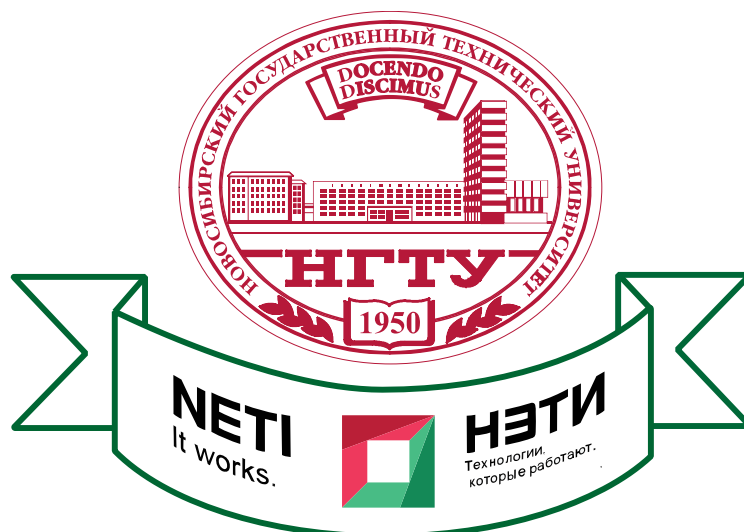


Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования

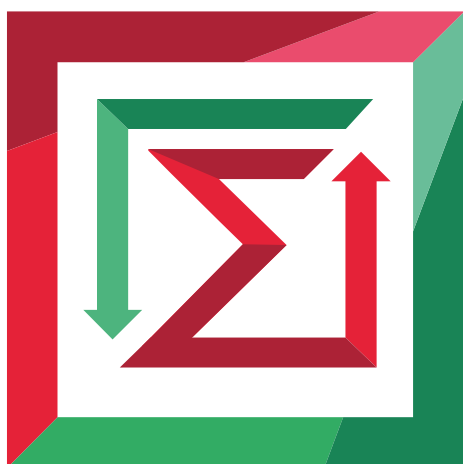
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



Прикладной математики

Практическая работа № 4
по дисциплине «Основы криптографии»

Симметричные криптографические алгоритмы



Факультет:	ПМИ
Группа:	ПМ-81
Вариант:	1
Студент:	Ефремов Артур
Преподаватель:	Ступаков Илья Михайлович

Новосибирск

2020

1. Цель работы

Научится использовать готовые криптографические примитивы для шифрования данных.

2. Задание

Обязательная часть (15 баллов)

Сделать программу которая шифрует и дешифрует некоторый файл с помощью алгоритма AES.

В качестве ключа использовать хеш (с возможностью выбора алгоритма) от вводимого пользователем пароля. Сам ключ в итоге нигде сохраняться не должен. Использовать режим CBC и в качестве IV взять ключ (такой подход считается плохим, подумайте почему).

Сделать селфи бригады (можно по отдельности и составить коллаж), зашифровать и выложить на общий диск. Изображение и пароль добавить в отчет.

Доп. Задания

(5 баллов) Сделать чтобы шифровалась только часть файла отвечающая за данные, т.е. после шифрования файл должен остаться валидным изображением и корректно открываться.

Привести зашифрованное изображение в отчете. Зашифровать и привести изображение используя режим ECB.

3. Текст программы

```
using System;
using System.Linq;
using System.IO;
using System.Security.Cryptography;
using System.Text;
using System.Drawing;
using System.Collections.Generic;

namespace CryptLab4
{
    class Program
    {
        static public string path = "../../../files/";
        static void Main(string[] args)
        {
            bool repeat = true;

            while (repeat)
            {
                Console.WriteLine("Выберете опцию:");
                Console.WriteLine("<1> Зашифровать файл");
                Console.WriteLine("<2> Дешифровать файл");
                Console.WriteLine("<3> Зашифровать только пиксели файла");
                Console.WriteLine("<4> Дешифровать только пиксели файла");
                Console.WriteLine("<0> Выйти(");

                string choice = Console.ReadLine();

                switch (choice)
                {
                    case "1":
                    {
                        Console.WriteLine("Введите название файла для шифрования:");
                        string inFileName = Console.ReadLine();
                        byte[] toEncrypt;
                        try
                        {
                            toEncrypt = File.ReadAllBytes(path + inFileName);
                        }
                        catch (Exception)
                        {
                            Console.WriteLine("Нет такого файла!");
                            break;
                        }
                    }

                    Console.WriteLine("Введите название файла для вывода:");
```

```

        string outFileName = Console.ReadLine();

        Console.WriteLine("Введите пароль для шифрования:");
        byte[] password = Encoding.UTF8.GetBytes(Console.ReadLine());

        File.WriteAllBytes(path + outFileName, Encrypt(toEncrypt, password,
        GetHashAlgorithm(), CipherMode.CBC, PaddingMode.PKCS7));

        Console.WriteLine("Шифрование прошло успешно!");
        break;
    }
    case "2":
    {
        Console.WriteLine("Введите название файла для дешифровки:");
        string inFileName = Console.ReadLine();
        byte[] toDecrypt;
        try
        {
            toDecrypt = File.ReadAllBytes(path + inFileName);
        }
        catch (Exception)
        {
            Console.WriteLine("Нет такого файла!");
            break;
        }

        Console.WriteLine("Введите название файла для вывода:");
        string outFileName = Console.ReadLine();

        Console.WriteLine("Введите пароль для дешифровки:");
        byte[] password = Encoding.UTF8.GetBytes(Console.ReadLine());

        try
        {
            File.WriteAllBytes(path + outFileName, Decrypt(toDecrypt, password,
            GetHashAlgorithm(), CipherMode.CBC, PaddingMode.PKCS7));
        }
        catch (Exception)
        {
            Console.WriteLine("Неверный пароль или алгоритм генерации хеша!");
            break;
        }
        Console.WriteLine("Дешифровка прошла успешно!");
        break;
    }
    case "3":
    {
        Console.WriteLine("Введите название файла для шифрования:");
        string inFileName = Console.ReadLine();

        Bitmap img;
        try
        {
            img = new Bitmap(path + inFileName);
        }
        catch (Exception)
        {
            Console.WriteLine("Нет такого файла!");
            break;
        }

        byte[] toEncrypt = PixelsToBytes(img);

        Console.WriteLine("Введите название файла для вывода:");
        string outFileName = Console.ReadLine();

        Console.WriteLine("Введите пароль для шифрования:");
        byte[] password = Encoding.UTF8.GetBytes(Console.ReadLine());
    }
}

```

```

        byte[] encData = Encrypt(toEncrypt.ToArray(), password, GetHashAlgorithm(),
CipherMode.ECB, PaddingMode.None);

        SetPixels(img, encData);

        img.Save(path + outFileName, System.Drawing.Imaging.ImageFormat.Png);

        Console.WriteLine("Шифрование прошло успешно!");
        break;
    }
    case "4":
    {
        Console.WriteLine("Введите название файла для дешифровки:");
        string inFileName = Console.ReadLine();

        Bitmap img;
        try
        {
            img = new Bitmap(path + inFileName);
        }
        catch (Exception)
        {
            Console.WriteLine("Нет такого файла!");
            break;
        }

        byte[] toDecrypt = PixelsToBytes(img);

        Console.WriteLine("Введите название файла для вывода:");
        string outFileName = Console.ReadLine();

        Console.WriteLine("Введите пароль для дешифровки:");
        byte[] password = Encoding.UTF8.GetBytes(Console.ReadLine());

        byte[] decData = Decrypt(toDecrypt.ToArray(), password, GetHashAlgorithm(),
CipherMode.ECB, PaddingMode.None);

        SetPixels(img, decData);

        img.Save(path + outFileName, System.Drawing.Imaging.ImageFormat.Png);

        Console.WriteLine("Дешифровка прошла успешно!");
        break;
    }
    case "0":
    {
        repeat = false;
        break;
    }
    default:
    {
        Console.WriteLine("Неверный выбор!");
        break;
    }
}
}
}

public static HashAlgorithm GetHashAlgorithm()
{
    bool repeat = true;
    HashAlgorithm res = HashAlgorithm.Create("SHA-256");

    while (repeat)
    {
        repeat = false;
        Console.WriteLine("Выберете алгоритм генерации хеша:");
        Console.WriteLine("<1> SHA-256");
        Console.WriteLine("<2> MD5");
    }
}

```

```

        Console.WriteLine("<3> SHA-512");

        string hashChoice = Console.ReadLine();

        switch (hashChoice)
        {
            case "1":
            {
                res = HashAlgorithm.Create("SHA-256");
                break;
            }
            case "2":
            {
                res = HashAlgorithm.Create("MD5");
                break;
            }
            case "3":
            {
                res = HashAlgorithm.Create("SHA-512");
                break;
            }
            default:
            {
                repeat = true;
                Console.WriteLine("Неверный выбор!");
                break;
            }
        }
    }
    return res;
}

public static byte[] Encrypt(byte[] dataToEncrypt, byte[] password, HashAlgorithm hashAlg,
CipherMode cipherMode, PaddingMode paddingMode)
{
    using (var aes = Aes.Create())
    {
        aes.Mode = cipherMode;
        aes.Key = hashAlg.ComputeHash(password).ToList().Take(32).ToArray();
        aes.IV = aes.Key.ToList().Take(16).ToArray();
        aes.Padding = paddingMode;

        return PerformCrypt(aes.CreateEncryptor(), dataToEncrypt);
    }
}

public static byte[] Decrypt(byte[] dataToDecrypt, byte[] password, HashAlgorithm hashAlg,
CipherMode cipherMode, PaddingMode paddingMode)
{
    using (var aes = Aes.Create())
    {
        aes.Mode = cipherMode;
        aes.Key = hashAlg.ComputeHash(password).ToList().Take(32).ToArray();
        aes.IV = aes.Key.ToList().Take(16).ToArray();
        aes.Padding = paddingMode;
        var decryptor = aes.CreateDecryptor();

        return PerformCrypt(aes.CreateDecryptor(), dataToDecrypt);
    }
}

public static byte[] PerformCrypt(ICryptoTransform transform, byte[] data)
{
    using (var msDecrypt = new MemoryStream())
    {
        using (var csEncrypt = new CryptoStream(msDecrypt, transform, CryptoStreamMode.Write))
        {
            csEncrypt.Write(data);
            csEncrypt.FlushFinalBlock();
        }
    }
}

```

```

        return msDecrypt.ToArray();
    }
}

public static byte[] PixelsToBytes(Bitmap img)
{
    int w = img.Width;
    int h = img.Height;
    byte[] res = new byte[h * w * 4];


    for (int i = 0; i < h; i++)
        for (int j = 0; j < w; j++)
        {
            Color pixel = img.GetPixel(j, i);
            res[(i * w + j) * 4 + 0] = pixel.A;
            res[(i * w + j) * 4 + 1] = pixel.R;
            res[(i * w + j) * 4 + 2] = pixel.G;
            res[(i * w + j) * 4 + 3] = pixel.B;
        }
    return res;
}

public static void SetPixels(Bitmap img, byte[] bytes)
{
    int w = img.Width;
    int h = img.Height;

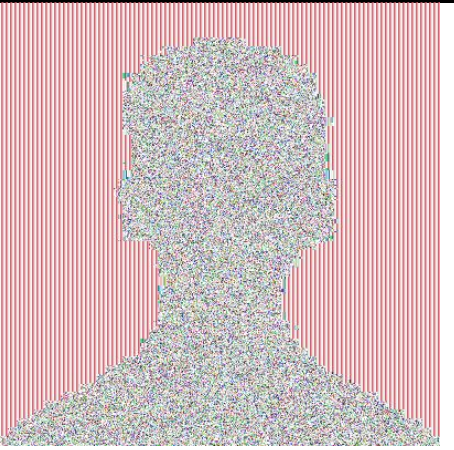

    for (int i = 0; i < h; i++)
        for (int j = 0; j < w; j++)
        {
            byte A = bytes[(i * w + j) * 4 + 0];
            byte R = bytes[(i * w + j) * 4 + 1];
            byte G = bytes[(i * w + j) * 4 + 2];
            byte B = bytes[(i * w + j) * 4 + 3];
            img.SetPixel(j, i, Color.FromArgb(A, R, G, B));
        }
}
}
}

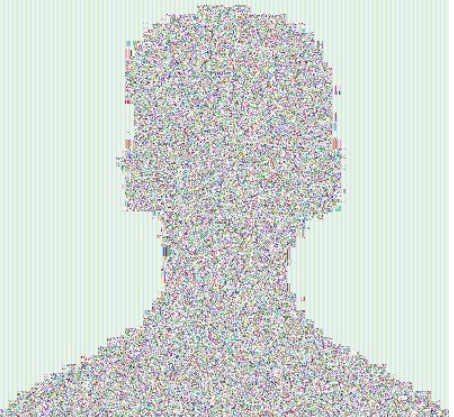

```

4. Тестирование


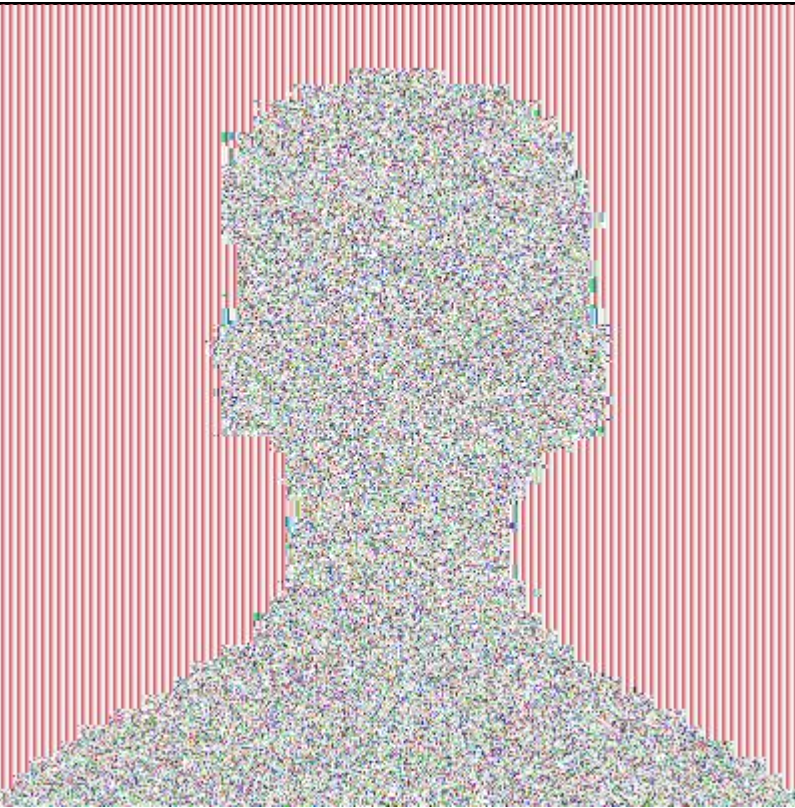
№	Тест	Назначение	Результат
1	<p>Выберете опцию:</p> <p><1> Зашифровать файл</p> <p><2> Дешифровать файл</p> <p><3> Зашифровать только пиксели файла</p> <p><4> Дешифровать только пиксели файла</p> <p><0> Выйти(</p> <p>1</p> <p>Введите название файла для шифрования:</p> <p>Arthur.png</p> <p>Введите название файла для вывода:</p> <p>Arthur_encrypted1.png</p> <p>Введите пароль для шифрования:</p> <p>crypt4</p> <p>Выберете алгоритм генерации хэша:</p> <p><1> SHA-256</p> <p><2> MD5</p> <p><3> SHA-512</p> <p>1</p> <p>Шифрование прошло успешно!</p>	Шифрование файла целиком.	Файл открывается, но изображение не отображается.
2	<p>Выберете опцию:</p> <p><1> Зашифровать файл</p> <p><2> Дешифровать файл</p> <p><3> Зашифровать только пиксели файла</p> <p><4> Дешифровать только пиксели файла</p> <p><0> Выйти(</p> <p>2</p> <p>Введите название файла для дешифровки:</p> <p>Arthur_encrypted1.png</p> <p>Введите название файла для вывода:</p> <p>Arthur_decrypted1.png</p> <p>Введите пароль для дешифровки:</p> <p>crypt4</p> <p>Выберете алгоритм генерации хэша:</p> <p><1> SHA-256</p> <p><2> MD5</p> <p><3> SHA-512</p> <p>1</p> <p>Дешифровка прошла успешно!</p>	Дешифровка файла правильным паролем.	 <p>Все нормально.</p>

3	<p>Выберете опцию:</p> <p><1> Зашифровать файл</p> <p><2> Дешифровать файл</p> <p><3> Зашифровать только пиксели файла</p> <p><4> Дешифровать только пиксели файла</p> <p><0> Выйти(</p> <p>2</p> <p>Введите название файла для дешифровки:</p> <p>Arthur_encrypted1.png</p> <p>Введите название файла для вывода:</p> <p>Arthur_decrypted2.png</p> <p>Введите пароль для дешифровки:</p> <p>random_password</p> <p>Выберете алгоритм генерации хеша:</p> <p><1> SHA-256</p> <p><2> MD5</p> <p><3> SHA-512</p> <p>1</p> <p>Неверный пароль или алгоритм генерации хеша!</p>	<p>Дешифровка файла</p> <p>неправильным паролем.</p>	<p>Получаем соответствующее сообщение.</p>
4	<p>Выберете опцию:</p> <p><1> Зашифровать файл</p> <p><2> Дешифровать файл</p> <p><3> Зашифровать только пиксели файла</p> <p><4> Дешифровать только пиксели файла</p> <p><0> Выйти(</p> <p>2</p> <p>Введите название файла для дешифровки:</p> <p>Arthur_encrypted1.png</p> <p>Введите название файла для вывода:</p> <p>Arthur_decrypted3.png</p> <p>Введите пароль для дешифровки:</p> <p>crypt4</p> <p>Выберете алгоритм генерации хеша:</p> <p><1> SHA-256</p> <p><2> MD5</p> <p><3> SHA-512</p> <p>2</p> <p>Неверный пароль или алгоритм генерации хеша!</p>	<p>Дешифровка файла</p> <p>неправильным алгоритмом генерации хеша.</p>	<p>Получаем соответствующее сообщение.</p>

<p>5</p>	<p>Выберете опцию: <1> Зашифровать файл <2> Дешифровать файл <3> Зашифровать только пиксели файла <4> Дешифровать только пиксели файла <0> Выйти(3 Введите название файла для шифрования: Arthur.png Введите название файла для вывода: Arthur_encrypted2.png Введите пароль для шифрования: crypt4 Выберите алгоритм генерации хеша: <1> SHA-256 <2> MD5 <3> SHA-512 1 Шифрование прошло успешно!</p>	<p>Шифрование только пикселей изображения.</p>	 <p>Arthur_encrypted2.png Видим черты исходного изображения, так как использовали режим шифрования ECB.</p>
<p>6</p>	<p>Выберете опцию: <1> Зашифровать файл <2> Дешифровать файл <3> Зашифровать только пиксели файла <4> Дешифровать только пиксели файла <0> Выйти(4 Введите название файла для дешифровки: Arthur_encrypted2.png Введите название файла для вывода: Arthur_decrypted2.png Введите пароль для дешифровки: crypt4 Выберите алгоритм генерации хеша: <1> SHA-256 <2> MD5 <3> SHA-512 1 Дешифровка прошла успешно!</p>	<p>Расшифровка только пикселей изображения.</p>	 <p>Arthur_decrypted2.png Пиксели изображения успешно расшифровались.</p>

7	<p>Выберете опцию:</p> <p><1> Зашифровать файл</p> <p><2> Дешифровать файл</p> <p><3> Зашифровать только пиксели файла</p> <p><4> Дешифровать только пиксели файла</p> <p><0> Выйти(</p> <p>3</p> <p>Введите название файла для шифрования:</p> <p>Arthur.png</p> <p>Введите название файла для вывода:</p> <p>Arthur_encrypted3.png</p> <p>Введите пароль для шифрования:</p> <p>crypt4</p> <p>Выберете алгоритм генерации хеша:</p> <p><1> SHA-256</p> <p><2> MD5</p> <p><3> SHA-512</p> <p>3</p> <p>Шифрование прошло успешно!</p>	<p>Шифрование с использованием алгоритма генерации хеша длины >32 байт.</p>	 <p>Arthur_encrypted3.png</p>
8	<p>Выберете опцию:</p> <p><1> Зашифровать файл</p> <p><2> Дешифровать файл</p> <p><3> Зашифровать только пиксели файла</p> <p><4> Дешифровать только пиксели файла</p> <p><0> Выйти(</p> <p>4</p> <p>Введите название файла для дешифровки:</p> <p>Arthur_encrypted3.png</p> <p>Введите название файла для вывода:</p> <p>Arthur_decrypted3.png</p> <p>Введите пароль для дешифровки:</p> <p>crypt4</p> <p>Выберете алгоритм генерации хеша:</p> <p><1> SHA-256</p> <p><2> MD5</p> <p><3> SHA-512</p> <p>3</p> <p>Дешифровка прошла успешно!</p>	<p>Дешифрование с использованием алгоритма генерации хеша длины >32 байт.</p>	 <p>Arthur_decrypted3.png</p>

5. Данные

№	Содержание	Назначение
1		<p>Исходное изображение Arthur.png</p>
2		<p>Изображение Arthur_encrypted2, с шифрованием только пикселей.</p>

3		Изображение Arthur_decrypted2, успешно расшифрованное.
4	crypt4	Пароль