Отчет по лаборатной работе №8

по предмету Информационная безопасность

Алхимова Дарья Сергеевна

Содержание

[Цель работы 1](#__RefHeading___Toc561_1177348333)

[Задание 1](#__RefHeading___Toc563_1177348333)

[Теоретическое введение 1](#__RefHeading___Toc565_1177348333)

[Выполнение лабораторной работы 3](#__RefHeading___Toc567_1177348333)

[Выводы 7](#__RefHeading___Toc569_1177348333)

[Список литературы 7](#__RefHeading___Toc571_1177348333)

# Цель работы

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

# Задание

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

# Теоретическое введение

Гаммирование представляет собой наложение (снятие) на открытые (зашифрованные) данные последовательности элементов других данных, полученной с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. Иными словами, наложение гаммы — это сложение её элементов с элементами открытого (закрытого) текста по некоторому фиксированному модулю, значение которого представляет собой известную часть алгоритма шифрования.

Наложение гаммы по сути представляет собой выполнение операции сложения по модулю 2 (XOR) (обозначаемая знаком +) между элементами гаммы и элементами подлежащего сокрытию текста. Напомним, как работает операция XOR над битами: 0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, 1 + 1 = 0. Такой метод шифрования является симметричным, так как двойное прибавление одной и той же величины по модулю 2 восстанавливает исходное значение, а шифрование и расшифрование выполняется одной и той же программой.

Открытый текст имеет символьный вид, а ключ — шестнадцатеричное представление. Ключ также можно представить в символьном виде, вос- пользовавшись таблицей ASCII-кодов.

Необходимые и достаточные условия абсолютной стойкости шифра:  
+ полная случайность ключа;  
+ равенство длин ключа и открытого текста;  
+ однократное использование ключа.

Метод гаммирования становится бессильным, если известен фрагмент исход- ного текста и соответствующая ему шифрограмма. В этом случае простым вычи- танием по модулю 2 получается отрезок псевдослучайной последовательности и по нему восстанавливается вся эта последовательность.

Алгоритм гаммирования:  
1. Генерация сегмента гаммы H(1) и наложение его на соответствующий участок шифруемых данных.  
2. Подсчет контрольной суммы участка, соответствующего сегменту гаммы H(1).  
3. Генерация с учетом контрольной суммы уже зашифрованного участка данных следующего сегмента гамм H(2).  
4. Подсчет контрольной суммы участка данных, соответствующего сегменту данных H(2) и т.д.

Шифротексты обеих телеграмм можно получить по формулам режима однократного гаммирования:

Открытый текст можно найти, зная шифротекст двух телеграмм, зашифрованных одним ключом. Для это оба равенства складываются по модулю 2. Тогда с учётом свойства операции XOR получаем:

Предположим, что одна из телеграмм является шаблоном — т.е. имеет текст фиксированный формат, в который вписываются значения полей. Допустим, что злоумышленнику этот формат известен. Тогда он получает достаточно много пар (известен вид обеих шифровок). Тогда зная имеем:

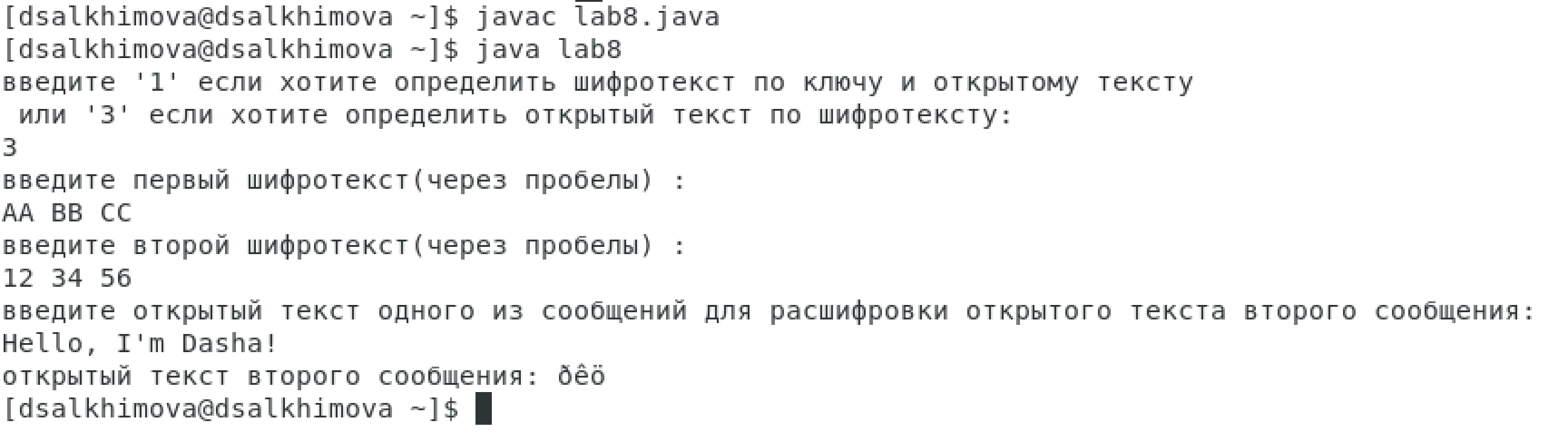
Таким образом, злоумышленник получает возможность определить те символы сообщения , которые находятся на позициях известного шаблона сообщения . В соответствии с логикой сообщения , злоумышленник имеет реальный шанс узнать ещё некоторое количество символов сообщения . Затем вновь используется равенство с подстановкой вместо полученных на предыдущем шаге новых символов сообщения . И так далее. Действуя подобным образом, злоумышленник даже если не прочитает оба сообщения, то значительно уменьшит пространство их поиска.

# Выполнение лабораторной работы

1. Создала файл программы lab8.java и открыла его для заполнения кодом. Модифицировала код программы предыдущей лабораторной работы.

* Полный текст программы:
* import java.util.HashMap;  
   import java.util.Iterator;  
   import java.util.Map;  
   import java.util.Scanner;  
    
   public class lab8 {  
   public static void main(String [] args) {  
    
   HashMap<Character, String> map = new HashMap<Character ,String>();  
   map.put('0', "0000");  
   map.put('1',"0001");  
   map.put('2',"0010");  
   map.put('3', "0011");  
   map.put('4', "0100");  
   map.put('5',"0101");  
   map.put('6',"0110");  
   map.put('7',"0111");  
   map.put('8',"1000");  
   map.put('9', "1001");  
   map.put('A', "1010");  
   map.put('B',"1011" );  
   map.put('C', "1100");  
   map.put('D', "1101");  
   map.put('E',"1110" );  
   map.put('F', "1111");  
    
   String text="";  
   String cipher;  
   String cipher2;  
   Scanner in = new Scanner(System.in);  
    
   System.out.println("введите '1' если хотите определить шифротекст по ключу и открытому тексту \n или '3' если хотите определить открытый текст по шифротексту: ");  
   int input = in.nextInt();  
    
   if(input==1) {  
    
   Scanner in2 = new Scanner(System.in);  
   System.out.println("введите ключ шифрования: ");  
   cipher= in2.nextLine();  
   System.out.println("введите открытый текст: ");  
   cipher2 = in2.nextLine();  
   cipher2= characterto16(cipher2,map);  
   String shifr = shifrovanie(cipher,cipher2,map);  
   System.out.println("шифротекст : "+shifr);  
    
   }else {  
    
   Scanner in2 = new Scanner(System.in);  
   System.out.println("введите первый шифротекст(через пробелы) : ");  
   cipher= in2.nextLine();  
   System.out.println("введите второй шифротекст(через пробелы) : ");  
   cipher2= in2.nextLine();  
   System.out.println("введите открытый текст одного из сообщений для расшифровки открытого текста второго сообщения:");  
   text =in2.nextLine();  
   String C1xorC2= shifrovanie(cipher,cipher2,map);  
   String cipher16=characterto16(text,map);  
   String result = shifrovanie(C1xorC2,cipher16,map);  
   System.out.println("открытый текст второго сообщения: "+tocharacter(result,map));  
   }  
    
   }  
    
   public static String characterto16 (String cipher,HashMap<Character, String> map) {  
    
   char[] chararray = cipher.toCharArray();  
   String finalcode="";  
   for(int i=0;i<chararray.length;i++) {  
   char character = chararray[i];  
   int ascii = (int) character;  
   String code = Integer.toString(ascii,2);  
   String curcode=code;  
    
   for(int j=0;j<8-code.length();j++) {  
   curcode="0"+curcode;  
   }  
    
   code = curcode;  
   String val = code.substring(0, 4);  
   String val2 = code.substring(4);  
   char nval = ' ';  
   char nval2 = ' ';  
   Iterator it = map.entrySet().iterator();  
    
   while (it.hasNext()) {  
   Map.Entry pair = (Map.Entry)it.next();  
    
   if(pair.getValue().equals(val)) {  
   nval=(char)pair.getKey();  
   }  
    
   if(pair.getValue().equals(val2)) {  
   nval2=(char)pair.getKey();  
   }  
    
   }  
    
   String v = String.valueOf(nval)+String.valueOf(nval2);  
   finalcode=finalcode+v+" ";  
   }  
    
   return finalcode;  
   }  
    
   public static String tocharacter(String cipher, HashMap<Character, String> map) {  
   String[] splt = cipher.split("\\s+");  
   String finalcode="";  
    
   for(int i=0;i<splt.length;i++) {  
   char[] symbols = splt[i].toCharArray();  
   String symbol = map.get(symbols[0])+map.get(symbols[1]);  
   int number = Integer.parseInt(symbol, 2);  
   finalcode+=Character.toString ((char) number);  
   }  
    
   return finalcode;  
   }  
    
   public static String shifrovanie(String cipher, String cipher2,HashMap<Character, String> map) {  
    
   String[] splt = cipher.split("\\s+");  
   String[] splt2 = cipher2.split("\\s+");  
   String finalcode="";  
    
   for(int i=0;i<splt.length;i++) {  
   char[] symbols = splt[i].toCharArray();  
   String symbol = map.get(symbols[0])+map.get(symbols[1]);  
   char[] symbols2 = splt2[i].toCharArray();  
   String symbol2 = map.get(symbols2[0])+map.get(symbols2[1]);  
   String newsymbol="";  
    
   for(int j=0;j<symbol2.length();j++) {  
   int number= Character.digit(symbol2.charAt(j), 10);  
   int number2 = Character.digit(symbol.charAt(j), 10);  
   newsymbol+=number^number2;  
   }  
    
   String val = newsymbol.substring(0, 4);  
   String val2= newsymbol.substring(4);  
   char nval=' ';  
   char nval2=' ';  
   Iterator it = map.entrySet().iterator();  
    
   while (it.hasNext()) {  
   Map.Entry pair = (Map.Entry)it.next();  
    
   if(pair.getValue().equals(val)) {  
   nval=(char)pair.getKey();  
   }  
    
   if(pair.getValue().equals(val2)) {  
   nval2=(char)pair.getKey();  
   }  
    
   }  
    
   String v = String.valueOf(nval)+String.valueOf(nval2);  
   finalcode=finalcode+v+" ";  
   }  
    
   return finalcode;  
   }  
   }

1. Скомпиллировала созданный программный файл и запустила. Проверила коорректность работы кода: при вводе двух шифротекстов и открытого сообщения отдного сообщения программа вычисляет текст второго сообщения.([рис. 1](images8/1.png)).

* 
* Создание файла программы

# Выводы

В процессе выполнения данной лабораторной работы я приобрела навыки применения режима однократного гаммирования.

# Список литературы

1. Описание лабораторной работы 8 - URL: <https://esystem.rudn.ru/pluginfile.php/1652177/mod_resource/content/2/008-lab_crypto-key.pdf>
2. Шифрование методом гаммирования - URL: <http://altaev-aa.narod.ru/security/XOR.html>
3. Режим гаммирования в блочном алгоритме шифрования <https://kabinfo.ucoz.ru/index/shifr_reshetka_kardano/0-374>