PRÁCTICA 3 - SMIME

Supongamos que tenemos dos emisores, que pretenden enviar un archivo en formato pdf, doblemente firmado, y cifrado, a un receptor. Dicho receptor deberá descifrar el pdf, verificar la firma, y finalmente, abrirlo con un visor de ficheros pdfs.

Pasos a seguir por el **Emisor 1**:

```
openssl smime -sign -binary -nodetach -in <pdf> -out <pdf>.sgn -signer
<cert_primer_emisor> -inkey <key_primer_emisor>
```

> **Explicación:** el emisor 1, invocando a smime, con -sign, firma el archivo PDF utilizando su certificado y su clave privada. Con la opción -binary, indica que el archivo no debe convertirse al formato canónico (útil para manejar datos binarios como PDFs). Con -nodetach, nera una firma opaca (la firma y el contenido original se integran en el archivo de salida). Con -in se indica el pdf a firmar, con -out, el pdf firmado y con -signer y -inkey, se especifican el certificado del emisor y su clave, necesarios para generar la firma.

```
openssl smime -encrypt -in <pdf>.sgn -out <pdf>.sgn.enc <cert_segundo_emisor>
```

> **Explicación:** el emisor 1, con -encrypt, cifra el archivo firmado (pdf.sgn) usando el certificado público del segundo emisor. Con -in, indica el archivo de entrada (pdf firmado previamente), con -out, se indica el fichero de salida (archivo firmado y cifrado), y se indica el certificado público del segundo emisor, utilizado para el cifrado.

Pasos a seguir por el **Emisor 2**:

```
openssl smime -decrypt -in <pdf>.sgn.enc -out <pdf>.sgn -recip
<cert_segundo_emisor> -inkey <key_segundo_emisor>
```

> **Explicación:** El emisor 2, invocando a smime con -decrypt, descifra el archivo firmado y cifrado (.sgn.enc) utilizando su certificado público y su clave privada.Con -in, indica el archivo de entrada cifrado (.sgn.enc). Con -out, especifica el archivo descifrado de salida (.sgn). Con -recip y -inkey, proporciona el certificado público del emisor 2 y su clave privada, necesarias para realizar la operación de descifrado. El archivo descifrado (pdf.sgn) contiene el PDF firmado por el primer emisor.

```
openssl smime -resign -binary -nodetach -in <pdf>.sgn -out <pdf>.sgn2 -signer
<cert_segundo_emisor> -inkey <key_segundo_emisor>
```

> **Explicación:** El emisor 2, utilizando smime con -resign, añade su propia firma al archivo firmado por el primer emisor (.sgn). Con -binary, mantiene el formato binario del archivo, necesario para manejar el contenido del PDF. Con -nodetach, genera una firma opaca (la firma del emisor 2 y el contenido original se integran en un único archivo). Con -in, especifica el archivo de entrada ya firmado (.sgn).Con -out, define el archivo de salida con ambas firmas (.sgn2). Con -signer y -inkey, proporciona el certificado público y la clave privada del emisor 2, necesarios para generar la nueva firma. El archivo (pdf.sgn2) contiene el pdf firmado por ambos emisores.

```
openssl smime -encrypt -in <pdf>.sgn2 -out <pdf>.sgn2.enc <cert_receptor>
```

> **Explicación:** El emisor 2, invocando a smime con -encrypt, cifra el archivo con doble firma (.sgn2) para el receptor, usando su certificado público. Con -in, especifica el archivo de entrada firmado por ambos emisores (.sgn2). Con -out, define el archivo cifrado de salida (.sgn2.enc). Se utiliza <cert_receptor>, el certificado público del receptor, para cifrar el archivo, garantizando que solo este pueda descifrarlo. El archivo cifrado (pdf.sgn2.enc) es enviado al receptos para su descifrado.

Pasos a seguir por el **Receptor**:

```
openssl smime -decrypt -in <pdf>.sgn2.enc -out <pdf>.sgn2 -recip <cert_receptor>
-inkey <key_receptor>
```

> **Explicación:** El receptor, utilizando smime con -decrypt, descifra el archivo cifrado (.sgn2.enc) utilizando su certificado público y su clave privada. Con -in, especifica el archivo cifrado de entrada (.sgn2.enc). Con -out, define el archivo descifrado de salida (.sgn2). Con -recip y -inkey, proporciona el certificado público y la clave privada del receptor, necesarios para descifrar. El archivo descifrado (pdf.sgn2) contiene el PDF con las firmas de ambos emisores.

```
openssl smime -pk7out -in <pdf>.sgn2 | openssl pkcs7 -print_certs -noout
```

> **Explicación:** El receptor utiliza smime con -pk7out para extraer la estructura PKCS#7 del archivo firmado (.sgn2) y posteriormente usa openssl pkcs7 para analizarla y mostrar los certificados incluidos en el archivo firmado.Con -in, especifica el archivo firmado (.sgn2). La tubería (|) conecta el resultado de la extracción al comando pkcs7 -print_certs -noout, que imprime los certificados de los firmantes en formato legible. El receptor obtiene información sobre los emisores que firmaron el archivo.

```
openssl smime -verify -binary -in <pdf>.sgn2 -noverify -out <pdf>
```

> **Explicación:** El receptor, utilizando smime con -verify, verifica las firmas del archivo firmado (.sgn2) y extrae el contenido original. Con -binary, asegura que los datos binarios del PDF no sean alterados durante la operación. Con -in, especifica el archivo con las firmas de los emisores (.sgn2). Con -noverify, omite la validación de la cadena de certificados (útil si no se dispone de los certificados intermedios). Con -out, define el archivo PDF extraído. El receptor obtiene el archivo pdf original, listo para ser visualizado con un visor de pdf.

A TENER EN CUENTA

Opción **-binary**

Por defecto, Openssl convierte los datos de entrada a un formato llamado "canónico". Este formato utiliza caracteres de control (CR; carriage return) y LF (line feed) como terminadores de línea, en cumplimiento con la especificación S/MIME. Entonces, cuando se utiliza la opción **-binary**, openssl omite la conversión de los datos al formato canónico. Esto significa que maneja los datos tal y como están, sin alterar sus terminadores de línea ni ninguna otra parte del formato. Esto es muy útil cuando se trabaja con datos que no siguen el formato MIME, como archivos binarios (imágenes, vídeos, PDFs), ya que una conversión automática podría corromperlos.

Si no usáramos -binary en la sentenciia, Openssl podría convertir un archivo que usa solo LF como terminador de línea (estándar de Unix) a CRLF (estándar de Windows).

En definitiva, se garantiza que los datos no sufran ninguna transformación, preservando su formato original. Si tenemos cualquien archivo binario (un video, o cualquier dato que no sea texto puro), usar **-binary** es obligatorio para asegurarse de que los datos no se alteren.

Opción **-nodetach**

Se utiliza para generar una **firma opaca** (o "in-line") cuando se firma un archivo o mensaje. Esto significa que la firma se inserta directamente en el contenido del mensaje o archivo en lugar de adjuntarse como un archivo separado.

Cuando se usar **-nodetach**, la firma digital no se coloca en una parte separada del mensaje o archivo, sino que se incluye dentro del mismo archivo. Esto crea una firma que está "pegada" al contenido y no se puede separar fácilmente del archivo firmado. Este formato es más resistente a ciertos ataques, como los intentos de retransmisión de correo, donde un mensaje o firma previamente enviada se vuelve a enviar sin cambios. En una firma opaca, dado que la firma está ligada al contenido específico, no se puede reusar sin invalidar la firma.

En caso de no usarse, se genera una firma separada o detached. En este caso, la firma se crea como un archivo independiente que se adjunta al archivo original como una firma digital.

Opción **-recip**

Tiene un papel crucial en el proceso de descifrado de un mensaje o archivo cifrado con SMIME. especifica a OpenSSL cuál es el certificado del destinatario cuyo certificado público se utilizó para cifrar el archivo. Si no se especifica el certificado del destinatario del archivo cifrado, se produce un error. El certificado público se utiliza para cifrar el contenido antes de enviarlo. Este certificado es accesible a cualquier persona que quiera cifrar datos para ese destinatario, pero solo el destinatario podrá descifrar el archivo con su correspondiente clave privada

Opción **-resign**

Se utiliza para volver a firmar un archivo que ya ha sido firmado. See añade una nueva firma, generada con el certificado y la clave privada proporcionadas y, por tanto, el archivo resultante contiene todas las firmas anteriores, más la nueva.

Opción **-noverify**

Tiene como función desactivar la verificación de los certificados de firma cuando se procesa un mensaje o archivo firmado. Esto significa que al usar -noverify, OpenSSL no realiza ninguna comprobación sobre la validez de los certificados que se utilizaron para firmar el mensaje o archivo, lo que incluye verificar si los certificados están revocados o si están expirados.

Orden openssl smime -pk7out -in .sgn2 | openssl pkcs7 -print_certs -noout

La primera sentencia extrae la estructura PKCS7 del archivo de entrada y luego, extrae y muestra los certificados contenidos en el archivo pkcs7 (con la opción -print_certs), evitando que openssl imprima de

forma completa la estructura del archivo pkcs7, con -noout, que también podría incluir datos como la firma o el mensaje original.

GLOSARIO DE TÉRMINOS USADOS

- **LF (Line feed)**: es un carácter especial utilizado para indicar un salto de línea. Se usa en sistemas Linux. Cuando pulsamos Enter en un editor de texto en Linux, el cursor baja a la siguiente linea usando un LF (código ASCII 10).

- **CRLF**: terminador de línea en sistemas Windows.

- **Estándar MIME (Multipurpose Internet Mail Extension)**: conjunto de reglas que extiende el protocolo de correo electrónico (SMTP) para enviar diferentes tipos de contenido, como texto, imágenes, vídeos y archivos binarios.

- **Estructura PKCS7**: formato estándar para almacenar firmas digitales, claves públicas, certificados y otros datos de seguridad. El archivo pkcs7, en este caso, contiene los dos certificados de los que han firmado, a parte de otra información como la firma o el mensaje original.