTLS curve over a 233 bit binary field
wap-wsg-idm-ecid-wtls11: NIST/SEC2/WTLS curve over a 233 bit binary field
wap-wsg-idm-ecid-wtls12: WTLS curve over a 224 bit prime field
Oakley-EC2N-3:
    IPSec/IKE/Oakley curve #3 over a 155 bit binary field.
    Not suitable for ECDSA.
    Questionable extension field!
Oakley-EC2N-4:
    IPSec/IKE/Oakley curve #4 over a 185 bit binary field.
    Not suitable for ECDSA.
    Questionable extension field!
brainpoolP160r1: RFC 5639 curve over a 160 bit prime field
brainpoolP160t1: RFC 5639 curve over a 160 bit prime field
brainpoolP192r1: RFC 5639 curve over a 192 bit prime field
brainpoolP192t1: RFC 5639 curve over a 192 bit prime field
brainpoolP224r1: RFC 5639 curve over a 224 bit prime field
brainpoolP224t1: RFC 5639 curve over a 224 bit prime field
brainpoolP256r1: RFC 5639 curve over a 256 bit prime field
brainpoolP256t1: RFC 5639 curve over a 256 bit prime field
brainpoolP320r1: RFC 5639 curve over a 320 bit prime field
brainpoolP320t1: RFC 5639 curve over a 320 bit prime field
brainpoolP384r1: RFC 5639 curve over a 384 bit prime field
brainpoolP384t1: RFC 5639 curve over a 384 bit prime field
brainpoolP512r1: RFC 5639 curve over a 512 bit prime field
brainpoolP512t1: RFC 5639 curve over a 512 bit prime field
SM2 : SM2 curve over a 256 bit prime field

```

- **¿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:

```
> cat privada.pem
-----BEGIN EC PARAMETERS-----
BgUrgQQAIg==
-----END EC PARAMETERS-----
-----BEGIN EC PRIVATE KEY-----
MIGkAgEBBDC8febPXyxtGhcxQWFfzBp0
N/dcONSHq12gBwYFK4EEACKhZANiAAT
y9704XeTI7PgkSW26BqQe95wWw0zqBg
F0vxBizLkx0Avf/30jltcKswBALf1nU
-----END EC PRIVATE KEY-----
```

- Imprimir las claves pública y privada en formato texto

```
> openssl ec -in privada.pem -text -noout
read EC key
Private-Key: (384 bit)
priv:
    bc:7d:e6:cf:5f:2c:6d:1a:17:31:41:61:5f:cc:1a:
    46:8c:15:67:8d:74:53:36:fc:f5:cf:7a:3c:d5:4c:
    b7:df:4d:2c:65:55:3f:80:07:c9:37:f7:5c:38:d4:
    87:ab:5d
pub:
    04:e4:c9:b9:cb:8e:af:ed:59:47:dc:44:e5:ef:1a:
    c7:10:8c:76:44:f5:35:45:96:c2:29:cb:de:ce:e1:
    77:93:23:b3:e0:91:25:b6:e8:1a:90:7b:de:70:5b:
    0d:33:a8:18:33:55:65:52:6e:af:88:f7:bc:b1:eb:
    90:ca:3d:6e:a4:a2:8b:19:7b:be:5d:e9:2c:b7:17:
    4b:f1:06:2c:cb:93:1d:00:bd:ff:f7:3a:39:6d:70:
    ab:30:04:09:5f:d6:75
ASN1 OID: secp384r1
NIST CURVE: P-384
```

- *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-----
MHYwEAYHKoZIzj0CAQYFK4EEACIDYgAE5Mm5y46v7VlH3ETl7xrHEIx2RF
KcvezuF3ky0z4JEl tugakHvecFsNM6gYM1VlUm6viPe8seuQyj1upKKLGX
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
0e100T}`00
0Y000000l00a:jpB0v00q0gM.=x00000IF000Hc-00W 0&:0V0cE0)
```

- **Pasar a firma base64**

```
uba/cripto/lab2
> base64 firma > firmab64
uba/cripto/lab2
> cat firmab64
MGUCMQCDqFR9YI/ZC91Z4p/gBPx/EMAR/9Bsg7Jh0mpwQr52Ft9//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.

EC Signature Generate & Verification

Elliptic Curve Generate Keys

Choose ECParm secp256k1

submit

☐ Generate Signature

☒ Verify Signature

Private Key

```
-----BEGIN EC PRIVATE KEY-----
MHQCAQEEIESL0S+gQN6yFQ96Vo
ykw/osd7WCR
/6NigZ6BRxslWloAcGBSuBBAK
oUQDQgAEy9MMNjhJMCKY1CEcx3
hc33jBR7YzYBwfZydl/iv4tSh8k0jXl
2I5FW86
W1bQk5pYg3n6UbYA/Zqpg4IKtb5
mQ==
-----END EC PRIVATE KEY-----
```

Public Key

```
-----BEGIN PUBLIC KEY-----
MHYwEAYHKoZIzj0CAQYFK4EEACI
DYgAEkLAZUVKernM/j89oZ
/Q8GcTopkoRsN4yp
J8qziLvNqZAbOpFWYRLpuAF7Nw
hSNVFQuabvqYzQw0hOk79JFPr5K
/SnYcCiech
xmhlQpvWKgFo/dZadFh6M1IfijiAun
pW
-----END PUBLIC KEY-----
```

Plain Text Message Message

Hola f. uba.ar

Output Signature

Signature Verification Passed

For Signature Verification provide signature digest in (Base64)

```
MGUCMBOkfO4Bp9DUkOPEeqd4ul3z0tYajYjUnZDVrP18SMxc3m3bBDqOKoL1
671L0xy5lgixALYA
dml1ytNDyZDXab5yxBmtxuBCLvlgqCxcgVMDzd6yXV2gWE58gtY29GFy1thcbQ
==
```

submit

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

