Universidad Tecnológica de Panamá

Facultad de Ingenieria en Sistemas

Gestión y Desarrollo de Software

Estudiante:

Joseph Guerrero

Docente:

Irina Fong

Curso:

Desarrollo de Software 7

Tema del Laboratorio:

# Firmas Digitales con OpenSSL en PHP

Fecha:

19 de Mayo de 2025

## Introducción al Tema

La criptografía moderna permite asegurar la integridad y autenticidad de los datos mediante el uso de firmas digitales.
En este laboratorio se exploró el uso de OpenSSL desde PHP para generar claves públicas y privadas, crear certificados X.509 y firmar mensajes.
Se demuestra cómo un mensaje puede ser firmado con una clave privada y verificado posteriormente usando la clave pública o certificado,
lo cual garantiza que los datos no han sido alterados.

## Objetivo del Laboratorio

- Aplicar herramientas criptográficas en PHP utilizando la librería OpenSSL.
- Generar claves RSA y certificados X.509.
- Firmar mensajes con clave privada.
- Verificar la validez de firmas con claves públicas o certificados.

1. **Nombre de la función**: Firma7

1.1 **Propósito general**: Realizar un proceso completo de firma digital de un mensaje utilizando criptografía de clave pública (RSA) y verificar la validez de dicha firma con la clave pública correspondiente, garantizando la autenticidad e integridad del mensaje.

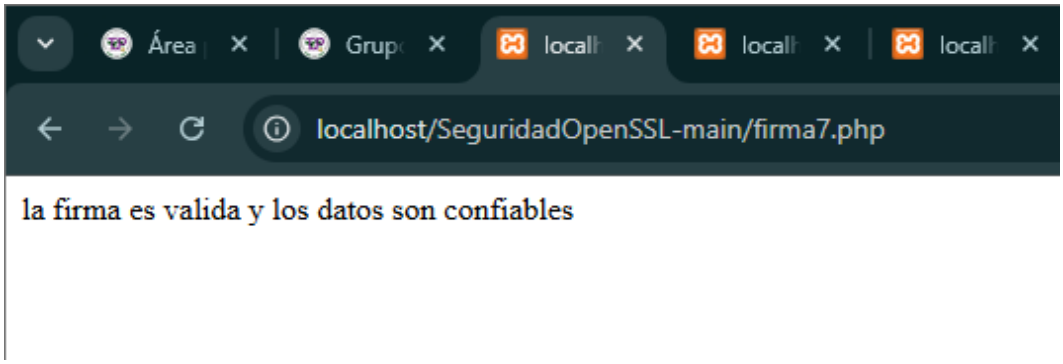
1.2 **Parámetros recibidos**: No recibe parámetros directamente, pero internamente trabaja con:

datos (string): El texto que se desea firmar.

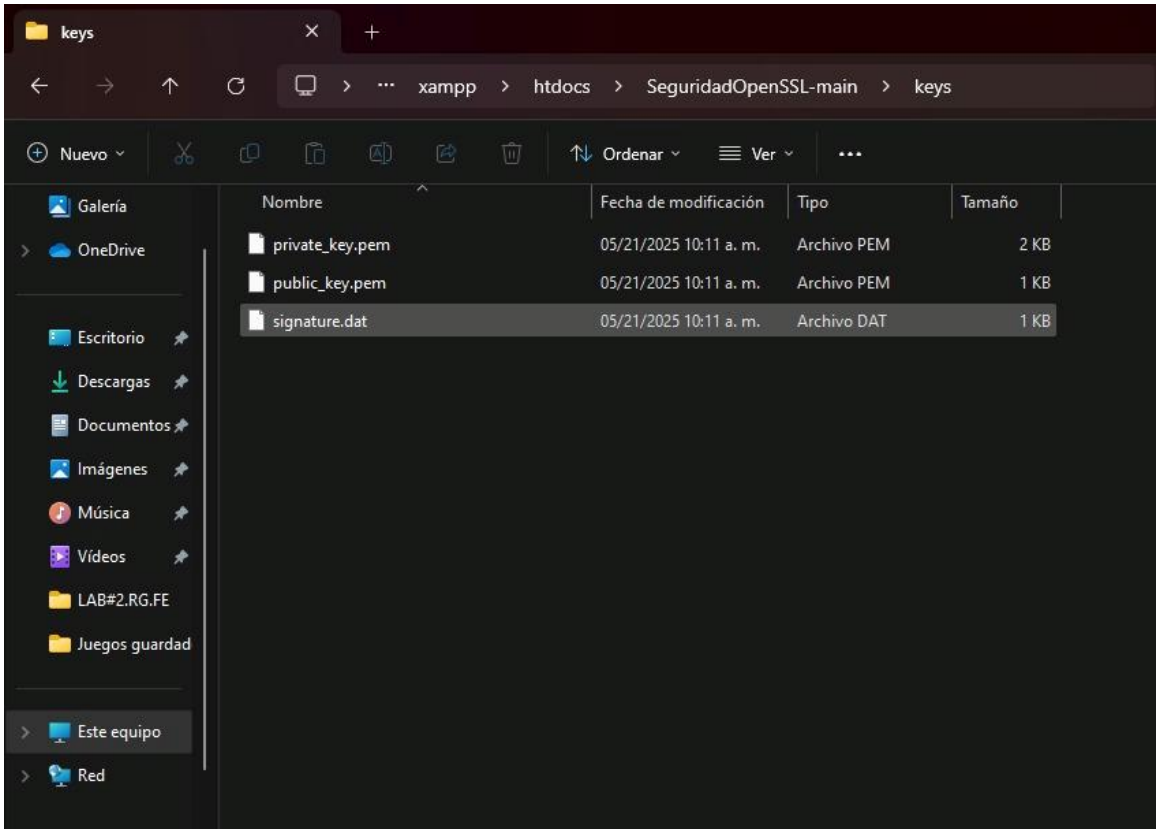
configArgs (array): Configuración necesaria para la generación del par de claves (incluye ruta del archivo de configuración de OpenSSL, tamaño de la clave, y tipo de clave).

Rutas de archivos para guardar y leer claves y firma (keys/private_key.pem, keys/public_key.pem, keys/signature.dat).

1.3 Valor devuelto:




Genera y guarda tres archivos en disco:



1.4 Explicación breve del proceso criptográfico aplicado:

El programa aplica criptografía de clave pública mediante el algoritmo RSA y el uso del hash SHA-256 para realizar una firma digital.

El proceso se divide en los siguientes pasos:

Generación de claves RSA: Se crea un par de claves (privada y pública) utilizando openssl_pkey_new.

Exportación y almacenamiento: Se extraen las claves en formato PEM y se guardan en archivos para su posterior uso.

Firma del mensaje: El mensaje original es firmado usando la clave privada. Se genera un hash SHA-256 del mensaje, el cual se cifra con la clave privada para crear la firma digital.

Verificación de la firma: Utilizando la clave pública, se verifica si la firma corresponde al mensaje original. Si el hash obtenido al verificar la firma coincide con el hash del mensaje, entonces se confirma que el mensaje no ha sido alterado y que fue firmado por el propietario de la clave privada.

2. **Nombre de la función:** FirmaOtra

**2.1 Propósito general:**

Crear un certificado digital X.509 autofirmado a partir de un par de claves RSA, para su uso en tareas como cifrado, firma digital o establecimiento de conexiones seguras (como HTTPS).

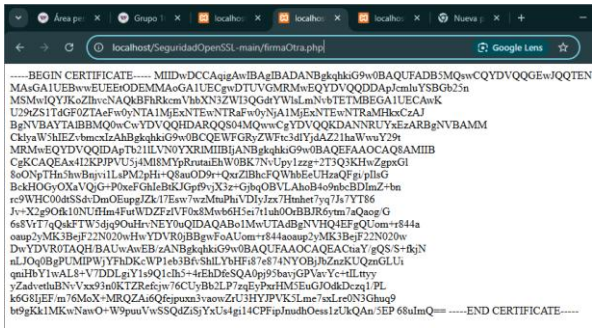
**2.2 Parámetros recibidos:**

No se reciben parámetros explícitamente al ejecutar el script, pero internamente se utilizan:

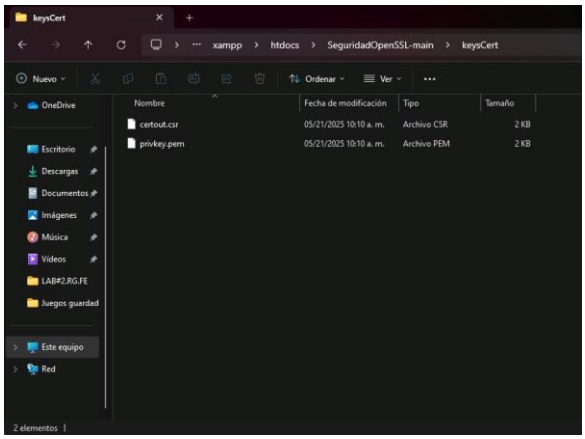
- $configArgs (array): Configuración de OpenSSL para generación de claves (ruta al archivo .cnf, tamaño de la clave, tipo de clave).
- $dn (array): Datos del sujeto del certificado, como país, localidad, organización, nombre común y correo electrónico.

2.3 **Valor devuelto:**

- Se imprime en pantalla el certificado X.509 generado (en formato PEM).




- Se guardan dos archivos en disco:




2.4 **Explicación breve del proceso criptográfico aplicado:**
El programa aplica un proceso criptográfico basado en **infraestructura de clave pública (PKI)** mediante **RSA** y OpenSSL. El flujo general es:

- **Generación del par de claves (RSA):**
Se crea una clave privada y su correspondiente clave pública mediante openssl_pkey_new.
- **Creación de la CSR (Solicitud de Firma de Certificado):**
Se genera una solicitud de firma de certificado con los datos de identidad del sujeto (como nombre, organización, país) utilizando openssl_csr_new.
- **Firma del certificado:**
Se autofirma la CSR usando la misma clave privada, generando un certificado X.509 válido por un año mediante openssl_csr_sign.
- **Exportación y almacenamiento:**
El certificado se convierte a formato PEM y se guarda junto con la clave privada en archivos, permitiendo su posterior uso en cifrado, firma digital o conexiones seguras.

## 3. Documentación del proceso criptográfico del programa

**3.1 Nombre de la función:** FirmarMesaje

**3.2 Propósito general:**

Firmar digitalmente un mensaje utilizando una clave privada previamente generada y luego verificar la validez de esa firma usando un certificado X.509. El objetivo es garantizar que el mensaje no ha sido modificado (integridad) y que proviene de una fuente confiable (autenticidad).

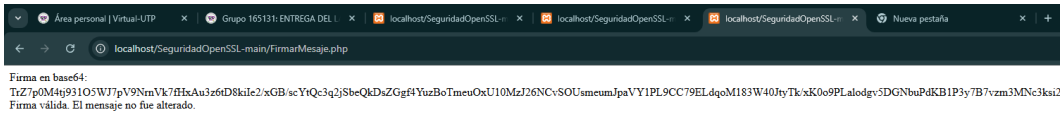
**3.3 Parámetros recibidos:**

Aunque el script no recibe parámetros externos, internamente maneja:

- $mensaje (string): Mensaje de texto que se desea firmar.
- $privateKey (string): Contenido de la clave privada en formato PEM.
- $firmaBase64 (string): Firma codificada en base64, útil para transmisión segura.
- $cert (string): Certificado X.509 (en formato PEM) que contiene la clave pública para la verificación.

**3.4 Valor devuelto:**

Imprime en pantalla la firma generada en formato base64 e indica si la firma es válida o inválida al verificarla, o muestra un mensaje de error en caso de fallo.




**3.5 Explicación breve del proceso criptográfico aplicado:**

El programa aplica un esquema de **firma digital con clave pública**, con el algoritmo **RSA** y la función hash **SHA-256**. El proceso tiene dos partes:

- **Firma del mensaje:**
  - Se carga la clave privada desde un archivo (privkey.pem).
  - Se calcula un hash del mensaje y se cifra con la clave privada mediante openssl_sign.
  - La firma resultante (binaria) se codifica en base64 para facilitar su transporte o almacenamiento.
- **Verificación de la firma:**
  - Se carga el certificado X.509 (certout.csr), del cual se extrae la clave pública.
  - Se decodifica la firma desde base64 a binario.
  - Se utiliza openssl_verify para verificar que el mensaje no haya sido alterado y que provenga del emisor legítimo.

Resultado de FirmarMensaje.php con la firma en base64 y la verificación exitosa del mensaje.

## Documentación de Funciones

1. openssl_pkey_new($configArgs): Crea un nuevo par de claves (privada/pública).
2. openssl_pkey_export(): Exporta la clave privada a una cadena legible.
3. openssl_csr_new(): Genera una solicitud de firma de certificado (CSR) con los datos del sujeto.
4. openssl_csr_sign(): Firma el CSR, generando un certificado X.509 válido.
5. openssl_sign(): Firma digitalmente un mensaje utilizando una clave privada.
6. openssl_verify(): Verifica la firma de un mensaje usando una clave pública o certificado.
7. base64_encode() / base64_decode(): Codifica/decodifica la firma para que pueda ser transportada de forma segura.

## Conclusiones Personales

El laboratorio permitió comprender de forma práctica el proceso de generación de claves, firma de mensajes y verificación mediante OpenSSL y PHP.

Se evidenció que la firma digital garantiza la integridad de los datos y que solo el titular de la clave privada puede realizar la firma.

Esta tecnología es fundamental en aplicaciones de seguridad web, autenticación y confidencialidad.