Название:	Независимый код
Категория:	Реверс-инжиниринг
Уровень:	Легкий
Очки:	280
Описание:	Данный код независимый. Это, конечно, хорошо, но что в нём?
Теги:	C#, .NET, IL-код
Автор:	ROP

Прохождение:

Изучаем данный код.

```
  /* 0x0000025C 00          */ IL_0000: nop
  /* 0x0000025D 281000000A  */ IL_0001: call      class [mscorlib]System.Text.Encoding [mscorlib]System.Text.Encoding::get_UTF8()
  /* 0x00000262 7201000070  */ IL_0006: ldstr     "qBFecRe#K2yff345"
  /* 0x00000267 6F1100000A  */ IL_000B: callvirt  instance uint8[] [mscorlib]System.Text.Encoding::GetBytes(string)
  /* 0x0000026C 0A          */ IL_0010: stloc.0
  /* 0x0000026D 281000000A  */ IL_0011: call      class [mscorlib]System.Text.Encoding [mscorlib]System.Text.Encoding::get_UTF8()
  /* 0x00000272 7223000070  */ IL_0016: ldstr     "ILc0dE_t@sK23###"
  /* 0x00000277 6F1100000A  */ IL_001B: callvirt  instance uint8[] [mscorlib]System.Text.Encoding::GetBytes(string)
  /* 0x0000027C 0B          */ IL_0020: stloc.1
  /* 0x0000027D 7245000070  */ IL_0021: ldstr     "Введите флаг: "
  /* 0x00000282 281200000A  */ IL_0026: call      void [mscorlib]System.Console::Write(string)
  /* 0x00000287 00          */ IL_002B: nop
  /* 0x00000288 281300000A  */ IL_002C: call      string [mscorlib]System.Console::ReadLine()
  /* 0x0000028D 0C          */ IL_0031: stloc.2
  /* 0x0000028E 08          */ IL_0032: ldloc.2
  /* 0x0000028F 06          */ IL_0033: ldloc.0
  /* 0x00000290 07          */ IL_0034: ldloc.1
  /* 0x00000291 2802000006  */ IL_0035: call      uint8[] Program::EncryptStringToBytes_Aes(string, uint8[], uint8[])
  /* 0x00000296 0D          */ IL_003A: stloc.3
  /* 0x00000297 1F20        */ IL_003B: ldc.i4.s  32
  /* 0x00000299 8D1A000001  */ IL_003D: newarr    [mscorlib]System.Byte
  /* 0x0000029E 25          */ IL_0042: dup
  /* 0x0000029F D001000004  */ IL_0043: ldtoken   field valuetype '<PrivateImplementationDetails>'/'__StaticArrayInitTypeSize=32' '<PrivateImplementati
  /* 0x000002A4 281400000A  */ IL_0048: call      void [mscorlib]System.Runtime.CompilerServices.RuntimeHelpers::InitializeArray(class [mscorlib]System
  /* 0x000002A9 1304        */ IL_004D: stloc.s   V_4
  /* 0x000002AB 09          */ IL_004F: ldloc.3
  /* 0x000002AC 1104        */ IL_0050: ldloc.s   V_4
  /* 0x000002AE 2803000006  */ IL_0052: call      bool Program::ByteArraysAreEqual(uint8[], uint8[])
  /* 0x000002B3 1305        */ IL_0057: stloc.s   V_5
  /* 0x000002B5 1105        */ IL_0059: ldloc.s   V_5
  /* 0x000002B7 2C0F        */ IL_005B: brfalse.s IL_006C

  /* 0x000002B9 00          */ IL_005D: nop
  /* 0x000002BA 7263000070  */ IL_005E: ldstr     "Успех, это он!"
  /* 0x000002BF 281500000A  */ IL_0063: call      void [mscorlib]System.Console::WriteLine(string)
  /* 0x000002C4 00          */ IL_0068: nop
  /* 0x000002C5 00          */ IL_0069: nop
```

Это IL-код от C#. Изучив его, можно восстановить код на C#.

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Text;

class Program
{
    static void Main()
    {
        byte[] key = Encoding.UTF8.GetBytes("qBFecRe#K2yff34
5");
        byte[] IV = Encoding.UTF8.GetBytes("ILc0dE_t@sK23##
#");

        Console.Write("Введите флаг: ");
        string flag = Console.ReadLine();

        byte[] encryptedFlag = EncryptStringToBytes_Aes(flag,
key, IV);

        byte[] targetBytes = { 0xB3, 0xCC, 0x93, 0xE5, 0x73,
0x94, 0xD2, 0x93, 0x43, 0xDE, 0x3F, 0x8B, 0x43, 0xFF, 0xC1, 0
x4D, 0x79, 0x1E, 0x99, 0xF2, 0x70, 0xE2, 0xCB, 0xA0, 0x8C, 0x
19, 0x9C, 0x5B, 0x92, 0xED, 0x3A, 0x62 };

        if (ByteArraysAreEqual(encryptedFlag, targetBytes))
        {
            Console.WriteLine("Успех, это он!");
        }
        else
        {
            Console.WriteLine("Не тот флаг :(");
        }
        Console.ReadLine();
    }
```

```
    static byte[] EncryptStringToBytes_Aes(string plainText,
byte[] Key, byte[] IV)
    {
        if (plainText == null || plainText.Length <= 0)
            throw new ArgumentNullException("plainText");
        if (Key == null || Key.Length <= 0)
            throw new ArgumentNullException("Key");
        if (IV == null || IV.Length <= 0)
            throw new ArgumentNullException("IV");

        byte[] encrypted;

        using (Aes aesAlg = Aes.Create())
        {
            aesAlg.Key = Key;
            aesAlg.IV = IV;

            ICryptoTransform encryptor = aesAlg.CreateEncrypt
or(aesAlg.Key, aesAlg.IV);

            using (MemoryStream msEncrypt = new MemoryStream
())
            {
                using (CryptoStream csEncrypt = new CryptoStr
eam(msEncrypt, encryptor, CryptoStreamMode.Write))
                {
                    using (StreamWriter swEncrypt = new Strea
mWriter(csEncrypt))
                    {
                        swEncrypt.Write(plainText);
                    }
                }
                encrypted = msEncrypt.ToArray();
            }
        }
```

```
        return encrypted;
    }

    static bool ByteArraysAreEqual(byte[] array1, byte[] arra
y2)
    {
        if (array1.Length != array2.Length)
            return false;

        for (int i = 0; i < array1.Length; i++)
        {
            if (array1[i] != array2[i])
                return false;
        }

        return true;
    }
}
```

Пользователь вводит флаг, далее он шифруется AES'ом и затем сравнивается с:

```
0xB3, 0xCC, 0x93, 0xE5, 0x73, 0x94, 0xD2, 0x93, 0x43, 0xDE, 0
x3F, 0x8B, 0x43, 0xFF, 0xC1, 0x4D, 0x79, 0x1E, 0x99, 0xF2, 0x
70, 0xE2, 0xCB, 0xA0, 0x8C, 0x19, 0x9C, 0x5B, 0x92, 0xED, 0x3
A, 0x62
```

Если наши зашифрованные байты равны этим, то выводится сообщение об успехе. Значит, можно расшифровать эти байты через AES.

```
    byte[] key = Encoding.UTF8.GetBytes("qBFecRe#K2yff345");
    byte[] IV = Encoding.UTF8.GetBytes("ILc0dE_t@sK23###");
```

Через CyberChef можно расшифровать эти байты.