УДК 004.021

КОМБИНИРОВАННЫЕ АЛГОРИТМЫ ШИФРОВАНИЯ, ЦЕЛЕСООБРАЗНОСТЬ ИХ ИСПОЛЬЗОВАНИЯ ДЛЯ ЗАЩИТЫ ФАЙЛОВ ОТ НЕСАНКЦИОНИРОВАННОГО ДОСТУПА

А.Г. АНДРЕЙЧИКОВ
(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)

Рассматриваются возможности комбинированных алгоритмов шифрования. Показаны преимущества и недостатки комбинированных алгоритмов при их использовании для защиты файлов от несанкционированного доступа.

Комбинированный (гибридный) метод шифрования позволяет сочетать преимущества высокой секретности, предоставляемые асимметричными криптосистемами с открытым ключом, с преимуществами высокой скорости работы, присущими симметричным криптосистемам с секретным ключом. При таком подходе криптосистема с открытым ключом применяется для шифрования, передачи и последующего дешифровки только секретного ключа симметричной криптосистемы. А симметричная криптосистема применяется для шифрования и передачи исходного открытого текста. В результате криптосистема с открытым ключом не заменяет симметричную криптосистему с секретным ключом, а лишь дополняет ее, позволяя повысить в целом защищенность передаваемой информации.

Для эффективного использование гибридных криптосистем необходимо рассмотреть отдельно симметричные и асимметричныеалгоритмышифрования их преимущества и недостатки.

Симметричные алгоритмы – это алгоритмы шифрования, в которых для шифрования и дешифровки используется один и тот же ключ, что, в свою очередь, обеспечивает высокую скорость работы.

Алгоритмы шифрования данных широко применяются в компьютерной технике в системах сокрытия конфиденциальной и коммерческой информации от злонамеренного использования сторонними лицами. Главным принципом в них является условие, что передатчик и приемник заранее знают алгоритм шифрования, а также ключ к сообщению, без которых информация представляет собой всего лишь набор символов, не имеющих смысла.

Классическими примерами таких алгоритмов являются симметричные криптографические алгоритмы:

- Простая перестановка.
- Одиночная перестановка по ключу.
- Двойная перестановка.
- Перестановка «Магический квадрат».

Основными достоинствами данного типа алгоритмов является скорость и простота реализации. Также стоит отметить, что при одинаковой длине ключа криптостойкость симметричных алгоритмов в несколько раз выше, чем асимметричных. Данный вид алгоритмов шифрования появился достаточно давно, и хорошо изучен.

Так как для шифрования и дешифровки используется один и тот же ключ, то самой большой проблемой при использовании данного типа алгоритмов является передача данного ключа. Передача ключа должна осуществляться по секретным каналам обеим сторонам. Эта проблема порождает ещё один важный недостаток – сложность управления ключами в случае большой сети, что сильно усложняет обмен данными и контроль за секретностью ключей.

Асимметричные криптографические системы были разработаны в 1970-х годах. Принципиальное отличие асимметричной криптосистемы от криптосистемы симметричного шифрования состоит в том, что для шифрования информации и ее последующего расшифровывания используются различные ключи:

- открытый ключ K используется для шифрования информации, вычисляется из секретного ключа $к$;
- секретный ключ $к$ используется для расшифровывания информации, зашифрованной с помощью парного ему открытого ключа K.

Эти ключи различаются таким образом, что с помощью вычислений нельзя вывести секретный ключ $к$ из открытого ключа K. Поэтому открытый ключ K может свободно передаваться по каналам связи.

Асимметричные системы называют также двухключевыми криптографическими системами, или криптосистемами с открытым ключом.

По сравнению с симметричными алгоритмами асимметричные имеют ряд преимуществ, однако есть и недостатки. Главное преимущество состоит в том, что нет необходимости предварительно передавать секретный ключ. Также преимуществом асимметричных алгоритмов перед симметричными является тот факт, что при использовании системы в большой сети количество ключей значительно меньше, что, в вою очередь, упрощает их распространение. В качестве главного недостатка выделим сложность

внесения изменений в алгоритм, поскольку при использовании алгоритма в практических целях часто возникает необходимость изменения алгоритма под конкретную задачу или проблему. Также нужно обратить внимание на большую размерность ключей по сравнению с симметричными алгоритмами при одинаковой криптостойкасти.

В качестве примера работы симметричного алгоритма реализована простая программа для шифрования введенной строки.

Листинг 1 – Пример алгоритма шифрования 3DES

```
public static string Encrypt(string source, string key)
      {
TripleDESCryptoServiceProviderdesCryptoProvider = new TripleDESCryptoServiceProvider();
         MD5CryptoServiceProvider hashMD5Provider = new MD5CryptoServiceProvider();

byte[] byteHash;
byte[] byteBuff;

byteHash = hashMD5Provider.ComputeHash(Encoding.UTF8.GetBytes(key));
desCryptoProvider.Key = byteHash;
desCryptoProvider.Mode = CipherMode.ECB; //CBC, CFB
byteBuff = Encoding.UTF8.GetBytes(source);

string encoded =
Convert.ToBase64String(desCryptoProvider.CreateEncryptor().TransformFinalBlock(byteBuff, 0, byte-
Buff.Length));
return encoded;
      }

    public static string Decrypt(string encodedText, string key)
      {
TripleDESCryptoServiceProviderdesCryptoProvider = new TripleDESCryptoServiceProvider();
         MD5CryptoServiceProvider hashMD5Provider = new MD5CryptoServiceProvider();

byte[] byteHash;
byte[] byteBuff;

byteHash = hashMD5Provider.ComputeHash(Encoding.UTF8.GetBytes(key));
desCryptoProvider.Key = byteHash;
desCryptoProvider.Mode = CipherMode.ECB; //CBC, CFB
byteBuff = Convert.FromBase64String(encodedText);

string plaintext = Encod-
ing.UTF8.GetString(desCryptoProvider.CreateDecryptor().TransformFinalBlock(byteBuff, 0, byte-
Buff.Length));
byte[] byt = desCryptoProvider.CreateDecryptor().TransformFinalBlock(byteBuff, 0, byteBuff.Length);

return plaintext;
      }
static void Main(string[] args)
{
Console.WriteLine("Введите строку для шифрования : ");
String str = Console.ReadLine();
Console.WriteLine("Введитеключдляшифрования :");
         String key = Console.ReadLine();

         String result = Encrypt(str, key);
Console.WriteLine("Результатшифрования :" + result);
         String deresult = Decrypt(result, key);
Console.WriteLine("Результат дешифровки с исходнимключём : " + deresult);
Console.ReadKey();

   }
```

Результат работы алгоритма 3DES представлены на рисунке 1.

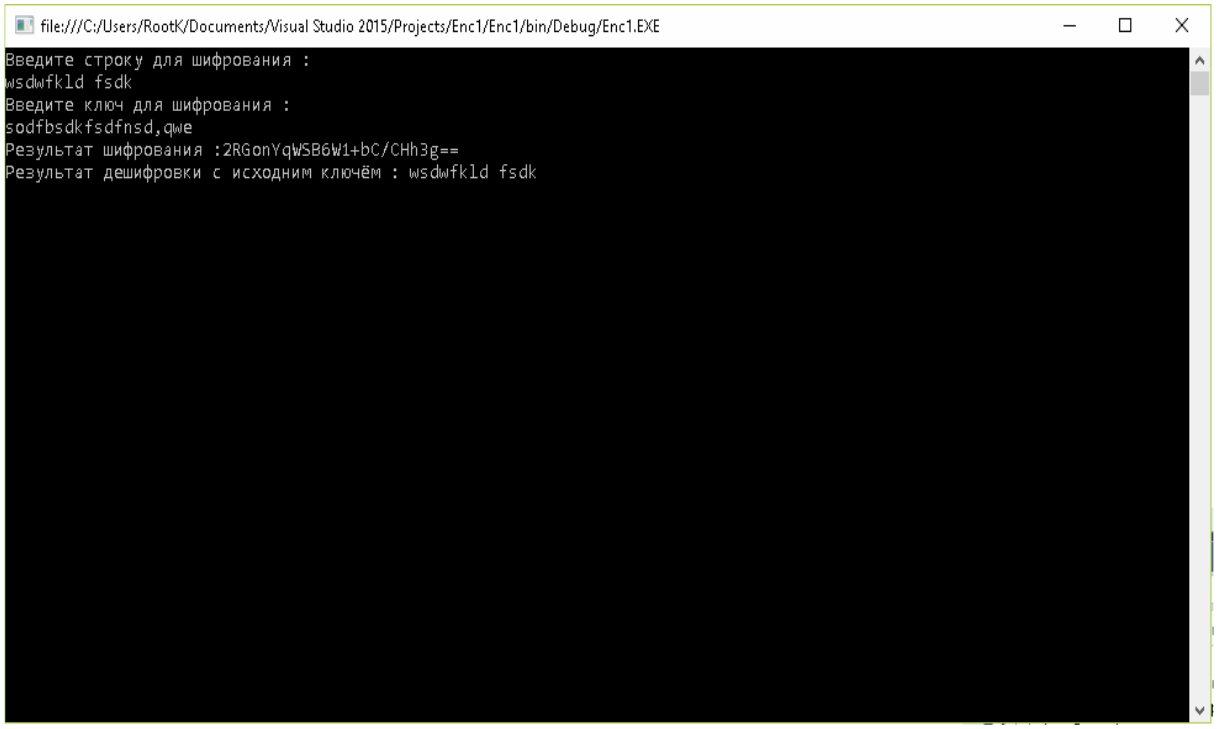

Рисунок 1. – Результат работы алгоритма 3DES

Заключение

В данной статье представлен краткий обзор особенностей симметричный и асимметричных алгоритмов шифрования. Показаны преимущества и недостатки этих типов алгоритмов шифрования. В качестве примера симметричного типа шифрования представлен алгоритм 3DES и результаты работы простейшей программы с его использованием.

ЛИТЕРАТУРА

1. Хабрахабр [Электронный ресурс] Пара слов о гибридном шифровании и эллиптических кривых. – Режим доступа: https://habrahabr.ru/post/106057. – Дата доступа: 24.09.2017.
2. Хабрахабр [Электронный ресурс] Немного практической криптографии под .NETдля чайников. – Режим доступа: https://habrahabr.ru/post/254909. – Дата доступа: 24.09.2017.
3. Хабрахабр [Электронный ресурс] RSA, а так ли все просто. – Режим доступа: https://habrahabr.ru/post/99376. – Дата доступа: 24.09.2017.