

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук
Кафедра прикладной информатики и теории вероятностей

Лабораторная работа №3 Шифрование гаммированием

*Дисциплина: Математические основы защиты
информации и информационной безопасности*

Царитова Нина Аведиковна

Группа: НФИмд-02-23

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Шифрование гаммированием	9
5	Выводы	11
	Список литературы	12

List of Figures

4.1	Гаммирование конечной гаммой	9
4.2	Гаммирование конечной гаммой	10
4.3	Результат гаммирования конечной гаммой	10

List of Tables

1 Цель работы

Целью данной лабораторной работы является ознакомление с шифрованием гаммированием, – а так же реализация шифрования гаммирования конечной гаммой.

2 Задание

Реализовать алгоритм шифрования гаммированием конечной гаммой.

3 Теоретическое введение

Гаммирование, или Шифр XOR — метод симметричного шифрования, заключающийся в «наложении» последовательности, состоящей из случайных чисел, на открытый текст. Последовательность случайных чисел называется гамма-последовательностью и используется для зашифровывания и расшифровывания данных. Суммирование обычно выполняется в каком-либо конечном поле. Например, в поле Галуа суммирование принимает вид операции «исключающее ИЛИ (XOR)»[wiki:xor?].

В криптографии простой шифр XOR является разновидностью аддитивного шифра, алгоритма шифрования, который работает в соответствии с принципами [wiki:xorEN?]:

Шифрование гаммированием

где \oplus обозначает операцию исключающей дизъюнкции (XOR). Эта операция иногда называется сложением по модулю 2 (или вычитанием, что идентично). С помощью данной логики строка текста может быть зашифрована путем применения побитового оператора XOR к каждому символу с использованием заданного ключа. Для расшифровки результата достаточно повторно применить функцию XOR с ключом, чтобы снять шифр [wiki:xorEN?].

Шифры гаммирования являются самыми эффективными с точки зрения стойкости и скорости преобразований (процедур зашифрования и дешифрования). По стойкости данные шифры относятся к классу совершенных. Для зашифрования и дешифрования используются элементарные арифметические операции – открытое/зашифрованное сообщение и гамма, представленные в числовом виде,

складываются друг с другом по модулю (mod)[**anisivov:xor?**].

Пусть символам исходного алфавита соответствуют числа от 0 (А) до 32 (Я). Если обозначить число, соответствующее исходному символу, x , а символу ключа – k , то можно записать правило гаммирования следующим образом: $z = x + k(mod N)$, где z – закодированный символ, N - количество символов в алфавите, а сложение по модулю N - операция, аналогичная обычному сложению, с тем отличием, что если обычное суммирование дает результат, больший или равный N , то значением суммы считается остаток от деления его на N [**elec:xor?**].

4 Выполнение лабораторной работы

4.1 Шифрование гаммированием

В соответствии с заданием, была написана программа для шифрования гаммированием. Программный код представлен ниже.

```
alphabet="АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"#задаем алфавит
alphabet_list=list(alphabet)#сделали алфавит списком
N=len(alphabet)#ввели размер алфавита
slovo="ПРИКАЗ"
key="ГАММА"
index_slovo=[]#ввели списки для индексов
index_key=[]#ввели списки для индексов
...

Находим индексы в соответствии с алфавитом
...

for i1 in slovo:
    index_slovo.append(alphabet.find(i1))
for i2 in key:
    index_key.append(alphabet.find(i2))
...

Находим индексы в соответствии с алфавитом (+смещение на 1 (из-за питона))
...

index_slovo_1=[]
index_key_1=[]
for j1 in range (0,len(index_slovo)):
    index_slovo_1.append(index_slovo[j1]+1)

for j2 in range (0,len(index_key)):
    index_key_1.append(index_key[j2]+1)
```

Figure 4.1: Гаммирование конечной гаммой

```

'''
Нахождение индексов букв будущего шифра (первые k символов, где k-длина ключа)
'''

ciphered_text_indexes=[]#ввели список для индексов будущего шифра
for l in range(len(index_key_1)):
    ciphered_text_indexes.append(index_slovo_1[l]+(index_key_1[l])%N)
'''

Поиск новых индексов для шифра
'''

difference=len(index_slovo_1)-len(index_key_1)#ввели разницу в длине
index_key_2=0#ввели индекс символа ключа, с которого будем начинать
index_slovo_2=len(index_key_1)#ввели индекс символа слова, с которого будем начинать
while difference>0:
    ciphered_text_indexes.append(index_slovo_1[index_slovo_2]+(index_key_1[index_key_2])%N)
    difference=difference-1
    index_key_2+=1
    index_slovo_2=index_slovo_2-1
    if index_key_2==len(index_key_1):
        index_key_2=0
#ВНИМАНИЕ! ДЛЯ ТОГО, ЧТОБЫ СХОДИЛОСЬ С ОТВЕТОМ,
#ДАННЫМ В ЛАБОРАТОРНОЙ РАБОТЕ НЕОБХОДИМО ВЗЯТЬ АЛАВИТ БЕЗ БУКВЫ Ё (т.е. 32 символа)
'''

Поиск шифра с помощью полученных индексов и алфавита
'''

ciphered_text=[]
for i in range(len(ciphered_text_indexes)):
    ciphered_text.append(alphabet_list[ciphered_text_indexes[i]-1])#вспомнили что в питоне индексация с 1!
print(ciphered_text_indexes)
print('Криптограмма:',"".join(ciphered_text),'')

```

Figure 4.2: Гаммирование конечной гаммой

Результаты выполнения программы представлены ниже.



`[20, 18, 22, 24, 2, 12]`
 Криптограмма: " УСХЧБЛ "

Figure 4.3: Результат гаммирования конечной гаммой

5 Выводы

Таким образом, была достигнута цель, поставленная в начале лабораторной работы: я ознакомилась с шифрованием гаммированием, а так же мне удалось реализовать алгоритм шифрования конечной гаммой на языке программирования Python.

Список литературы

1. Википедия. <https://ru.m.wikipedia.org/wiki/Гаммирование>
2. <https://altaev-aa.narod.ru/security/XOR.html>