|  |  |
| --- | --- |
| Министерство образования Республики Беларусь  Учреждение образования «Полоцкий государственный университет» | |
|  | Факультет информационных технологий  Кафедра технологий программирования |
| Лабораторная работа №4 по курсу «Теория информации»  «Криптоанализ методов простой подстановки» | |
| Выполнил | Студент гр. 21-ИТ-1  Катушёнок И.В. |
| Проверила | Васильева Д.М. |
| Полоцк, 2023г. | |

**Ход работы**

Простейшие шифры подстановки (substitution) реализуют замену каждого символа исходного текста на один из символов алфавита шифротекста. В общем случае подстановочный шифр описывается таблицей подстановки, состоящей из двух строк и n столбцов. Количество столбцов таблицы подстановки соответствует количеству различных символов в алфавите исходного текста. Верхняя строка таблицы подстановки содержит все возможные символы исходного текста, а нижняя – соответствующие им символы шифротекста.

**Задание:**

1. Реализовать программное средство, осуществляющее шифрование и дешифрование текстового файла, содержащего текст на заданном языке (нечетный номер в списке – английский язык, четный номер – русский).

2. Реализовать программное средство, осуществляющее криптоанализ зашифрованного по методу Виженера текста. Для криптоанализа использовать тест Касиски.

3. Провести экспериментальное исследование зависимости вероятности успешного проведения атаки по методу Касиски от длины шифротекста.

4. Провести экспериментальное исследование зависимости вероятности успешного проведения атаки по методу Касиски от длины использованного при шифровании ключевого слова.

Листинг 1 - реализация программы:

internal class Vijener

{

public static string VijenerEncrypt(string inputString, string keyWord)

{

inputString = inputString.ToLower();

keyWord = keyWord.ToLower();

string encryptedString = "";

for (int i = 0; i < inputString.Length; i++)

{

if (inputString[i] == ' ')

{

encryptedString += ' ';

continue;

}

int encryptingChar = (char)(inputString[i] + (keyWord[i % keyWord.Length] - 'a'));

if (encryptingChar > 122) encryptingChar = (encryptingChar % 122) + 96;

encryptedString += (char)encryptingChar;

}

return encryptedString;

}

public static string VijenarDecrypt(string inputString, string keyWord)

{

inputString = inputString.ToLower();

keyWord = keyWord.ToLower();

string decryptedString = "";

for (int i = 0; i < inputString.Length; i++)

{

if (inputString[i] == ' ')

{

decryptedString += ' ';

continue;

}

int decryptingChar = (char)(inputString[i] - (keyWord[i % keyWord.Length] - 'a'));

if (decryptingChar < 97) decryptingChar = decryptingChar + 26;

decryptedString += (char)decryptingChar;

}

return decryptedString;

}

}

class Attack

{

string alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

public static Dictionary<string, List<int>> RepeatedBlocks(string input)

{

var repeatedMap = new Dictionary<string, List<int>>();

for (var i = 0; i < input.Length - 3; i++)

{

string block = input.Substring(i, 3);

if (!repeatedMap.ContainsKey(block))

repeatedMap[block] = new List<int>();

repeatedMap[block].Add(i);

}

return repeatedMap;

}

public static List<int> CalcKeyLengths(Dictionary<string, List<int>> repeatedMap)

{

var keyLengths = new List<int>();

foreach (var map in repeatedMap)

{

var positions = map.Value;

for (var i = 0; i < positions.Count - 1; i++)

{

for (var j = i + 1; j < positions.Count; j++)

{

var distance = positions[j] - positions[i];

if (distance > 0)

keyLengths.Add(distance);

}

}

}

return keyLengths;

}

}

class Program

{

public static void Main()

{

Vijener vijener = new Vijener();

string words = "cryptography and data security",

key = "Mouse";

var cryptedText = Vijener.VijenerEncrypt(words, key);

Console.WriteLine(cryptedText);

string encr = Vijener.VijenerEncrypt(words,key);

Console.WriteLine(Vijener.VijenarDecrypt(encr,key));

var lengths = Attack.CalcKeyLengths(Attack.RepeatedBlocks(cryptedText));

foreach (var length in lengths)

Console.WriteLine($"Possible key length: {length}");

}

}

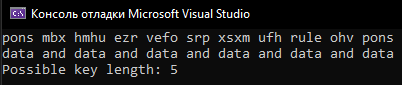


Рисунок 1. Результат работы программы