

Отчёт по лабораторной работе 3. Шифрование гаммированием

Ильин Никита Евгеньевич

Содержание

| | | |
|----------|---------------------------------------|-----------|
| 1 | Цель работы | 5 |
| 2 | Задание | 6 |
| 3 | Теоретическое введение | 7 |
| 4 | Выполнение лабораторной работы | 9 |
| 5 | Выводы | 11 |

Список иллюстраций

| | | |
|-----|---|----|
| 4.1 | Программная реализация шифра Цезаря | 9 |
| 4.2 | Результат работы программы | 10 |

Список таблиц

1 Цель работы

Цель данной работы – научиться программировать Шифрование гаммированием/

2 Задание

1. Реализовать алгоритм шифрование гаммированием

3 Теоретическое введение

Из всех схем шифрования простейшей и наиболее надежной является схема однократного использования (рис. 1). Формируется t -разрядная случайная двоичная последовательность - ключ шифра. Отправитель производит побитовое сложение по модулю два ($\text{mod } 2$) ключа $k = k_1 k_2 \dots k_i \dots k_t$ и t -разрядной двоичной последовательности $p = p_1 p_2 \dots p_i \dots p_t$, соответствующей посылаемому сообщению: $C_i = p_i \oplus k_i$; $i = 1, m$, где p_i - i -й бит исходного текста, k_i - i -й бит ключа, \oplus - операция побитового сложения (XOR), c_i - i -й бит получившейся криптограммы $C = C_1 C_2 \dots C_i \dots C_m$. Операция побитного сложения является обратимой, т.е. $(x \oplus y) \oplus y = x$, поэтому дешифрование осуществляется повторным применением операции XOR к криптограмме: Ключ k Исходная информация p XOR $P_i = c$, $0 \leq k, i = 1, m$. Ключ k Зашифрованная информация c XOR Рис. 1 Расшифрованная информация p

Основным недостатком такой схемы является равенство объема ключевой информации и суммарного объема передаваемых сообщений. Данный недостаток можно убрать, используя ключ в качестве «зародыша», порождающего 12-значную (или более) длинную ключевую последовательность. Представлена такая схема, которая и называется гаммированием. На рис. 2 Ключ k Исходная информация p Зашифрованная информация c Y Расшифрованная информация p F-1 Ключ k GG Гаммирование - процедура наложения при помощи некоторой функции F на исходный текст гаммы шифра, т.е. псевдослучайной последовательности (ПСП) с выходов генератора G . Псевдослучайная последовательность по своим статистическим свойствам неотличима от случайной последовательности, но является детерминированной, т.е. известен алгоритм ее формирования.

Чаще Обычно в качестве функции F берется операция поразрядного сложения по модулю два или по модулю N (N - число букв алфавита открытого текста). Простейший генератор • псевдослучайной последовательности можно представить рекуррентным соотношением: $V_i = a \cdot V_{i-1} + b \bmod (m)$, $i = 1, m$, где V_i - i -й член последовательности псевдослучайных чисел, a, Y_0, b - ключевые параметры. Такая последовательность состоит из целых чисел от 0 до $t-1$. Если элементы V_i и Y_j совпадут, то совпадут и последующие участки: $Y_{j+1} Y_{j+2}, V_{i+1} V_{i+2} = V_{j+2}$. Таким образом, ПСП является периодической. Знание периода гаммы существенно облегчает криптоанализ. Максимальная длина периода равна t . Для ее достижения необходимо удовлетворить следующим условиям: 1. b и t - взаимно простые числа; 2. $a - 1$ делится на любой простой делитель числа t ; 3. $a - 1$ кратно 4, если t кратно 4. 13

Стойкость шифров, основанных на процедуре гаммирования, зависит от характеристик гаммы - длины и равномерности распределения вероятностей появления знаков гаммы. При использовании генератора ПСП получаем бесконечную гамму. Однако, возможен режим шифрования конечной гаммы. В роли конечной гаммы может выступать фраза. Как и ранее, используется алфавитный порядок букв, т.е. буква «а» имеет порядковый номер 1, «б» - 2 и т.д. Например, зашифруем слово «ПРИКАЗ» («16 17 09 1 01 08») гаммой «ГАММА» («04 01 13 13 01»). Будем использовать операцию побитового сложения по модулю 3 ($\bmod 3$). Получаем: $C_1 = 16 + 4 (\bmod 3) = 20$, $2c = 71 + 1 (\bmod 3) = 81$, $C_3 = 9 + 13 (\bmod 3) = 22$, $C_4 = 11 + 13 (\bmod 3) = 24 = s$, $(1 + 1 (\bmod 3))^2 = c_6 = 8 + 4 (\bmod 3) = 12$. Криптограмма: «УСХЧБЛ» («20 18 22 24 02 12»).

4 Выполнение лабораторной работы

1. Для начала реализуется алгоритм шифра на языке Python (рис. 4.1).

```
def gamma_encrypt(message, gamma):  
  
    def encrypt(letters_pair):  
        idx = (letters_pair[0]+1)+(letters_pair[1]+1) % len(alph)  
        if idx > len(alph):  
            idx = idx - len(alph)  
        return idx - 1  
  
    alph = list(map(chr, range(ord('a'), ord('я')+1)))  
    message_clear = list(filter(lambda s: s.lower() in alph, message))  
    gamma_clear = list(filter(lambda s: s.lower() in alph, gamma))  
    message_ind=list(map(lambda s: alph.index(s.lower()),message_clear))  
    gamma_ind=list(map(lambda s: alph.index(s.lower()),gamma_clear))  
  
    for i in range(len(message_ind) - len(gamma_ind)):  
        gamma_ind.append(gamma_ind[i])  
  
    print(f'{message.upper()} -> {message_ind}\n{gamma.upper()} -> {gamma_ind}')  
    encrypted_ind = list(map(lambda s: encrypt(s), zip(message_ind, gamma_ind)))  
    print(f'Зашифрованное сообщение: {encrypted_ind}\n')  
    return ''.join(list(map(lambda s: alph[s], encrypted_ind))).upper()
```

Рис. 4.1: Программная реализация шифра Цезаря

2. Зашифрованное сообщение выглядит следующим образом (рис. 4.2).

```
message = 'приказ'
gamma = 'гамма'

gamma_encrypt(message, gamma)
✓ 0.0s

ПРИКАЗ -> [15, 16, 8, 10, 0, 7]
ГАММА -> [3, 0, 12, 12, 0, 3]
Зашифрованное сообщение: [19, 17, 21, 23, 1, 11]

'УСХЧБЛ'
```

Рис. 4.2: Результат работы программы

5 Выводы

В ходе работы был реализован алгоритм шифрования гаммированием.