Escuela Politécnica Superior de Elche
Grado en Ingeniería Informática en Tecnologías de la Información
Área de Arquitectura y Tecnología de Computadores

Seguridad en Sistemas Informáticos

Práctica 2 (sesión 2)

Utilización de una librería criptográfica (Crypto++)

Introducción

Actualmente, uno de los dispositivos donde más se usan los algoritmos criptográficos es el ordenador. La mayoría de las soluciones de seguridad están basadas en la criptografía. Existen multitud de *librerías de programación* criptográficas que implementan los algoritmos de cifrado/descifrado que permiten realizar programas que incorporen características de seguridad sin tener que programar dichos métodos. En esta práctica se utilizará la librería criptográfica denominada Crypto++.

Objetivos de la práctica

- Conocer la librería criptográfica Crypto++.
- Utilizar dicha librería dentro de un programa.

Tareas a realizar

- Modificar el programa de ejemplo añadiendo las opciones indicadas más abajo.

Memoria de la práctica

La entrega de esta sesión de la práctica se realiza conjuntamente con la de la sesión 1 de la misma práctica. En el fichero **`test.cpp`** debe incluirse la nueva opción desarrollada en esta sesión y en el fichero **`usage2.dat`** debe ir la ayuda para el manejo de la misma. Para ver el detalle de los ficheros a incluir se debe consultar el guión de la sesión 1.

Características

Para el desarrollo de la práctica se utilizará el compilador Microsoft Visual C++. Este compilador es gratuito para los estudiantes de la UMH.

Desarrollo

Los pasos a realizar para el desarrollo de la práctica son los siguientes:

• Añadir opciones al programa de ejemplo.
Añadir la siguiente opción al programa:

- Opción "fuerzaB". Realiza un ataque por fuerza bruta para descifrar un texto cifrado con Triple-DES (DES-EDE) en modo CBC *(Cipher Block Chaining)*. Debe pedir por pantalla el texto cifrado, la longitud máxima de la clave a probar (1-4) y si se debe probar únicamente las letras o bien las letras y los dígitos. Como resultado se debe mostrar el texto descifrado y la clave (si se ha conseguido descifrar correctamente), o bien un mensaje indicando que no se ha podido descifrar el mensaje. Además debe aparecer el alfabeto con el que se han realizado las pruebas (letras o letras+dígitos) y el número de claves probadas. También debe aparecer el tiempo empleado en realizar la búsqueda de la clave (tanto en milisegundos como en segundos).

Para la clave se debe usar el alfabeto en minúsculas (incluyendo la letra ñ) y opcionalmente se deben incluir también los dígitos (0-9).

Esta nueva opción del programa se debe documentar adecuadamente en la parte final del fichero usage2.dat para que cuando se ejecute cryptest.exe sin parámetros aparezca la ayuda sobre su funcionamiento.

En el apartado "Memoria de la práctica" de la sesión 1 se indican los ficheros que se deberán entregar. En el apartado "Notas adicionales" se proporciona ayuda para la implementación de la opción pedida.

Notas adicionales

A continuación se proporciona ayuda para la realización de la práctica.

El programa de ejemplo de Crypto++ (*test.cpp*), cuando encripta un texto usando Triple-DES (DES-EDE) en modo CBC (opción "t"), adjunta al texto cifrado cierta información (calculando el valor *hash* de la *passphrase*) que sirve para comprobar si la *passphrase* usada cuando se descifra el mensaje es la correcta o no. En el caso de que al tratar de desencriptar el mensaje no se use la *passphrase* con la que se encriptó el texto, el programa genera una excepción que será capturada mediante el mecanismo de ***try-catch***, que informará de que la *passphrase* no es la correcta. Por este motivo no es posible desencriptar un texto cifrado con una *passphrase* diferente a la usada para encriptarlo. (Incluso si nos saltásemos la comprobación de que la clave es la correcta, al intentar descifrar el texto con una passphrase que no es la adecuada también se generaría una excepción, ya que, al descifrar el texto con una *passphrase* incorrecta, el mecanismo de *padding* usado nos indicaría que no se puede descifrar el texto correctamente).

Basándonos en la comprobación que hace el programa de si la *passphrase* es la correcta y usando el mecanismo de ***try-catch***, la opción a implementar irá probando todas las posibles claves (con longitud máxima de 1 a 4) y mostrará el texto en claro y la clave encontrada cuando se descifre el criptograma correctamente, o bien el mensaje "Clave no encontrada" si no se puede desencriptar correctamente. Para ello deberemos implementar un *try-catch* similar al que ya está implementado en el programa, pero dentro de la nueva opción "fuerzaB", de forma que, al intentar descifrar el mensaje con una clave, si se captura una excepción, se seguirá con la siguiente clave a probar, y, si se descifra el mensaje sin producirse ninguna excepción, entonces mostraremos los resultados y finalizaremos el programa.

Notas y ejemplos sobre los tipos de datos y las funciones

Para evitar errores al usar la letra ñ se deben definir los alfabetos de la forma siguiente:

```
char alfabeto_letras[] = "abcdefghijklmn\xA4opqrstuvwxyz";
```

y el alfabeto que incluye los dígitos:

```
char alfabeto_con_digitos[] = "abcdefghijklmn\xA4opqrstuvwxyz0123456789";
```

(Los errores con la letra ñ son debidos a que cuando se ejecuta el programa en modo consola se utiliza el valor ASCII de la ñ y en el entorno de programación se utiliza el valor UNICODE de la ñ, que son distintos).

Para seleccionar uno u otro alfabeto bastará hacer una copia de la dirección del alfabeto escogido a un puntero:

```
char *alfabeto;
alfabeto = alfabeto_letras;
```

(o en su caso) `alfabeto = alfabeto_con_digitos;`

Para saber la longitud del alfabeto y así poder recorrerlo se puede usar la función ***strlen*** de la forma explicada a continuación:

```
int longitud;
longitud = strlen (alfabeto);
```

Para calcular el tiempo empleado en la realización del ataque por fuerza bruta se pueden usar las funciones `QueryPerformanceCounter` y `QueryPerformanceFrequency`.

Notas sobre la evaluación de la práctica

La práctica se evaluará valorando el nivel de implementación del algoritmo de fuerza bruta. Por lo tanto es aconsejable ir desarrollándolo por etapas en lugar de afrontar la realización completa de la práctica en un único paso. Por ejemplo, en primer lugar se puede implementar la generación de todas y cada una de las claves a probar (y mostrarlas por pantalla, como comprobación de que se está haciendo correctamente). Una vez que esta etapa funcione correctamente, se puede añadir la llamada a la función de desencriptar con cada una de las claves correspondientes. Posteriormente se puede añadir el mecanismo de *try-catch* para que vaya descartando las claves incorrectas. Finalmente, se puede añadir el contador de tiempo para calcular lo que el programa tarda en realizar el ataque.