**УВАЖАЕМЫЕ СТУДЕНТЫ! Изучите теоретические сведения к лабораторной работе, выполните практическое задание, дайте ответы на контрольные вопросы.**

**Результаты работы, фотоотчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 20.03.2023.**

**Требования к отчету:**

**Отчет предоставляется преподавателю в электронном варианте и должен содержать:**

- **название работы, постановку цели, вывод;**
- **ответы на контрольные вопросы, указанные преподавателем.**

**При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini**

***ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).***

## Лабораторная работа № 35

## Тема: «Шифрование информации с использованием стандарта DES»

Цель: получение навыков работы с пространством CryptoAPI для шифрования информации; разработка Web-приложения для шифрования информации с использование м стандарта DES.

### Теоретическая часть

Симметричные методы, используют один и тот же ключ для шифрования и дешифрования, поэтому его нужно хранить в секрете от третьих лиц, поскольку, зная ключ и имея соответствующий криптоалгоритм, злоумышленник может дешифровать данные. Симметричные системы отличаются высокой скоростью работы (по сравнению с асимметричными), что позволяет применять их для шифрования крупных объёмов данных.

Симметричные криптосистемы делятся в свою очередь ещё на две группы: на блочные и поточные методы.

Блочные шифры применяют одно и то же преобразование к тексту, разбитому на блоки, длина которых может быть равна 8, 16, 24, 32 байтам в зависимости от криптометода. Криптосистема, предоставляемая .NET, работает по принципу построения цепочки блочных шифров (Cipher Block Chaining - CBC), которая использует ключ и вектор инициализации (Initialization Vector - IV). Простая блочная система шифрования, не использующая вектор инициализации, преобразует один и тот же блок исходного текста в тот же самый блок зашифрованного текста, т. е. безо всяких перемещений и рекомбинаций. Если у вас был дублированный блок в исходном тексте, то он появится и в зашифрованном. В результате, зная структуру исходного текста, злоумышленник может дешифровать соответствующую часть криптограммы и определить ключ шифрования. Чтобы этого избежать, в технологии CBC информация из предыдущего блока внедряется в шифруемый следующий блок. Поскольку такой подход использует предыдущий блок для шифрования следующего, то IV шифрует самый первый блок. Это позволяет защитить заголовок (первый блок), чтобы взломщик не мог использовать его для получения ключа.

Другая группа симметричных криптосистем - поточные методы. Эти методы применяют изменяющиеся во время шифрования преобразования к наборам символов, образующим единый исходный текст.

Среди симметричных алгоритмов можно выделить следующие: DES (Data Encryption Standard), DES 2, различные вариации TripleDES, Rijndael/AES (Advanced Encryption Standard), RC2, RC4, ГОСТ 28147-89...

Если взглянуть на их алгоритмы, то очевидно, что многие из них основаны на операторе XOr. Вообще, впервые оператор " исключающий или " был применён в так называемом методе одноразовых блокнотов, изученном Клодом Шенноном. Действовал этот метод по следующему принципу: бралась строка исходного текста и строка ключа такой же длины, после чего

каждый символ исходного текста XOr'ился с соответствующим символом строки ключа. Этот метод работает и как шифровщик, и как дешифровщик, поскольку повторный вызов функции XOr возвращает исходное значение.

Инфраструктура .NET Framework содержит классы для работы со следующими методами: DES, TripleDES, RC2, Rijndael. Но если воспользоваться услугами CryptoAPI, то можно также получить возможность работать с поточным симметричным методом шифрования RC4. В большинстве случаев применение криптографических инструментов в среде .NET Framework сводится к обращению к пространству имён System.Security.Cryptography, поскольку именно в нём содержатся все основные классы и интерфейсы для выполнения криптоопераций.

Для выполнения нашей лабораторной работы мы воспользуемся созданием Web-приложения.

ASP.NET - это последняя платформа веб-разработки от Microsoft. Хотя ASP.NET имеет некоторые сходства с предыдущей версией, которая называлась ASP (Active Server Pages - активные серверные страницы), ASP.NET была полностью переработана на основе .NET Framework.

Чтобы создать веб-приложение на Visual Basic .NET, вы создаете в среде разработки Visual Basic новый проект ASP.NET Web Site (Web-приложение ASP.NET), а затем используете Web Forms Designer (Конструктор Web Forms) для создания одной или нескольких веб-форм, которые будут представлять вашу программу. Каждая веб-форма содержит две части - страницу Web Forms и файл базового кода. Страница Web Forms содержит HTML и элементы управления, создающие интерфейс пользователя. Файл базового кода - это модуль с кодом программы, который выполняется за рамками страницы Web Forms. Это разделение концептуально похоже на формы Windows, которые вы создавали на Visual Basic - существует компонента интерфейса пользователя и компонента модуля кода. Код обеих этих компонент может храниться в одном файле с

расширением .aspx, но обычно страница Web Forms хранится в файле .aspx, а файл кода хранится в файле .aspx.vb.

На следующей иллюстрации показана концепция того, как веб-приложение с использованием ASP.NET отображается в веб-браузере.


В дополнение к веб-формам, веб-приложения могут содержать модули кода (файлы .vb), страницы HTML (файлы .htm), конфигурационную информацию (файл web.config), общую информацию о веб-приложении (файл Global.asax) и другие компоненты. Для быстрого и эффективного переключения между этими компонентами можно использовать Конструктор Web Forms и Обозреватель решений.

## Практическая часть

**Задание 1.**

*Создание нового Web -приложения*

1. Запустите Visual Studio и откройте пункт меню File.

2. В контекстном меню выберите New щелкните на Web Site. Когда вы выбираете этот значок, Visual Studio подготавливает среду разработки и файлы вашей программы для интернет-программирования. Создание нового проекта веб-приложения ASP.NET аналогично созданию проекта Windows Application. Однако текстовое поле Name (Имя) отключено, а текстовое поле Location (Расположение) предназначено для другого типа установки. В среде

веб-приложения вам предлагается указать веб-сервер для вашего проекта или принять значение по умолчанию **http://localhost**. При создании проекта вы можете выбрать для него локальный или удаленный веб-сервер (на котором установлены .NET Framework и файлы поддержки), и Visual Studio будет использовать указанный веб-сервер для размещения и организации файлов вашего проекта. Веб-сервер определяется не с помощью имен диска и папки, а с помощью корректного адреса в интернете (URL).

**Создание Web – формы**

Как отмечалось ранее, Web Forms хранится в файле .aspx. В нашем случае это будет default.aspx. Для создания интерфейса мы можем:

1. Оставаться в окне редактора кода и формировать интерфейс с использование стандартных html-тэгов

2. Перейти в режим конструктора и наполнить форму необходимыми компонентами.

При использовании второго метода заполнения необходимо перейти в режим конструктора (кнопка Design) и перетащить (удерживая компонент левой кнопкой мыши) на форму необходимые компоненты (Button и текстовое поле). Отредактируйте свойства этих компонентов в соответствии со следующим кодом (для просмотра редактора перейдем в соответствующее окно кода нажав кнопку Sourse):

```
<%@ Page Language="vb" AutoEventWireup="false" Inherits="CryptoTest.WebForm1" CodeFile="default.aspx.vb" %>
<html>
  <head>
    <title>CryptoTest</title>
  </head>
  <body>

    <form id="Form1" method="post" runat="server">
        <b>Clear Text:</b><br>
        <textarea id="txt" runat=server></textarea>
        <asp:Button ID="btnEncrypt" Text="Encrypt" Runat=server/><br><br>

        <b>Encrypted Text:</b><br>
        <table><tr><td bgcolor=LightGrey>
            <asp:Label ID="lblResult" Runat=server/>
        </td></tr></table>
    </form>

  </body>
</html>
```


**Реализация стандарта DES**

Все основные действия происходят в обработчике события нажатия кнопки btnEncrypt, поэтому необходимо вызвать этот обработчик, в нашем случае это default.aspx.vb, и прописать туда соответствующий участок кода. Для начала надо подключить компоненты шифрования и созданную нами Web - форму следующим кодом:

```
Imports System.Security.Cryptography
Imports System.Text
Imports System.IO


Namespace CryptoTest


Partial Class WebForm1
    Inherits System.Web.UI.Page
```

Давайте рассмотрим подробнее событие btnEncrypt_Click. Вначале создаётся экземпляр провайдера шифрования. Для каждого криптометода существует свой провайдер, например, DESCryptoServiceProvider, RSACryptoServiceProvider, RijndealManaged и др. Их имена объявлены в следующих форматах: <ИмяКриптометода>CryptoServiceProvider или <ИмяКриптометода>Managed.

```
Web Form Designer Generated Code

    Private Sub btnEncrypt_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnEncrypt.Click

        Dim DES As New DESCryptoServiceProvider
```

После инициализации криптопровайдера создаётся переменная, реализующая интерфейс-шифратор:

```
Dim DES_Encryptor As ICryptoTransform = DES.CreateEncryptor
```

Далее открывается поток для записи данных в файл:

```
Dim fs As New FileStream(Server.MapPath("temp.dat"), FileMode.Create)
```

После этого инициализируется криптопоток, в аргументах которого задаётся конечный поток данных (в нашем случае - это файловый поток **fs**); способ криптографической трансформации, т. е. интерфейс-шифратор, и действие, которое необходимо выполнить с данными. Из возможных действий выделяются **чтение** и **запись**.

```
Dim DESCryptoStream As New CryptoStream(fs, _
    DES_Encryptor, CryptoStreamMode.Write)
```

В качестве конечного потока данных (первый атрибут конструктора класса CryptoStream) чаще всего выступает файловый поток, но это вовсе не означает, что вы не можете использовать иные типы потоков. Например, в ASP.NET-приложениях можно использовать поток Response.OutputStream для вывода информации.

На этом возведение подготовительной площадки для дальнейших криптоопераций заканчивается. В следующих строках кода создаётся класс для получения массив байтов и получаем массив байтов из строки поля txt, поскольку, как уже было сказано, все криптометоды среды .NET Framework работают не со строками, а с массивами байтов:

```
Dim enc As New UnicodeEncoding, bytes() As Byte

bytes = enc.GetBytes(txt.Value)
```

Наконец, происходит шифрование:

```
DESCryptoStream.Write(bytes, 0, bytes.Length)
```

В этом отрезке

кода происходят те же действия, что и при обычных операциях с потоками: методу Write передаётся массив байтов, указывается смещение и длина этого массива. Таким образом, весь процесс шифрования уместился всего в одну строчку кода, но перед этим пришлось выстроить массивный подготовительный плацдарм!

В конце все потоки закрываются

```
DESCryptoStream.Close()
fs.Close()
fs = Nothing
```

потом открываются снова, но уже для чтения из предварительно созданного файла (temp.dat)

```
fs = New FileStream(Server.MapPath("temp.dat"), FileMode.Open)
```

инициализируется класс-читатель потока и считывается шифр-текст и файлового потока

```
Dim sr As New StreamReader(fs)
lblResult.Text = sr.ReadToEnd
```

Далее файловый поток вновь закрывается, а временный файл удаляется.

```
        sr.Close()
        fs.Close()

        File.Delete(Server.MapPath("temp.dat"))
    End Sub
End Class

End Namespace
```

В итоге мы можем видеть шифр-текст в поле Encrypted Text.


Если ваше приложение не работает, проверьте подключение компонента .NetFramework AssemblyInstaller. Для этого в меню Tools-ChooseToolboxItems...выбрать вкладку .Net Framework Components и поставить галочку возле компонента AssemblyInstaller, нажать OK.

Сохраните созданное вами приложение.

**Задание 2.**

Предыдущий пример был простейшим: мы только зашифровали текст без передачи каких-либо дополнительных параметров. Но как было сказано, что для эффективной защиты на симметричные криптометоды нужно накладывать ключ и **инициализационный вектор** (IV). Это мы и осуществим в данном задании, а также, чтобы не повторяться, откажемся от файлового потока в пользу потока вывода Response.OutputStream.

1. Создайте новый Web - сайт по аналогии с предыдущим. Количество элементов на форме уменьшится, т. к. для отображения результата используется поток вывода объекта Response, поэтому место для записи шифр-текса на форме определять не будем, т.е. будет отсутствовать такой фрагмент html-кода основной формы:

```
<b>Encrypted Text:</b><br>
<table><tr><td bgcolor=LightGrey>
    <asp:Label ID="lblResult" Runat=server/>
</td></tr></table>
```

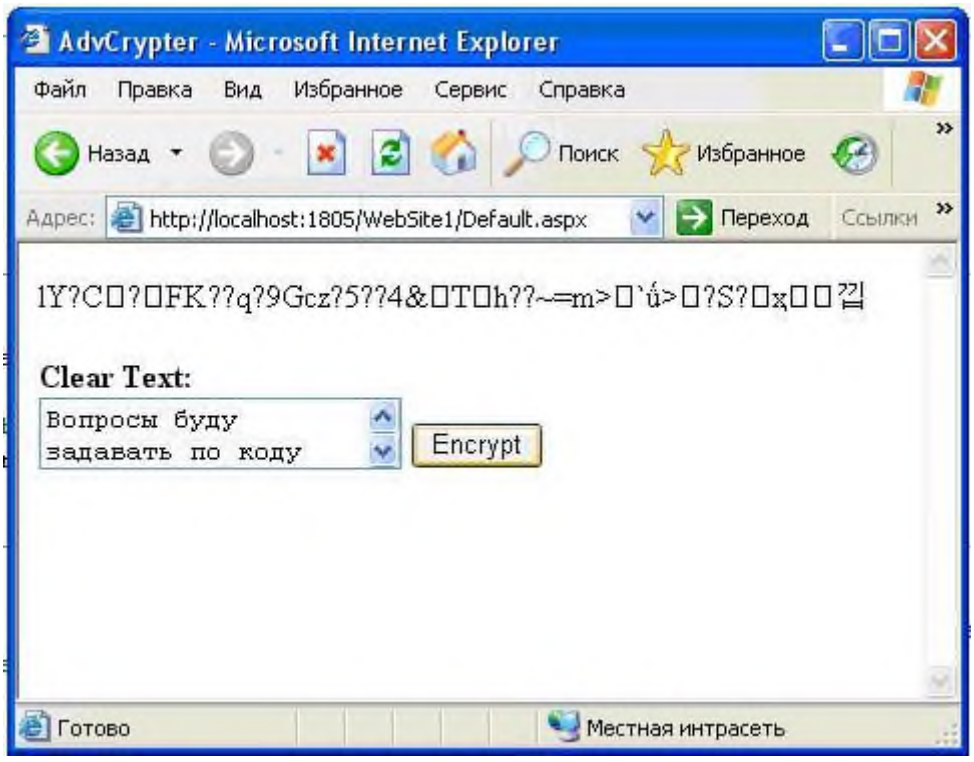
2. Как и в предыдущем случае, основные действия происходят в обработчике события нажатия кнопки btnEncrypt, поэтому необходимо вызвать этот обработчик и прописать туда практически такой же участок кода как и в первом случае. В коде появятся 2 новые строки, в которых через провайдер алгоритма DES генерируются ключ (DES.GenerateKey()) и инициализационный вектор (DES.GenerateIV()).

Затем для получения ссылки на интерфейс трансформации используется другой прототип перегруженной функции DES.CreateEncryptor, в котором в качестве аргументов передаются ключ и инициализационный вектор (*Dim* DES_Encryptor *As* ICryptoTransform = DES.CreateEncryptor(DES.Key, DES.IV)).

GenerateKey и GenerateIV - это (с точки зрения VB.NET) не функции, а процедуры, т. е. они не возвращают значения, а задают свойства Key и IV в объекте SymmetricAlgorithm. Именно поэтому в коде сначала вызываются функции генерации, а потом значения берутся из свойств Key и IV. Таким образом, код должен выглядеть следующим образом:

```
Imports System.Security.Cryptography
Imports System.Text


Namespace AdvCrypter
    Partial Class WebForm1
        Inherits System.Web.UI.Page
#Region " Web Form Designer Generated Code "
        <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()
        End Sub

        Private Sub Page_Init(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Init
            InitializeComponent()
        End Sub

#End Region

        Private Sub btnEncrypt_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnEncrypt.Click
            Dim DES As New DESCryptoServiceProvider
            DES.GenerateKey()
            DES.GenerateIV()

            Dim DES_Encryptor As ICryptoTransform = DES.CreateEncryptor(DES.Key, DES.IV)
            Dim DESCryptoStream As New CryptoStream(Response.OutputStream, _
                 DES_Encryptor, CryptoStreamMode.Write)
            Dim enc As New UnicodeEncoding, bytes() As Byte

            bytes = enc.GetBytes(txt.Value)
            DESCryptoStream.Write(bytes, 0, bytes.Length)
            DESCryptoStream.Close()
        End Sub
    End Class

End Namespace
```

а рабочий проект:




## Оформление отчета

**Включить в отчет**

1.Тема лабораторной работы

2. Цель лабораторной работы

3. Созданные вами Web формы и их назначение

4. Программа шифрования информации (в электронном или письменном виде)

**Контрольные вопросы:**

1. К какому методу шифрования относится криптостандарт DES?

2. Какое действие предполагает следующий участок кода:

3. В чем разница между шифрованием с использованием вектора инициализации и без него?

4. Укажите структуру инициализации криптопровайдера DES.

5. Какое действие предполагает следующий участок кода:

```
Dim sr As New StreamReader(fs)

lblResult.Text = sr.ReadToEnd
```