Data Encryption algorithms in C# language, based on bitwise “XOR” operator

M. Kulev, S. Persianov
Technical University of Moldova
mkmk@mail.md, persianov.s@gmail.com

***Abstract* — Nowadays the Internet has a huge importance in people’s lives. It offers a lot of possibilities, such as: e-commerce, mailing, chatting, data transfer, etc. All this methods of communication are not secured, because Ethernet Networks liable to having sniffers and someone can read or stole your confidential information. So there appears necessity to protect the information which flows through the NET, called data encryption. Today, there are a big amount of encryption algorithms, like DES, Triple-DES or Blowfish, but how older is the algorithm the higher is probability to find some bugs in it (from Cryptanalysis point of view).**

***Index Terms* — Bitwise operator, Cryptography, key, XOR.**

I. Introduction

Cryptography is a practice of hiding information, transforming it into an unreadable format. The sender retained the ability to decrypt the data, using the secret key-word. So it allows us to transfer different data (information about transactions and credit cards, for example) through unsecured networks.

There are two types of Cryptographic Systems[1]:

- Restricted use systems – must keep nature of encoding and decoding secret;
- General use systems – nature of encoding and decoding is generally known (but it uses a key to help safeguard system).

General use systems are divided in two groups:

- Secret-key systems – most traditional systems (same key for encoding and decoding);
- Public-key systems – public key provided for encoding and a private key used for decoding.

A key is a small amount of information needed to use cryptographic system. It determines the functional output of the algorithm or cipher. Without a correct key, the algorithm will produce no useful result.

II. Software Description

The “BitProtect Crypto-Service” is a small application designed to encrypt different data. It is a General use system with a secret-key (same key for encoding and decoding). It was developed in C# .NET and has a nice and friendly GUI (Graphical User Interface)[Appendix A].

This application offers you two encryption services:

- String Encoding (plaintext);
- Files Encryption.

Each service has three encryption algorithms of different complexity and level of security.

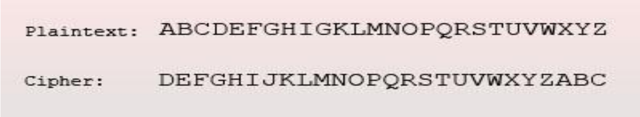
Another feature is the possibility to send the encrypted text or file to someone, using SMTP protocol. You should only fill the fields with receiver mail address and subject of email.

III. Text Cipher

Text Cipher is designed for texts encoding, using a key word. It has 3 algorithms:

- Caesar’s cipher algorithm;
- Simple XOR algorithm;
- Medium XOR algorithm;

Caesar’s cipher, is one of the simplest and most known encryption techniques. It is a type of substitution cipher, in which each letter in plaintext is replaces by another one from the same position in replaced alphabet (shifting depends on key-word).

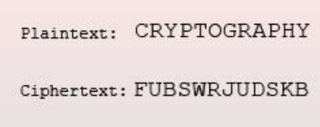


```
Plaintext:  ABCDEFGHIGKLMNOPQRSTUVWXYZ

Cipher:     DEFGHIJKLMNOPQRSTUVWXYZABC
```

Fig. 1

If we want to encrypt the word “CRYPTOGRAPHY” the cipher with the previous table will give the following result:



```
Plaintext:  CRYPTOGRAPHY

Ciphertext: FUBSWRJUDSKB
```

Fig. 2

Also, using modular arithmetic the encryption can be represented, by first transforming the letters to numbers, according to the scheme, A = 0, B = 1, … , Z = 25. So the encryption of letter x by shifting m can be described mathematically as:

$$E_n(x) = (x + n) \mod 26;$$
$$D_n(x) = (x - n) \mod 26.$$

Medium XOR algorithm. This algorithm is stronger against attacks than the Caesar’s cipher. It is based on bitwise operator XOR.

It is called Exclusive Or. This operator takes two bit patterns of equal length and performs the logical XOR operation on each pair of corresponding bits.[Appendix B]

The algorithm transforms the input data into an array of bytes, and encrypt each byte of array with each byte of key-word. Graphically it looks like:

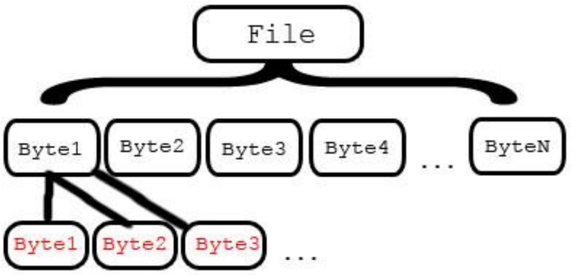


Fig. 3

XOR Blocks Algorithm. Like the previous algorithm, thisone is also based on XOR operator. The main idea is dividing the file byte into blocks. The number of blocks is equal to number of words in key-phrase.

This algorithm is stronger against attacks.

IV. Advantages

- You can easily hide information and keep it in safety on your hard drive, without uploading it on different hosting services;
- If your personal files were stolen, that person won't be able to read and to use them;
- The algorithms which were implemented in BitProtect Crypto-Service, can be used in applications like LAN/Internet Chats, to encrypt and decrypt messages and files which are transferring;
- You can easily send encrypted texts/files to any mail you want;
- The software encrypt all files of different sizes and different extensions.

V. Disadvantages

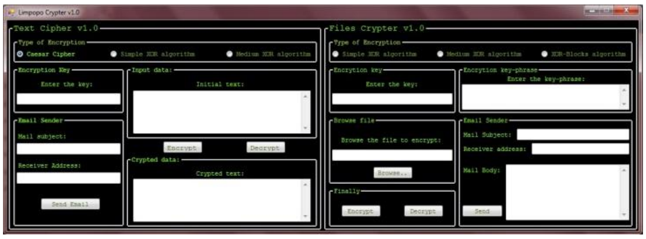
- There are more powerful data encryption algorithms (DES, Triple – DES, Blowfish, CAST, etc.)[2];
- The BitProtect Crypto-Service algorithms weren't analyzed using Cryptanalysis.

VI. Conclusions

In this paper I presented a solution for keeping our files in safety. I used bitwise operator XOR, which allows me to implement different algorithms of different complexity.

This program will save a lot of time, if user wants to send the encrypted data to someone's mail box (using SMTP). Files which were encrypted, are saved with .crypted extension, and have the same size as the initial files. This application is cross-platform, because it was written in C# language and it works on Microsoft Operating Systems and Linux Distributions, with Project Mono installed.

APPENDIX A

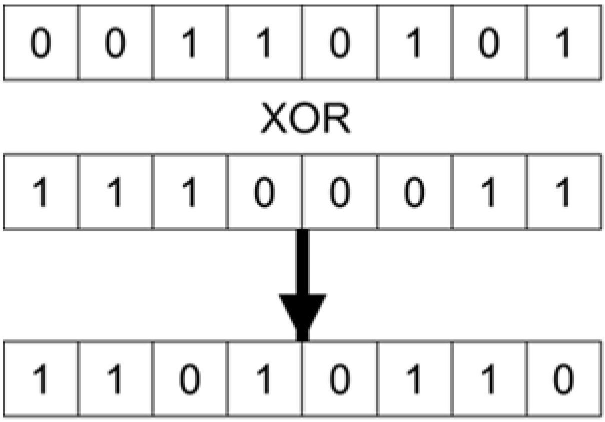


Type of Encryption
● Caesar Cipher ○ Simple XOR algorithm ○ Medium XOR algorithm

Type of Encryption
○ Simple XOR algorithm ○ Medium XOR algorithm ○ XOR-Blocks algorithm

BitProtect Crypto-Service. GUI

APPENDIX B



0 0 1 1 0 1 0 1

XOR

1 1 1 0 0 0 1 1

1 1 0 1 0 1 1 0

REFERENCES

[1] University of Maine. Department of Computer Science. http://laptops.maine.edu/crypto/sld031.htm

[2] Kremlin Encrypt. http://www.kremlinencrypt.com/algorithms.htm