PRÁCTICA 3

Comunicación segura con SSL

Grupo-2361
Pareja 6

Celia Mateos de Miguel

Beatriz de Pablo García

Alfonso Sebares Mecha

Contenido

1. Introducción

En esta práctica se nos ha pedido desarrollar un servidor echo que se comunique con un cliente a través de una conexión segura con SSL.

SSL es un protocolo de nivel de aplicación diseñado para permitir conexiones seguras sobre TCP, proporcionando confidencialidad e integridad a la comunicación. Los mensajes se envían utilizando cifrado de clave simétrica, en la que el cliente y el servidor comparten una misma clave secreta que utilizan para posteriormente cifrar y descifrar los datos.

Esta clave es acordada entre ambos y se intercambia utilizando un protocolo de handshake basado en cifrado de clave pública. En este protocolo, el cliente y el servidor tienen dos claves: una pública que se distribuye libremente y una privada que ha de mantenerse en secreto, de manera que, el cliente y el servidor acuerdan de manera segura una clave sin conocerse previamente, y sin riesgo de que sea interceptada por terceros.

Como este mecanismo no garantiza la seguridad de la conexión se van a generar unas claves públicas que se van a intercambiar junto con campos de identificación por medio de certificados. Al iniciarse la comunicación, el cliente verifica los campos anteriormente nombrados y utiliza la clave pública proporcionada en el certificado para cifrar un mensaje de prueba. Para que la conexión sea satisfactible, el servidor se encargará de probar la identidad mediante su clave privada para descifrar el mensaje y enviarlo al cliente de nuevo.

2. Diseño

Hemos realizado un módulo con las funciones SSL necesarias para establecer comunicaciones entre un servidor y un cliente. De este modo, podemos utilizar esta librería para hacer el servidor y el cliente echo y para modificar nuestras prácticas y añadirlas seguridad. Con este fichero, hemos generado la librería G-2361-06_P3-lib-funciones_ssl.

La organización de los ficheros es:

- Carpeta **src**:
    - **G-2361-06-P3-echo_clientSSL.c**. Desarrollado el main que activa el funcionamiento del cliente echo con SSL.
    - **G-2361-06-P3-echo_serverSSL.c**. Desarrollado el main que activa el funcionamiento del servidor echo con SSL.
    - **G-2361-06-P1-echo_client.c**. Desarrollado el main que activa el funcionamiento del servidor echo sin SSL.
    - **G-2361-06-P2-echo_server.c**. Desarrollado el main que activa el funcionamiento del servidor echo sin SSL.
- Carpeta **srclib:**

        - **G-2361-06-P3-funciones_ssl.c**. En esta carpeta se encuentran las funciones para la implementación del encriptado SSL.

- Carpeta **includes**:

    - Aquí se encuentran cada uno de los .h que se corresponden a los distintos .c de la carpeta src.

- Carpeta **man**:

    - En esta carpeta se encuentra toda la documentación de cada uno de los ficheros generada con doxygen.

- Carpeta **certs**:
    - En esta carpeta se encuentran todos los certificados SSL.
    - **ca.pem**: certificado del CA.
    - **clientcert.pem**: certificado auto-firmado del cliente.
    - **servercert**.**pem**: certificado auto-firmado del servidor.
    - **cliente**.**pem**: certificado del cliente.
    - **servidor**.**pem**: certificado del servidor.
    - **clientkey**.**pem**: clave privada del cliente.
    - **serverkey.pem**: clave privada del servidor.

- Carpeta **echo**:
    - En esta carpeta se encuentran los ficheros de los ejecutables del servidor y del cliente echo con SSL.

En el directorio raíz se encuentra el Makefile. Con hacer make se compila el servidor y el cliente, a parte de generar el ejecutable y los certificados.

3. Funcionalidad IRC

La funcionalidad que se ha implementado para hacer la conexión segura con SSL es la siguiente:

- **Inicializar_nivel_SSL**: Realiza todas las llamadas necesarias para que la aplicación pueda usar la capa segura SSL.

- **Fijar_contexto_SSL**: Inicializa correctamente el contexto que será utilizado para la creación de canales seguros mediante SSL. Recibe información sobre las rutas a los certificados y claves con los que vaya a trabajar la aplicación.

- **Conectar_canal_seguro_SSL**: Obtiene un canal seguro SSL iniciando el proceso de handshake con el otro extremo dado un contexto SSL y un descriptor de socket.

- **Evaluar_post_conectar_SSL**: Comprueba una vez realizado el handshake que el canal de comunicación se puede considerar seguro.

- **Enviar_datos_SSL**: Envía datos a través del canal seguro. Es equivalente a la función de envío de mensajes que se hizo en la práctica 1 (creación del servidor IRC).

- **Aceptar_canal_seguro_SSL**: Bloquea la aplicación, que se quedará esperando hasta recibir un handsake por parte del cliente.

- **Recibir_datos_SSL**: Envía datos a través del canal seguro. Es equivalente a la función de lectura de mensajes que se hizo en la práctica 1 (creación del servidor IRC).

- **Cerrar_canal_SSL**: Libera todos los recursos y cierra el canal de comunicación seguro que se ha creado previamente.

4. Conclusiones Técnicas

Hemos realizado las funciones a parte por lo que hemos podido integrar la librería para que se pudiera usar en todos los proyectos.

5. Conclusiones Personales

Asimismo hemos aprendido a crear certificados SSL para que el cliente pueda verificar la identidad del servidor y viceversa además para la solicitud de firma de certificados.

La utilización de librerías es útil para utilizar las funciones que ya se habían implementado anteriormente, así como el uso de los manuales para saber cómo se utilizan las funciones.

Hemos intentado añadir dichas funciones en el servidor IRC y en el cliente IRC debido a que no conseguimos que realizara la función de impresión para las diferentes pruebas.