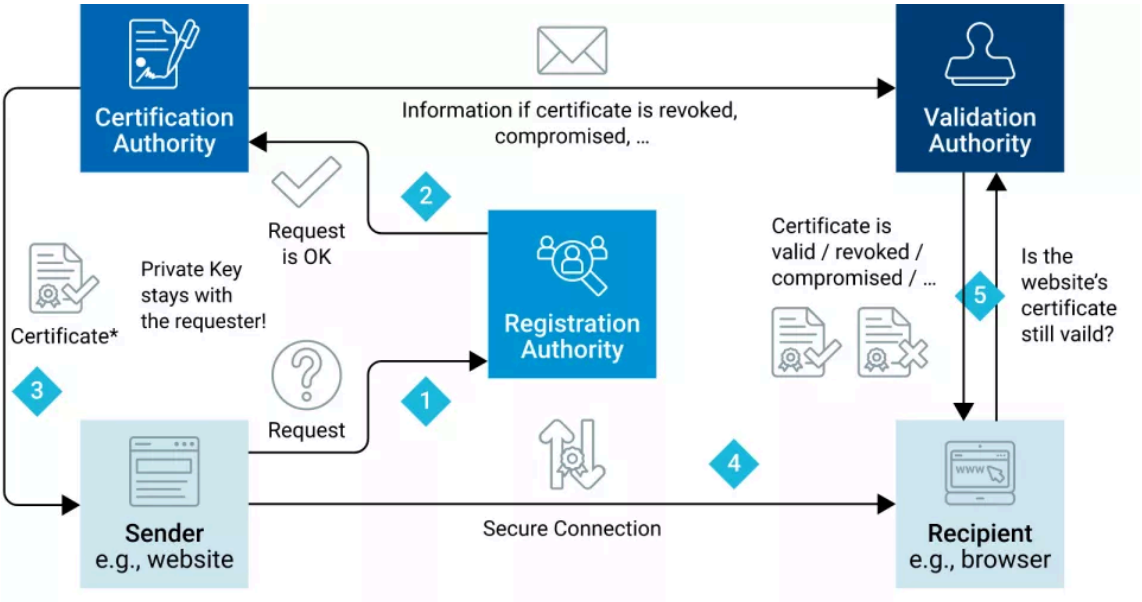
SAD - U2.5. Infraestructura de Clave Pública

Descargar estos apuntes

Índice

1. Infraestructura de Clave Pública (PKI)



La PKI (Public Key Infraestructure) hace referencia a herramientas utilizadas para crear y gestionar la seguridad basada en cifrado asimétrico o de clave pública.

Este tipo de cifrado se utiliza para asegurar las comunicaciones en Internet, y lo usamos todo el tiempo en nuestros navegadores, en los servidores web, en la infraestructura de correo electrónico, etc, por lo que entender su funcionamiento es parte fundamental para garantizar la seguridad en nuestras infraestructuras.

> **El cifrado asimétrico en el mundo real**
>
> Ya sabemos cómo funciona el cifrado y descifrado utilizando criptografía asimétrica en OpenSSL.
>
> Ahora bien, ¿cómo se usa esto en el mundo real?
>
> Básicamente, las entidades que necesitan garantizar autenticidad, como los servidores web por ejemplo, tienen su par de claves, pública y privada, y encapsulan su clave pública dentro de un certificado digital x509.
>
> Este certificado añade metadatos como la validez del mismo, firmas digitales, **autoridad certificante**, etc.
>
> Y aquí surge un término importante en la PKI: la **CA** o Autoridad Certificante (Certificate Authority), y una nueva variable en las ecuaciones criptográficas: la **confianza**.

1.1. ¿Cómo funciona una PKI?

La PKI se basa en el uso de certificados digitales, que son documentos digitales que contienen la clave pública de un usuario o entidad, y que han sido firmados digitalmente por una entidad de confianza, llamada Autoridad Certificante (CA).

Pongamos un ejemplo, como los vistos en el caso de las claves privadas y públicas vistas en el apartado anterior:

- **Alice** quiere enviar un mensaje a **Bob** de forma segura.
- **Bob** genera un par de claves, una privada y otra pública.
- **Bob** envía su clave pública a **Alice**. --> `Equivale a un certificado digital`.
- **Alice** cifra el mensaje con la clave pública de **Bob**.
- **Alice** envía el mensaje cifrado a **Bob**.
- **Bob** descifra el mensaje con su clave privada.

En este caso, **Bob** ha generado su par de claves, y ha enviado su clave pública a **Alice**. **Alice** ha utilizado esta clave para cifrar el mensaje, y **Bob** ha utilizado su clave privada para descifrarlo.

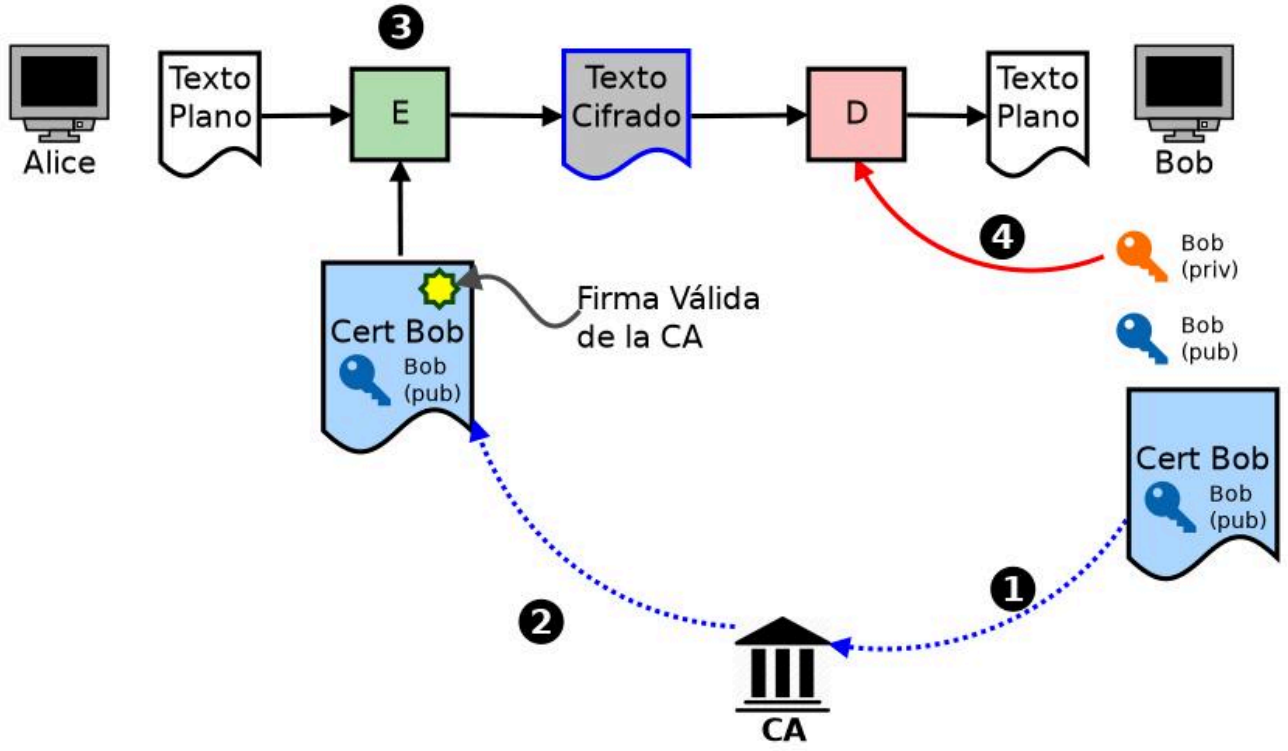
> **Ataque MITM**
>
> Ahora, imaginemos que un atacante, intercepta el mensaje en el que Bob le envía su clave pública a Alice.
> Nada le impide al atacante crear su propio par de claves.
> Con ese par de claves puede suplantar la identidad de Bob. Le envía su clave publica a Alice.
> A partir de ese momento puede establecer una comunicación con Alice haciéndose pasar por Bob.
> Como tiene la clave pública de Bob, el atacante puede cifrar los mensajes con esa clave y hacérselos llegar a Bob.De esta forma puede descifrar tanto los mensajes de Alice a Bob como los de Bob a Alice.

Aquí es donde la PKI aporta su granito de arena para lograr la comunicación segura, y dificulta a los atacantes realizar este tipo de ataque.

> El problema con el caso anterior es que, si bien Alice puede verificar la firma recibida utilizando la clave incluida en el certificado digital recibido, no es capaz de verificar la autenticidad de dicho certificado.

2. La solución: Las CA's

La **CA es una tercera entidad dentro de la PKI**, que también dispone de su clave privada y su clave pública encapsulada en un certificado digital x.509.



Una **CA** es una entidad en la que ambos, Alice y Bob, **confían**. Tanto Alice como Bob tienen el certificado de la CA, firmado por la propia CA para verificar su autenticidad.

> La CA es la entidad encargada de emitir y revocar certificados digitales (pares de claves privadas y públicas).

Verificación de claves

Supongamos que Alice quiere enviar a Bob un mensaje encriptado usando la clave pública de Bob.
Ahora Bob tiene un certificado firmado por una CA.
Alice, cuando recibe un certificado puede verificar que dicho certificado es efectivamente de Bob, y no de un intermediario como intentando suplantar su identidad.

2.1. Otras entidades en la PKI

Existen otras entidades involucradas en la PKI, pero no se las suele nombrar demasiado porque muchas veces su funcionalidad está incorporada en la propia CA, o en las implementaciones de software de los emisores y receptores dentro de una comunicación segura.

Entre ellas se encuentra la **Autoridad de Registro**, o `RA (Registration Authority)`, que verifica que las claves públicas incluidas en los certificados a firmar por la CA pertenezcan efectivamente a la entidad que solicita la firma. Por ejemplo, si

Alice quiere firmar su certificado con una CA confiable, la RA debe verificar que la clave pública incluida en el certificado de Alice pertenezca efectivamente a Alice.

Por otro lado, existe la **Autoridad de Validación**, o `VA (Validation Authority)`, encargada de comprobar la validez de los certificados digitales, tanto la validez de su firma digital, como las fechas de expiración.

Finalmente, la **Autoridad de Sellado de Tiempo**, o `TSA (Time Stamp Authority)`, encargada de firmar documentos con el fin de constatar que dichos documentos existían en un determinado instante de tiempo.

3. Seguridad en la PKI

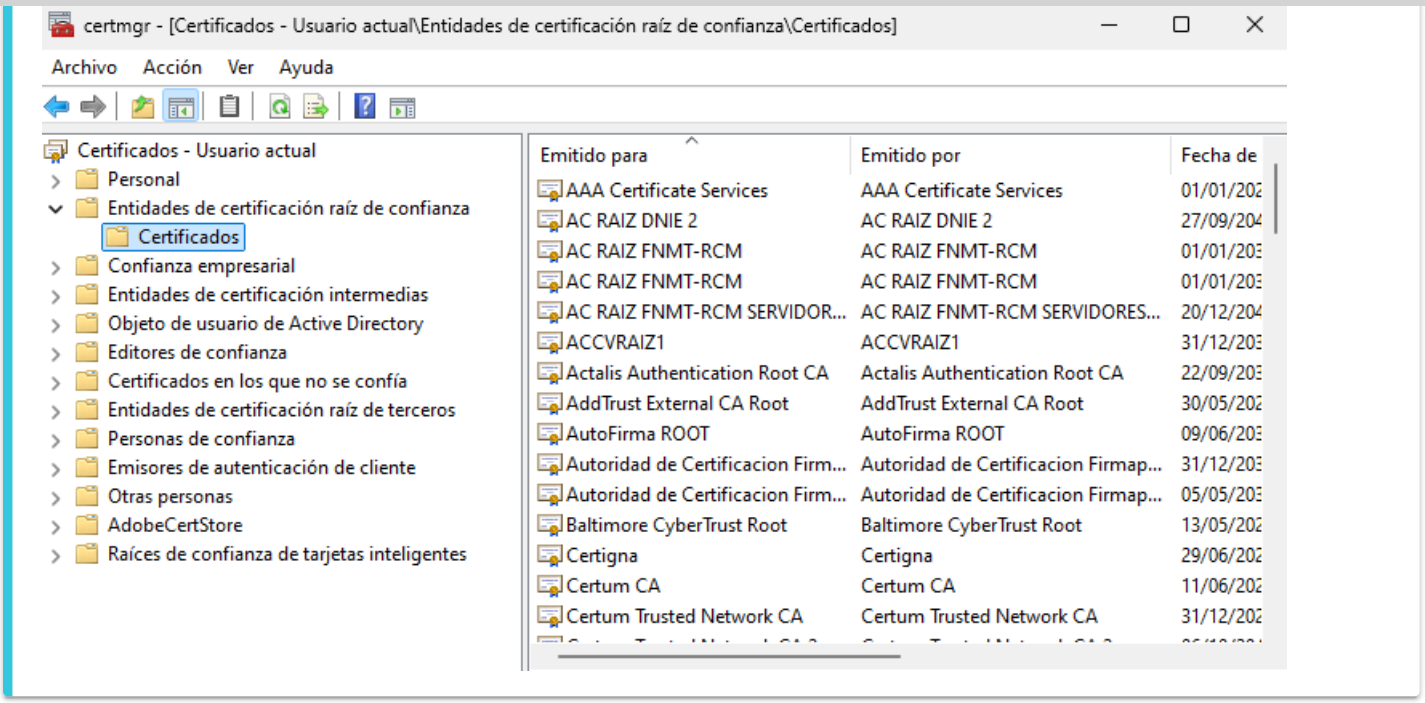
La seguridad en la PKI se basa en la confianza que se tiene en las CA's. Si una CA es comprometida, toda la PKI se ve comprometida.

En primer lugar, y como aspecto más importante, al igual que en criptografía asimétrica, la seguridad en la infraestructura PKI depende principalmente de la **protección que demos a las claves privadas**.

En segundo lugar, cabe mencionar los aspectos de seguridad de una de las herramientas que utilizan PKI y que usamos todo el tiempo: nuestros navegadores web (como parte del Sistema Operativo de un dispositivo).

1. Cuando ingresamos a un sitio seguro, vemos en la URL que el protocolo es HTTPS (HTTP over SSL/TLS). Eso significa que el navegador pudo establecer un canal seguro contra el servidor del sitio.
2. Para lograr esto, el servidor del sitio le envió al navegador su **certificado digital firmado por una CA reconocida en la que el navegador confía**. Esto significa que el navegador tiene una lista de CAs en las que confía almacenadas en sus configuraciones, y asociadas a sus certificados digitales.
   - Cuando un sitio le envía al navegador un certificado digital firmado por una CA, **el navegador utiliza la clave pública de esa CA para verificar la firma**.
     - Si el navegador posee dicha clave pública (certificado digital de la CA), podrá verificar la veracidad del sitio web.
     - En caso contrario, el navegador nos advertirá que el certificado no pudo verificarse, y que el sitio no es confiable.
       Seguramente también nos permita añadir el certificado a los sitios confiables, cosa que no debemos realizar nunca, salvo que sepamos lo que hacemos.

> **Gestor de certificados en Windows**
>
> En Windows, podemos acceder al gestor de certificados digitales desde el panel de control, en la sección de seguridad.
>
> La forma más rápida de acceder a esta configuración es ejecutando la consola de administracion de certificados, `certmgr.msc`.
>
> También se puede acceder a esta configuración desde los navegadores. Por ejemplo, en **Chrome** la tenemos en `Configuración > Privacidad y Seguridad > Seguridad > Gestionar certificados`



Un **certificado digital x.509** es un documento digital que ha sido codificado y/o firmado digitalmente de acuerdo a la RFC 5280.

> **x.509**
>
> De hecho, el término «Certificado x.509» usualmente se refiere al certificado PKIX de la IETF, y al perfil CRL del estándar de certificados digitales x.509 v3 especificado en la RFC 5280, donde PKIX hace referencia a la Infraestructura de clave pública x.509 (Public Key Infrastructure X.509).

El estándar X.509 solo define la sintaxis de los certificados, por lo que no está atado a ningún algoritmo en particular, y contempla los siguientes campos en su estructura:

- Version.
  Especifica cuál de las tres versiones de x.509 se aplica a este certificado.
- Numero de serie.
  Es un identificador único para todos los certificados de una misma CA.
- Identificador del algoritmo empleado para la firma digital.
  Qué algoritmo utilizó la CA del certificado para firmarlo.
- Nombre del certificador.
  Nombre de la CA.
- Periodo de validez.
  Por seguridad, los certificados tiene un tiempo de vida.
- Nombre del sujeto.
  Nombre del dueño del certificado.
- Clave publica del sujeto.
  Clave pública del dueño del certificado, y algoritmo asociado.
- Extensiones.
  Información adicional permitida por el estándar.

- Firma digital de todo lo anterior generada por el certificador.
  Firma digital de la CA.

Un ejemplo de certificado digital x.509 sería el siguiente:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            0a:01:41:42:00:00:01:53:85:73:6a:0b:85:ec:a7:08
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: O = Digital Signature Trust Co., CN = DST Root CA X3
        Validity
            Not Before: Mar 17 16:40:46 2016 GMT
            Not After : Mar 17 16:40:46 2021 GMT
        Subject: C = US, O = Let's Encrypt, CN = Let's Encrypt Authority X3
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:9c:d3:0c:f0:5a:e5:2e:47:b7:72:5d:37:83:b3:
                    [...]
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE, pathlen:0
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
            Authority Information Access:
                OCSP - URI:http://isrg.trustid.ocsp.identrust.com
                CA Issuers - URI:http://apps.identrust.com/roots/dstrootcax3.p7c
            X509v3 Authority Key Identifier:
                C4:A7:B1:A4:7B:2C:71:FA:DB:E1:4B:90:75:FF:C4:15:60:85:89:10
            X509v3 Certificate Policies:
                Policy: 2.23.140.1.2.1
                Policy: 1.3.6.1.4.1.44947.1.1.1
                  CPS: http://cps.root-x1.letsencrypt.org
            X509v3 CRL Distribution Points:
                Full Name:
                  URI:http://crl.identrust.com/DSTROOTCAX3CRL.crl
            X509v3 Subject Key Identifier:
                A8:4A:6A:63:04:7D:DD:BA:E6:D1:39:B7:A6:45:65:EF:F3:A8:EC:A1
    Signature Algorithm: sha256WithRSAEncryption
    Signature Value:
        dd:33:d7:11:f3:63:58:38:dd:18:15:fb:09:55:be:76:56:b9:
        [...]
```

1.1. Formato de un certificado digital

Un punto importante a entender, y que puede generar confusiones entre los usuarios de certificados digitales X.509, son las extensiones (o formatos, mejor) que éstos pueden tener.

Existe mucha confusión al respecto de las extensiones **DER, PEM, CRT y CER**, y generalmente, y de manera incorrecta, muchos piensan que pueden utilizarse e intercambiarse sin generar conflictos.

Mientras que en ciertos casos algunas de estas extensiones pueden intercambiarse, la mejor práctica es la de identificar cómo está codificado internamente el certificado, y entonces etiquetarlo correctamente con la extensión que corresponda, ya que esto evitará problemas luego a la hora de manipular los certificados digitales.

Codificación de los certificados digitales

Las siguientes codificaciones de formato también suelen utilizarse como extensiones en los archivos de certificados digitales. A saber:

- **DER**
  La extensión DER es utilizada para `certificados digitales codificados en forma binaria`. Estos archivos también suelen tener la extensión CRT o CER. DER es un tipo de codificación de certificados X.509, NO un tipo de certificado, de modo que la expresión «Tengo un certificado DER» es incorrecta, y debería optarse por «Tengo un certificado X.509 codificado en formato DER».
- **PEM**
  Estos `certificados están codificados en Base64 / ASCII`, y generalmente se utilizan para certificados digitales X.509 v3. Son aquellos que al abrirlos con un editor de texto comienzan y terminan con líneas similares a las siguientes:

  ```
  -----BEGIN CERTIFICATE-----
  MIIDdzCCAl+gAwIBAgIJAJ7z7U5z6g8IMA0GCSqGSIb3DQEBCwUAMIGUMQswCQYD
  VQQGEwJVUzETMBEGA1UECBMKU29tZS1TdGF0ZTEhMB8GA1UEChMYSW50ZXJuZXQg
  [...]
  -----END CERTIFICATE-----
  ```

  Los archivos PEM pueden contener certificados, claves privadas, y otros datos. En el caso de los certificados digitales, la extensión PEM es la más común.

Extensiones de certificados digitales X.509

- **CRT**
  Esta extensión es utilizada para certificados propiamente dichos. Los certificados pueden ser codificados en binarios DER o como ASCII PEM. Las extensiones CER y CRT podrían considerarse sinónimos.
  Es la extensión más utilizada en sistemas *nix, como Linux o Unix.
- **CER**
  Es una extensión alternativa a la CRT pero siguiendo las convenciones de Microsoft. La extensión CER es también reconocida por Internet Explorer como un comando para correr un ejecutable de la API de criptografía de Microsoft, y es utilizada principalmente por rundll32.exe, cryptext.dll, y CryptExtOpenCER, que muestran un mensaje de diálogo dando la posibilidad de importar el certificado en el sistema, o analizar su contenido.
  La única ocasión en la que podemos intercambiar libremente las extensiones CRT y CER sin tener problemas, es cuando el certificado tiene la misma codificación… es decir, por ejemplo, cuando el certificado tiene extensión CRT o CER, pero su codificación es PEM.
- **KEY**
  Esta extensión es utilizada tanto para la clave pública como la privada de los estándares PKCS. Estas claves pueden estar codificadas en formato binario DER o ASCII PEM.

####Uso de los certificados digitales

Los certificados digitales suelen emplearse para:

- Sirven para identificarse ante terceros.
- Permiten tomar precauciones contra la suplantación de identidad.

- Los certificados digitales pueden garantizar:
  la autentificación de las partes.
  la integridad de la transacción.
  la confidencialidad de la información.
  el no repudio

2. Certificados digitales X.509 con OpenSSL

Vamos a analizar cuatro formas básicas de manipular certificados digitales X.509 utilizando openssl. Estas son, ver el certificado, convertirlo, combinarlo, y extraer claves desde el mismo.

2.1. Ver un certificado

Los certificados en general no tienen un formato legible al humano, incluso los formatos Base64 PEM, por lo que haremos uso de las herramientas provistas por OpenSSL para analizar su contenido.

Para ver el contenido de un certificado en formato PEM (independientemente de su extensión) podemos utilizar el siguiente comando:

```
$ openssl x509 -in /etc/ssl/cert.pem -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 6828503384748696800 (0x5ec3b7a6437fa4e0)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: CN = ACCVRAIZ1, OU = PKIACCV, O = ACCV, C = ES
        Validity
            Not Before: May  5 09:37:37 2011 GMT
            Not After : Dec 31 09:37:37 2030 GMT
        Subject: CN = ACCVRAIZ1, OU = PKIACCV, O = ACCV, C = ES
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:9b:a9:ab:bf:61:4a:97:af:2f:97:66:9a:74:5f:
                    [...]
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            Authority Information Access:
                CA Issuers - URI:http://www.accv.es/fileadmin/Archivos/certificados/raizaccv1.crt
                OCSP - URI:http://ocsp.accv.es
            X509v3 Subject Key Identifier:
                D2:87:B4:E3:DF:37:27:93:55:F6:56:EA:81:E5:36:CC:8C:1E:3F:BD
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Authority Key Identifier:
                D2:87:B4:E3:DF:37:27:93:55:F6:56:EA:81:E5:36:CC:8C:1E:3F:BD
            X509v3 Certificate Policies:
                Policy: X509v3 Any Policy
                  User Notice:
                    Explicit Text:
                  CPS: http://www.accv.es/legislacion_c.htm
            X509v3 CRL Distribution Points:
                Full Name:
                  URI:http://www.accv.es/fileadmin/Archivos/certificados/raizaccv1_der.crl
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            X509v3 Subject Alternative Name:
                email:accv@accv.es
    Signature Algorithm: sha1WithRSAEncryption
    Signature Value:
        97:31:02:9f:e7:fd:43:67:48:44:14:e4:29:87:ed:4c:28:66:
        [...]
```

Warning

En el caso de obtener un error similar a este:

```
unable to load certificate
12626:error:0906D06C:PEM routines:PEM_read_bio:no start line:pem_lib.c:647:Expecting: TRUSTED CERTIFICATE
```

> Es posible que el archivo no sea un certificado en formato PEM, sino en formato DER. En este caso, se debe convertir el certificado a formato PEM antes de poder visualizarlo. También podemos usar un comando similar a este para visualizar certificados en formato DER:
>
> ```
> $ openssl x509 -in certificado.der -inform der -text -noout
> ```

2.2. Convertir entre formatos de certificados digitales

A veces, y dependiendo de las aplicaciones en las que vayamos a utilizar los certificados, puede que sea necesario cambiar el formato de un certificado.

Esto no es tarea difícil para la suite OpenSSL, y podemos llevarla a cabo de las siguientes maneras:

Convirtiendo PEM a DER:

```
openssl x509 -in cert.crt -outform der -out cert.der
```

Convirtiendo DER a PEM:

```
openssl x509 -in cert.crt -inform der -outform pem -out cert.pem
```

2.3. Extraer claves desde un certificado

Por último, y sabiendo que un certificado digital X.509 se compone de metainformación agregada a una clave pública, utilizando la suite OpenSSL también podemos extraer la clave desde el certificado.

Supongamos que tenemos un certificado en formato PEM (si es DER, podemos convertirlo como se vio mas arriba). Podemos extraer la calve pública con un comando similar a este:

```
openssl x509 -inform pem -in cert.crt -pubkey -noout > pub.key
```

Donde pub.key es la clave pública resultante, también en formato PEM.

3. Creación de certificados digitales X.509 con OpenSSL

Para crear un certificado digital X.509 con OpenSSL, necesitamos una clave privada y un archivo de configuración que defina los campos del certificado.