УДК 004.031

Студ. В.А. Колодко
Науч. рук. ст. преп. Н.А. Жиляк
(кафедра информационных систем и технологий, БГТУ)

БЛОКНОТ 2.0

Вопрос об информационной безопасности на сегодняшний день стоит очень остро. Сегодня у каждого человека есть множество личных данных (такие как: пароли от банковских карт, пароли от аккаунтов, личные счета) и их конфиденциальность интересует каждого. Передавать данные по сети в «голом» виде не безопасно, так как существует множество программ для перехвата их. В этом и стоит вопрос о защите и способе шифровании данных.

Криптография зародилось еще до нашей эры. Уже в то время при передаче информации, пытались максимально защитить ее, чтобы недруги не смогли завладеть этой информацией. Одни из первых техник защиты текста: скитала, квадрат Полибия, шифр Цезаря, диск Энея.

Цель работы: разработать приложение по защите личной информации пользователя.

Реализованы следующие задачи и этапы для достижения поставленной цели:

1. ***Архитектура сетевого приложения***. Приложение должно состоять из клиентской части и серверной. Для удобства пользователя клиентская часть будет состоять из приложения для ПК и мобильного устройства. Сервер будет выполнять задачи хранилища, а клиент – отображать информацию и взаимодействовать с пользователем.

При разработке были использованы такие технологии как: язык программирования C#, WPF для написания клиентской части на ПК, Xamarin для написания клиентской части на мобильное устройство, EntityFramework для взаимодействия сервера и базы данных.

2. ***Передача зашифрованных данных***. Отправка осуществляется по принципу HTTPS. У клиента и сервера есть пара ключей (публичный и приватный) благодаря которым можно зашифровать и расшифровать данные. Данные передаются по протоколу TCP, после подключения сервер и клиент обмениваются публичными ключами, затем происходит общение между ними в зашифрованном виде.

```
byte[] bytes = new byte[1024];
int bytesRec = handler.Receive(bytes);

publickClientK = Encoding.GetEncoding(1251).GetString(bytes, 0, bytesRec);
handler.Send(Encoding.GetEncoding(1251).GetBytes(publicK));

bytesRec = handler.Receive(bytes);
```

Рисунок 1 – Обмен публичными ключами

3. ***Шифрование и дешифрование***. Когда клиент подключился к серверу, идет выборка из имеющихся шифровок. С помощью функции RandomEncryption, случайным образом выбирается способ шифрования на один сеанс. Так же, каждое сообщение перед отправкой подвергается транспозиции функцией TransPosition. То есть каждое сообщение, уходит в сеть задом наперед.

```
public String RandomEncryption()
{
    Random rand = new Random();

    switch (rand.Next(0, 3))
    {
        case 0: return "RSA";
        case 1: return "DES";
        case 2: return "AES";
        case 3: return "RC2";
    }
    return "";
}


public byte[] TransPosition(byte[] buff)
{
    buff.Reverse().ToArray();
    return buff;
}
```

Рисунок 2 – Функции RandomEncryption и TransPosition

4. ***Дополнительная защита.*** Сервер, получивший сообщение, сохраняет его в файле, название которого совпадает с логином пользователя, файл имеет формат “.nb2”.

```
byte[] bytes = new byte[1024];
int bytesRec = con.Receive(bytes);
String mess = Encoding.GetEncoding(1251).GetString(rsa.RSADycrypt(Encoding.GetEncoding(1251).GetBytes(Encoding.GetEncoding(1251)
con.Send(rsa.RSAEcrypt(Encoding.GetEncoding(1251).GetBytes("FileSave"), Encoding.GetEncoding(1251).GetBytes(publicClientK)));
con.Shutdown(SocketShutdown.Both);
con.Close();

StreamWriter savefile = new StreamWriter(@"Client\" + Login + ".nb2", false, System.Text.Encoding.Default);
savefile.WriteLine(mess);
savefile.Close();
```

Рисунок 3 – Сохранение информации в файл

Перед сохранением в файл, сообщение проходит шифрование, метод которого выбирается с помощью функции RandomEncryption. Вся информация о пользователе хранится в таблице БД, там же есть информация об времени последнего шифрования. Каждый час, сервер перезаписывает файл пользователя, без потери данных. Проверка времени производится с помощью функции TimeCheck. Вызов этой функции происходит каждые 5 минут. Благодаря этой системе, производится дополнительная защита информации пользователя.

```
public void ReEncrypt(String Login)
{
    byte[] buff = Decrypt(Login);
    buff = Encrypt(buff, Login, RandomEncryption());
    TransPosition(buff);
    ReSave(buff);
}


public void TimeCheck()
{
    var clients = se.Client.ToList();
    DateTime date = DateTime.Now;
    foreach(var client in clients)
    {
        if(date.Subtract(client.DataLastSave.Value).Minutes == 60)
        {
            ReEncrypt(client.ClientLogin);
        }
    }
}
```

Рисунок 4 – Функции ReEncrypt и TimeCheck

Таким образом, программа будет полезна для людей всех возрастов, которые хотят защитить личные данные и всегда иметь их под рукой, так как записи на бумаге не надежны.

ЛИТЕРАТУРА

1. WindowsPresentationFoundation[Электронный ресурс] / Режим доступа: https://docs.microsoft.com/en-us/dotnet/framework/wpf/ - Дата доступа: 15.11.2018.

2. XamarinDocumentation [Электронный ресурс] / Режим доступа: https://docs.microsoft.com/en-us/xamarin/ – Дата доступа: 10.04.2019.