ECC y certificados digitales

CRIPTOGRAFÍA - QT 24-25

Alumnos: Huilin Ni, Victor Gesiarz
Fecha de entrega: Viernes 20 de diciembre, 2024

APARTADO 1

Capturad una conexión TLS 1.3 con `www.wikipedia.org` que use un certificado con una clave pública ECC (Elliptic Curve).

(a) Comprobad que el número de puntos (orden) de la curva usada en el certificado es primo.

Al inspeccionar la traza podemos ver que la curva utilizada es "secp256r1", con la librería ECPY podemos fácilmente extraer esto con el siguiente código:

```python
cv = Curve.get_curve('secp256r1')
curve_order = cv.order
print(curve_order)
print(isprime(curve_order))
```

Este código nos muestra que el orden es:
115792089210356248762697446949407573529996955224135760342422259061068512044369
y que este es, efectivamente, **un número primo.**

(b) Comprobad que la clave pública P de `www.wikipedia.org` es realmente un punto de la curva.

Para este apartado podemos usar el siguiente código, que nos da como resultado que el **punto es de la curva**.

```python
try:
    pub_key = ECPublicKey(punto)
    print("APARTADO 1.b: El punto es de la curva: TRUE")
except:
    print("APARTADO 1.b: El punto es de la curva: FALSE")
```

(c) Calculad el orden del punto P.

Dado que el orden de la curva es primo, una propiedad de los *Grupos* nos dice que cualquier punto de la curva es generador y que el orden de cualquier múltiplo de un generador será también primo y, de hecho, será el primo mismo. Lo podemos demostrar con la siguiente propiedad:

$$orden\,(r \cdot P) \;=\; \frac{orden\,(P)}{\gcd\,(r,\; orden\,(P))}$$

Sabemos que el orden del generador, que en nuestro caso P equivale a la Q, la clave pública, es primo. El máximo común divisor entre un primo y cualquier otro, en este caso es la r, es equivalente a 1. Teniendo eso en cuenta, tendríamos que:

$$orden\,(r \cdot P) = \frac{primo}{1} = primo$$

Por lo tanto, el orden de cualquier múltiplo de un punto (generador) de la curva, es igual al orden de la misma curva.

(d) Comprobad que la firma ECDSA es correcta.

Para hacer la comprobación de la firma utilizamos el siguiente código:

```python
cv = Curve.get_curve('secp256r1')
pub_key = ECPublicKey(Point(Qx, Qy, cv))
signer = ECDSA()
is_valid = signer.verify(
    m_bytes,
    signature,
    pub_key
)
```

Se puede ver como hemos extraído “m_bytes”, “signature” y el punto Q en el código, que se ha basado en los pasos explicados en las diapositivas.

Este código da como resultado que **la firma es válida**.

APARTADO 2

Conectaros con `www.fib.upc.edu`. En esta conexión os enviarán el certificado del servidor de la FIB.

(a) Obtened el periodo de validez del certificado y la clave pública (módulo y exponente, en base 10) de la web de la FIB. ¿Cuántos dígitos tiene el módulo?

El periodo de validez del certificado es:

- Not Before: **05/12/2024, 01:00:00 CET**
- Not After: **06/12/2025, 00:59:59 CET**

La clave pública es:

- $\text{Módulo}_{\text{base 10}}$:
  5216751630076633628016377796003040909893028742267921956616218713443141666849723694405344975720215391610437463462260975027528422793819426704394955181375253559490948747677843666318516252006531710445279302886106274991898971057155656932222257739950489017175625788923242032186082016621692697729256789257299209023966887272193867456089977195751946788951514658069128891757218062709040652492101770872862051065212805321415686747955964074296154183424184099345061361380835106772819091711055779223799306309608122528242322765146684044018141002258004685074358575766269332458801746115195650093971122378689573098175389568882935029791836069084971350536151808707623977192150417701705174790765865406397614283962435501787686908180147738738802573732732134108800391018615830271209964287064021108018953096224215212534727431066880107424040439885682937028168246483584371313651019664477173260078654151672684875922173500473841795229538430756850287739803

Con longitud de 3072 bits.

- $\text{Exponente}_{\text{base 10}}$: **17**
  con longitud 17 bits.

(b) En el certificado encontraréis un enlace a la política de certificados (CPS) de la autoridad certificadora firmante. ¿Qué tipo de claves públicas y tamaños admite?

Accediendo al enlace, en el apartado 6.1.5. (página 47/75) nos habla de los tipos de claves públicas y tamaños que admite la autoridad.

> **6.1.5. Key sizes**
>
> This CP requires use of **RSA PKCS #1**, **RSASSA-PSS**, **DSA**, or **ECDSA** signatures; additional restrictions on key sizes and hash algorithms are detailed below. Certificates issued under this policy SHOULD contain RSA or elliptic curve Public Keys.
>
> All Certificates that expire on or before December 31, 2030 SHOULD contain subject Public Keys of at least **2048 bits for RSA/DSA**, at least **256 bits for elliptic curve**, and be signed with the corresponding Private Key.

> All Certificates that expire after December 31, 2030 SHOULD contain subject Public Keys of at least **3072 bits for RSA/DSA**, at least **256 bits for elliptic curve**, and be signed with the corresponding Private Key.
>
> CAs that generate Certificates and CRLs under this policy SHOULD use the **SHA-256**, or **SHA-384** hash algorithm when generating digital signatures.
>
> ECDSA signatures on Certificates and CRLs SHOULD be generated using **SHA-256** or **SHA-384**, as appropriate for the key length.
>
> Where implemented, CSSs SHALL sign responses using the same signature algorithm, key size, and hash algorithm used by the CA to sign CRLs.

(c) En el certificado encontraréis un enlace a un punto de distribución de la CRL de la autoridad certificadora. ¿Cuántos certificados revocados contiene la CRL?

Ejecutando el siguiente comando de Openssl:

```
Unset
>>> openssl crl -in ./GEANTOVRSACA4.crl -inform DER -text -noout | grep "Serial Number" | wc -l
```

Y nos dice que hay **25841** certificados revocados.

(d) En el certificado encontraréis la dirección OCSP (Online Certificate Status Protocol) a la que se puede preguntar por el estatus del certificado. ¿Cuál es el estatus del certificado y hasta cuándo es válido dicho estatus?

Como el certificado de la autoridad certificadora se descarga en formato .crt y el de la fib se descarga en formato .pem, primero lo pasamos al mismo formato con el siguiente formato:

```
Unset
>>> openssl x509 -in ./GEANTOVRSACA4.crt -inform DER -out ./GEANTOVRSACA4.pem -outform PEM
```

Una vez tenemos todos los archivos en el formato que toca, con el siguiente comando:

```
Unset
>>> openssl ocsp -issuer ./GEANTOVRSACA4.pem -cert ./www-fib-upc-edu.pem -url http://GEANT.ocsp.sectigo.com
```

Obtenemos el siguiente output:

```
Unset
WARNING: no nonce in response
Response verify OK
./www-fib-upc-edu.pem: good
      This Update: Dec 17 16:31:34 2024 GMT
      Next Update: Dec 24 16:31:33 2024 GMT
```

Podemos ver que el certificado es válido y este estatus es válido hasta el "**Dec 24 16:31:33 2024 GMT**".