Отчёт по лабораторной работе №8

Элементы криптографии. Шифрование (кодирование) различных исходных текстов одним ключом

Ле Тиен Винь

Содержание

[I.Цель работы 1](#_Toc180866375)

[II. Задание 1](#_Toc180866376)

[III. Выполнение задания 1](#_Toc180866377)

[Код приложения 1](#_Toc180866378)

[Анализ кода 2](#_Toc180866379)

[Результат программы 3](#_Toc180866380)

[IV. Вывод 3](#_Toc180866381)

# I.Цель работы

Освоить на практике применение режима одноключевого кодирования на примере кодирования различных исходных текстов одним ключом.

# II. Задание

Два текста кодируются одним ключом (однократное гаммирование). Требуется не зная ключа и не стремясь его определить, прочитать оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты P1 и P2 в режиме однократного гаммирования. Приложение должно определить вид шифротекстов C1 и C2 обоих текстов P1 и P2 при известном ключе ; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить.

# III. Выполнение задания

## Код приложения

#include <iostream>  
#include <string>  
using namespace std;  
  
string xorOperator(const string &input, const string &key) {  
 string output = input;  
 for (size\_t i = 0; i < input.size(); ++i) {  
 output[i] = input[i] ^ key[i % key.size()];  
 }  
 return output;  
}  
  
int main() {  
 string P1 = "ThisIsSecret";  
 string P2 = "DontTellThat";  
 string key = "123456789123";  
   
 string ciphertext1 = xorOperator(P1, key);  
 string ciphertext2 = xorOperator(P2, key);  
  
 cout << "Ciphertext 1: " << ciphertext1 << endl;  
 cout << "Ciphertext 2: " << ciphertext2 << endl;  
  
 string Text1 = xorOperator(xorOperator(ciphertext1,ciphertext2),P1);  
 string Text2 = xorOperator(xorOperator(ciphertext1,ciphertext2),P2);  
  
 cout << "Text 1: " << Text1 << endl;  
 cout << "Text 2: " << Text2 << endl;  
  
 return 0;  
}

## Анализ кода

* Мы используем метод шифрования: Выполнение операции сложения по модулю 2 (XOR).

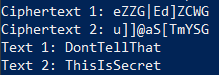
string xorOperator(const string &input, const string &key) {  
 string output = input;  
 for (size\_t i = 0; i < input.size(); ++i) {  
 output[i] = input[i] ^ key[i % key.size()];  
 }  
 return output;  
}

* Функция преобразует каждый элемент введенного текста в новый элемент, зашифрованный на основе ключа, с помощью операцией сложения по модулю 2 (XOR): Ci = Pi + Ki (1).
* Где Ci — i-й символ получившегося зашифрованного послания, Pi — i-й символ открытого текста, Ki — i-й символ ключа, i = 1, …, m.

int main() {  
 string P1 = "ThisIsSecret";  
 string P2 = "DontTellThat";  
 string key = "123456789123";  
   
 string ciphertext1 = xorOperator(P1, key);  
 string ciphertext2 = xorOperator(P2, key);  
  
 cout << "Ciphertext 1: " << ciphertext1 << endl;  
 cout << "Ciphertext 2: " << ciphertext2 << endl;  
  
 string Text1 = xorOperator(xorOperator(ciphertext1,ciphertext2),P1);  
 string Text2 = xorOperator(xorOperator(ciphertext1,ciphertext2),P2);  
  
 cout << "Text 1: " << Text1 << endl;  
 cout << "Text 2: " << Text2 << endl;  
  
 return 0;  
}

* В main мы определем 2 исходного текста с называниями P1 и P2 и ключ key.
* Используовать функцию “xorOperator” для генерации зашифрованного текста и вывода зашифрованного текста на экран.
* В ситуации, когда злоумышленник знал один из двух текста, он может прочитать остальный, не зная ключа и не стремясь его определить, на основе свойства операции XOR: 1 + 1 = 0, 1 + 0 = 1.
* Получаем С1 + С2 = P1 + K + P2 + K = P1 + P2, следует C1 + C2 + P1 = P1 + P2 + P1 = P2.

## Результат программы



# IV. Вывод

После лаборатоной работы я получил практические навыки по применению режима одноключевого кодирования на примере кодирования различных исходных текстов одним ключом.