7° C

Universidad Tecnológica de Puebla

Tecnologías de la información Área Desarrollo de Software

Producto 2: Aplicación con algoritmos de encriptación

Integrantes de equipo:

- **Ramírez Huerta Juan**
- **Amaro Galindo Cecilia**
- **Romero Ramírez Miriam**
- **Aquino Juárez Armando**

Seguridad informática

Profesora: Mather Xochitl Mendoza Piscil

Septiembre-Diciembre 2021

Índice

1. Características de los algoritmos programados

1.1. Algoritmo: cifrado de cesar

El cifrado César consiste en sustituir cada letra del abecedario por una letra desplazada un número determinado de posiciones (clave).

- Tipo de cifrado por sustitución en el que una letra en el texto original es reemplazada por otra letra que se encuentra un número fijo de posiciones más adelante en el alfabeto.
- Puede formar parte de sistemas más complejos de codificación, como el cifrado Vigenère, e incluso tiene aplicación en el sistema ROT13.
- Se descifra con facilidad y en la práctica no ofrece mucha seguridad en la comunicación.
- Es sencillo si se dispone de una gran cantidad de texto cifrado, y se basa en el estudio de las frecuencias relativas de las letras en cada idioma.

1.2. Algoritmo: simétrico

Se basa en la utilización de una única clave secreta que se encargará de cifrar y descifrar la información, ya sea información en tránsito con protocolos como TLS, o información en un dispositivo de almacenamiento extraíble. La criptografía simétrica fue el primer método empleado para el cifrado de la información, se basa en que se utilizará la misma contraseña tanto para el cifrado como el descifrado, por tanto, es fundamental que todos los usuarios que quieran cifrar o descifrar el mensaje, tengan esta clave secreta, de lo contrario, no podrán hacerlo.

Hay varios puntos que debe cumplir un algoritmo de clave simétrica para que su uso sea seguro:

- Una vez que se cifra el mensaje, no se podrá obtener la clave de cifrado ni tampoco el mensaje en claro por ningún método.
- Si conocemos el mensaje en claro y el cifrado, se debe gastar más tiempo y más dinero en obtener la clave para acceder al mensaje en claro, que el posible valor que pueda tener la información que se consiga robar.

1.3. Algoritmo: Hash

MD5 son las siglas de Message-Digest Algorithm 5 (algoritmo de resumen de mensajes 5), que se utiliza para garantizar una transmisión de información completa y coherente. Es uno de los algoritmos hash más utilizados por las computadoras. Este algoritmo tiene como objetivo producir una cadena única de longitud fija, el valor hash o “resumen del mensaje”, para cualquier dato o “mensaje” dado.

- **Compresibilidad:** puede devolver una cadena encriptada MD5 de longitud fija para que los datos, cadenas, etc. se encripten con diferentes longitudes. (Por lo general, una cadena hexadecimal de 32 bits);
- **El cifrado es irreversible:** el proceso de cifrado es casi irreversible. A menos que se mantenga una enorme base de datos de valores clave para el descifrado de colisiones, es casi imposible desbloquearlo.
- **Fácil de calcular:** es fácil calcular el valor MD5 a partir de los datos originales.

- **Anti-modificación:** Para una cuerda fija. Números, etc., la cadena después del cifrado MD5 es fija, lo que significa que no importa cuántas veces se cifre MD5, se obtendrá el mismo resultado. Y si se modifica uno de los bytes, el valor MD5 obtenido varía mucho.
- **Fuerte anticolisión:** conociendo los datos originales y su valor MD5, es muy difícil encontrar un dato con el mismo valor MD5 (es decir, datos falsos).

  Basado en las características anteriores de MD5, a menudo se utiliza como un algoritmo de procesamiento de cifrado para contraseñas de usuario y otros datos importantes aplicables al cifrado irreversible.

  Implementación de la función hash

- Protección de contraseñas
- Se utilizan como parte de algunos de los pasos de los algoritmos de cifrado simétrico y asimétrico.
- Es una parte fundamental del mecanismo de firma digital.
- Se emplea para garantizar la integridad de un flujo de datos.

2. Código de la programación de algoritmos

2.1. Código: Cifrado de Cesar

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ClassCriptografiaCesar
{
    public class MetodosCesar
    {
        public string encriptarPass(string contrasena, int key)
        {
            string                                          abc                                                 =
"abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ0123456789.#@_$%/&" + "  ";
            string encriptado = "";
            for (int x = 0; x < contrasena.Length; x++)
            {
                string recorrido = contrasena.Substring(x, 1);
                int posicion = abc.IndexOf(recorrido);
                int encriptar = (posicion + key) % 73;
                if (encriptar < 0)
                {
                    encriptar = encriptar + 73;
                }
                encriptado = encriptado + abc.Substring(encriptar, 1);
            }
            return encriptado;
        }
        public string Descifrar(string msj, int clave)
        {
            string                                          abc                                                 =
"abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ0123456789.#@_$%/&" + "  ";
            string descifrado = "";
            for (int x = 0; x < msj.Length; x++)
            {
                string recorrido = msj.Substring(x, 1);
                int posicion = abc.IndexOf(recorrido);
                int descifrar = (posicion - clave) % 73;
                if (descifrar < 0)
                {
                    descifrar = descifrar + 73;
                }
                descifrado = descifrado + abc.Substring(descifrar, 1);
            }
            return descifrado;
        }
    }
}
```

2.2.Código: Cifrado Hash MD5

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Security.Cryptography;

namespace ClassEncriptadoHash
{
    public class MD5EncripDecenc
    {
        public string Encriptar(string texto, string llave)
        {
            //arreglo de bytes donde guardaremos la llave
            byte[] keyArray;
            //arreglo donde guardaremos el texto que se va a encripta
            byte[] ArregloCifrar = UTF8Encoding.UTF8.GetBytes(texto);
            //clases de encriptacion MD5
            MD5CryptoServiceProvider hashmd5 = new MD5CryptoServiceProvider();
            //se guarda la llave para que se realice hashing
            keyArray = hashmd5.ComputeHash(UTF8Encoding.UTF8.GetBytes(llave));
            hashmd5.Clear();
            TripleDESCryptoServiceProvider tdes = new
TripleDESCryptoServiceProvider();
            tdes.Key = keyArray;
            tdes.Mode = CipherMode.ECB;
            tdes.Padding = PaddingMode.PKCS7;
            //encriptacion de la cadena
            ICryptoTransform cTransform = tdes.CreateEncryptor();
            //arreglo de bytes donde se guarda la cadena cifrada
            byte[] ArrayResultado = cTransform.TransformFinalBlock(ArregloCifrar, 0,
ArregloCifrar.Length);
            tdes.Clear();
            //se regresa el resultado en forma de cadena
            return Convert.ToBase64String(ArrayResultado, 0, ArrayResultado.Length);
        }
        public string Desencriptar(string textoEncriptado, string llave)
        {
            byte[] keyArray;
            //convierte el texto en una secuencia de bytes
            byte[] ArrayDesifrar = Convert.FromBase64String(textoEncriptado);
            //md5
            MD5CryptoServiceProvider hashmd5 = new MD5CryptoServiceProvider();
            keyArray = hashmd5.ComputeHash(UTF8Encoding.UTF8.GetBytes(llave));
            hashmd5.Clear();
            TripleDESCryptoServiceProvider tdes = new
TripleDESCryptoServiceProvider();
            tdes.Key = keyArray;
```

```
            tdes.Mode = CipherMode.ECB;
            tdes.Padding = PaddingMode.PKCS7;
            ICryptoTransform cTransform = tdes.CreateDecryptor();
            byte[]  resultArray  =  cTransform.TransformFinalBlock(ArrayDesifrar,  0,
ArrayDesifrar.Length);
            tdes.Clear();
            //se regresa
            return UTF8Encoding.UTF8.GetString(resultArray);
        }
    }
}
```

2.3. Código: Cifrado simétrico AES

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Security.Cryptography;
using System.IO;
namespace ClassEncriptadoAsimetrico
{
    public class AESEncripDecenc
    {
        public static string Encrypt(string encryptString)
        {
            string EncryptionKey = "0";
            byte[] clearBytes = Encoding.Unicode.GetBytes(encryptString);
            using (Aes encryptor = Aes.Create())
            {
                Rfc2898DeriveBytes pdb = new Rfc2898DeriveBytes(EncryptionKey, new byte[] {
                0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d, 0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
                encryptor.Key = pdb.GetBytes(32);
                encryptor.IV = pdb.GetBytes(16);
                using (MemoryStream ms = new MemoryStream())
                {
                    using (CryptoStream cs = new CryptoStream(ms, encryptor.CreateEncryptor(),
CryptoStreamMode.Write))
                    {
                        cs.Write(clearBytes, 0, clearBytes.Length);
                        cs.Close();
                    }
                    encryptString = Convert.ToBase64String(ms.ToArray());
                }
            }
            return encryptString;
        }
        public static string Decrypt(string cipherText)
        {
            string EncryptionKey = "0";
            cipherText = cipherText.Replace(" ", "+");
            byte[] cipherBytes = Convert.FromBase64String(cipherText);
            using (Aes encryptor = Aes.Create())
```

```
            {
                Rfc2898DeriveBytes pdb = new Rfc2898DeriveBytes(EncryptionKey, new byte[] {
                0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d, 0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
                encryptor.Key = pdb.GetBytes(32);
                encryptor.IV = pdb.GetBytes(16);
                using (MemoryStream ms = new MemoryStream())
                {
                    using (CryptoStream cs = new CryptoStream(ms, encryptor.CreateDecryptor(),
CryptoStreamMode.Write))
                    {
                        cs.Write(cipherBytes, 0, cipherBytes.Length);
                        cs.Close();
                    }
                    cipherText = Encoding.Unicode.GetString(ms.ToArray());
                }
            }
            return cipherText;
        }
    }
}
```

3. Justificación de la programación de algoritmos

Los algoritmos que hemos elegido para encriptar información en la base de datos de nuestra aplicación web han sido del tipo simétrico con el algoritmo AES, HASH utilizando MD5 y el cifrado con el método de Cesar.

La información recabada en el sitio web no es mucha, únicamente se recolecta la información de un usuario al registrarse y así poder crearse un perfil con el que puede acceder a la información de la pagina dirigida hacia el Ecosistema Estatal De Robótica. Para evitar el robo de la información se han encriptado los siguientes datos al tomar un registro:

#	Dato recabado	Tipo de encriptación
1	Nombre completo	MD5
2	Apellido(s)	MD5
3	Correo electrónico	MD5
4	Nombre de usuario	AES
5	Contraseña	Cifrado de Cesar
6	Clave de seguridad	AES
7	Llave (Aleatorio)	AES

La ventaja al usar MD5 en los datos que se seleccionaron, fue debido a que es la información personal del usuario y por lo tanto es el único que puede consultar esa información si tiene consigo su clave de seguridad (palabra o frase preferida).

Utilizar el cifrado de Cesar nos permite encriptar nuestra contraseña con un método distinto al MD5, además de que se nos da una “llave” aleatoria para diferenciarse de las contraseñas de los demás usuarios.

Por último, AES fue el método de encriptado para tener nuestra base de datos aun mas segura, sobre todo para cifrar claves importantes, como la “llave” o código para encriptar la contraseña por el método Cesar. La clave de seguridad para este método es de forma predeterminado, es decir, los únicos que pueden modificarla son los desarrolladores, sin embargo, no habrá problema en cuanto obtener los datos que se solicitan, si la Clave de seguridad, la Llave o el Nombre de usuario coinciden con la información de la base de datos, se podrá acceder sin problemas a todas las operaciones que realice el usuario.

Bibliografía

El cifrado de Cesar. (s. f.). El cifrado de Cesar. Recuperado 1 de diciembre de 2021, de http://www.ugr.es/%7Eanillos/textos/pdf/2012/EXPO-1.Criptografia/02a04.htm

4. Algoritmo de cifrado hash - Seguridad Informatica - Victor Marek. (s. f.). Seguridad informática. Recuperado 1 de diciembre de 2021, de https://sites.google.com/site/sescinformaticavictormarek/seguridad-informatica-2osmr/unidad---4/4-algoritmo-de-cifrado-hash

CaracterÃsticas del algoritmo de cifrado MD5 e implementaciÃ³n simple (Java) - Code World. (s. f.). CodeTD. Recuperado 1 de diciembre de 2021, de https://www.codetd.com/es/article/12757791

Cifrado César. (s. f.). EduEscapeRoom. Recuperado 1 de diciembre de 2021, de https://eduescaperoom.com/cifrado-cesar/