

# **Отчет по лабораторной работе №8**

**по дисциплине: Информационная безопасность**

Ким Михаил Алексеевич

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
4.1	Создание программы . . . . .	7
<b>5</b>	<b>Анализ результатов</b>	<b>18</b>
<b>6</b>	<b>Выводы</b>	<b>19</b>
	<b>Список литературы</b>	<b>20</b>

## Список иллюстраций

4.1	Однократное гаммирование различных текстов одним ключом на С++. Листинг. 1 . . . . .	14
4.2	Однократное гаммирование различных текстов одним ключом на С++. Листинг. 2 . . . . .	15
4.3	Однократное гаммирование различных текстов одним ключом на С++. Листинг. 3 . . . . .	16
4.4	Компиляция и выполнение программы . . . . .	17

# 1 Цель работы

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

## 2 Задание

Два текста кодируются одним ключом (однократное гаммирование). Требуется не зная ключа и не стремясь его определить, прочесть оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты  $P_1$  и  $P_2$  в режиме однократного гаммирования. Приложение должно определить вид шифротекстов  $C_1$  и  $C_2$  обоих текстов  $P_1$  и  $P_2$  при известном ключе. Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочесть оба текста, не зная ключа и не стремясь его определить.

### 3 Теоретическое введение

- Терминал (или «Bash», сокращение от «Bourne-Again shell») — это программа, которая используется для взаимодействия с командной оболочкой. Терминал применяется для выполнения административных задач, например: установку пакетов, действия с файлами и управление пользователями. [1]
- 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)» [2].

## 4 Выполнение лабораторной работы

### 4.1 Создание программы

1. Напишем программу на C++, удовлетворяющую всем условиям задания. Программа будет содержать четыре вспомогательные функции и их перегрузки: вывод в консоль информации в 16-ричной системе счисления, вывод в консоль вектора строк, кодирование и декодирование информации, определение фрагмента текста по шаблону и шифротекстам. (рис. 4.1, 4.2, 4.3).

```
#include <cstdlib>
#include <iostream>
#include <vector>
#include <string>
```

```
using std::cin;
using std::cout;
using std::endl;
using std::string;
using std::vector;
```

```
void printHex(const vector<unsigned char>& vec)
```

```

{
    cout << std::hex << std::uppercase;

    for (size_t i = 0; i < vec.size(); i++)
    {
        cout << static_cast<short>(vec[i]) << ' ';
    }

    cout << std::dec << std::nouppercase << endl;
}

void printHex(const vector<vector<unsigned char>>& vec)
{
    cout << std::hex << std::uppercase;

    for (size_t i = 0; i < vec.size(); i++)
    {
        for (size_t j = 0; j < vec[i].size(); j++)
        {
            cout << static_cast<short>(vec[i][j]) << ' ';
        }
        cout << endl;
    }

    cout << std::dec << std::nouppercase;
}

void printStrings(const vector<string>& str)
{

```



```

    for (size_t i = 0; i < str.size(); i++)
    {
        cout << str[i] << endl;
    }
}

vector<vector<unsigned char>> cryptGamming(const vector<unsigned char>& key,
{
    size_t elemSize = inpTextVector[0].size();
    for (size_t i = 0; i < inpTextVector.size(); i++)
    {
        if (inpTextVector[i].size() != elemSize)
        {
            cout << "Input texts lengths are not the same!" << endl;
            return vector<vector<unsigned char>>();
        }
    }

    if (key.size() != elemSize)
    {
        cout << "Key length and input texts length are not the same!" << endl;
        return vector<vector<unsigned char>>();
    }

    vector<vector<unsigned char>> outTextVector(inpTextVector.size(), vector
    for (size_t i = 0; i < outTextVector.size(); i++)
    {
        for (size_t j = 0; j < elemSize; j++)

```

```

        {
            outTextVector[i][j] = inpTextVector[i][j] ^ key[j];
        }
    }

    return outTextVector;
}

vector<string> cryptGamming(const vector<unsigned char>& key, vector<vector<
{
    size_t elemSize = inpTextVector[0].size();
    for (size_t i = 0; i < inpTextVector.size(); i++)
    {
        if (inpTextVector[i].size() != elemSize)
        {
            cout << "Input texts lengths are not the same!" << endl;
            return vector<string>();
        }
    }

    if (key.size() != elemSize)
    {
        cout << "Key length and input texts length are not the same!" << endl;
        return vector<string>();
    }

    vector<string> outTextVector(inpTextVector.size(), string(elemSize, '0'))
    for (size_t i = 0; i < outTextVector.size(); i++)

```

```

    {
        for (size_t j = 0; j < elemSize; j++)
        {
            outTextVector[i][j] = inpTextVector[i][j] ^ key[j];
        }
    }

    return outTextVector;
}

string hackText(const string& templateP1, const vector<unsigned char>& encTe
{
    if (templateP1.size() != encText1.size() || templateP1.size() != encText
    {
        cout << "Template and encrypted messages lengths are not the same!"
        return string();
    }

    string hackedTextP2(templateP1.size(), '0');

    for (size_t i = 0; i < hackedTextP2.size(); i++)
    {
        hackedTextP2[i] = (encText1[i] ^ encText2[i] ^ templateP1[i]);
    }

    return hackedTextP2;
}

```

```

int main()
{
    std::setlocale(LC_ALL, "ru");

    const string P1 = "НаВашисходящийот1204";
    const string P2 = "ВСеверныйфилиалБанка";
    vector<string> messages = { P1, P2 };
    cout << "Input texts:\n" << P1 << '\n' << P2 << endl;

    const vector<unsigned char> key = {
        0x05, 0x0C, 0x17, 0x7F, 0x0E, 0x4E, 0x37, 0xD2, 0x94, 0x10,
        0x09, 0x2E, 0x22, 0x57, 0xFF, 0xC8, 0x0B, 0xB2, 0x70, 0x54
    };
    cout << "\nInput key:" << endl;
    printHex(key);

    vector<vector<unsigned char>> encTexts(cryptGamming(key, messages));
    cout << "\nEncrypted texts:" << endl;
    printHex(encTexts);

    vector<string> decTexts(cryptGamming(key, encTexts));
    cout << "\nDecrypted texts from encrypted texts:" << endl;
    printStrings(decTexts);

    string templateP1(P1);
    for (size_t i = 0; i < templateP1.size() / 3; i++)
    {
        templateP1[i] = '?';
    }
}

```

```
cout << "\nP1 template:\n" << templateP1 << endl;

string hackedP2(hackText(templateP1, encTexts[0], encTexts[1]));
cout << "\nDecrypted part of text P2:\n" << hackedP2 << endl;

return EXIT_SUCCESS;
}
```

```

1  #include <cstdlib>
2  #include <iostream>
3  #include <vector>
4  #include <string>
5
6
7  using std::cin;
8  using std::cout;
9  using std::endl;
10 using std::string;
11 using std::vector;
12
13
14 void printHex(const vector<unsigned char>& vec)
15 {
16     cout << std::hex << std::uppercase;
17
18     for (size_t i = 0; i < vec.size(); i++)
19     {
20         cout << static_cast<short>(vec[i]) << ' ';
21     }
22
23     cout << std::dec << std::nouppercase << endl;
24 }
25
26 void printHex(const vector<vector<unsigned char>>& vec)
27 {
28     cout << std::hex << std::uppercase;
29
30     for (size_t i = 0; i < vec.size(); i++)
31     {
32         for (size_t j = 0; j < vec[i].size(); j++)
33         {
34             cout << static_cast<short>(vec[i][j]) << ' ';
35         }
36         cout << endl;
37     }
38
39     cout << std::dec << std::nouppercase;
40 }
41
42 void printStrings(const vector<string>& str)
43 {
44     for (size_t i = 0; i < str.size(); i++)
45     {
46         cout << str[i] << endl;
47     }
48 }

```

Рис. 4.1: Однократное гаммирование различных текстов одним ключом на C++.

Листинг. 1

```

50 vector<vector<unsigned char>> cryptGaming(const vector<unsigned char>& key, const vector<string>& inpTextVector)
51 {
52     size_t elemSize = inpTextVector[0].size();
53     for (size_t i = 0; i < inpTextVector.size(); i++)
54     {
55         if (inpTextVector[i].size() != elemSize)
56         {
57             cout << "Input texts lengths are not the same!" << endl;
58             return vector<vector<unsigned char>>();
59         }
60     }
61
62     if (key.size() != elemSize)
63     {
64         cout << "Key length and input texts length are not the same!" << endl;
65         return vector<vector<unsigned char>>();
66     }
67
68     vector<vector<unsigned char>> outTextVector(inpTextVector.size(), vector<unsigned char>(elemSize));
69     for (size_t i = 0; i < outTextVector.size(); i++)
70     {
71         for (size_t j = 0; j < elemSize; j++)
72         {
73             outTextVector[i][j] = inpTextVector[i][j] ^ key[j];
74         }
75     }
76
77     return outTextVector;
78 }
79
80
81 vector<string> cryptGaming(const vector<unsigned char>& key, vector<vector<unsigned char>>& inpTextVector)
82 {
83     size_t elemSize = inpTextVector[0].size();
84     for (size_t i = 0; i < inpTextVector.size(); i++)
85     {
86         if (inpTextVector[i].size() != elemSize)
87         {
88             cout << "Input texts lengths are not the same!" << endl;
89             return vector<string>();
90         }
91     }
92
93     if (key.size() != elemSize)
94     {
95         cout << "Key length and input texts length are not the same!" << endl;
96         return vector<string>();
97     }
98
99     vector<string> outTextVector(inpTextVector.size(), string(elemSize, '0'));
100     for (size_t i = 0; i < outTextVector.size(); i++)
101     {
102         for (size_t j = 0; j < elemSize; j++)
103         {
104             outTextVector[i][j] = inpTextVector[i][j] ^ key[j];
105         }
106     }
107
108     return outTextVector;
109 }
110

```

Рис. 4.2: Однократное гаммирование различных текстов одним ключом на C++.

Листинг. 2

```

112 string hackText(const string& templateP1, const vector<unsigned char>& encText1, const vector<unsigned char>& encText2)
113 {
114     if (templateP1.size() != encText1.size() || templateP1.size() != encText2.size())
115     {
116         cout << "Template and encrypted messages lengths are not the same!" << endl;
117         return string();
118     }
119
120     string hackedTextP2(templateP1.size(), '0');
121
122     for (size_t i = 0; i < hackedTextP2.size(); i++)
123     {
124         hackedTextP2[i] = (encText1[i] ^ encText2[i] ^ templateP1[i]);
125     }
126
127     return hackedTextP2;
128 }
129
130
131 int main()
132 {
133     std::setlocale(LC_ALL, "ru");
134
135     const string P1 = "НаВашисходящий1204";
136     const string P2 = "ВСеверныйФилиалБанка";
137     vector<string> messages = { P1, P2 };
138     cout << "Input texts:\n" << P1 << '\n' << P2 << endl;
139
140     const vector<unsigned char> key = {
141         0x05, 0x0C, 0x17, 0x7F, 0x0E, 0x4E, 0x37, 0xD2, 0x94, 0x10,
142         0x09, 0x2E, 0x22, 0x57, 0xFF, 0xC8, 0xB2, 0x0B, 0x70, 0x54
143     };
144     cout << "\nInput key:" << endl;
145     printHex(key);
146
147     vector<vector<unsigned char>> encTexts(cryptGamming(key, messages));
148     cout << "\nEncrypted texts:" << endl;
149     printHex(encTexts);
150
151     vector<string> decTexts(cryptGamming(key, encTexts));
152     cout << "\nDecrypted texts from encrypted texts:" << endl;
153     printStrings(decTexts);
154
155     string templateP1(P1);
156     for (size_t i = 0; i < templateP1.size() / 3; i++)
157     {
158         templateP1[i] = '?';
159     }
160
161     cout << "\nP1 template:\n" << templateP1 << endl;
162
163     string hackedP2(hackText(templateP1, encTexts[0], encTexts[1]));
164     cout << "\nDecrypted part of text P2:\n" << hackedP2 << endl;
165
166
167     return EXIT_SUCCESS;
168 }

```

Рис. 4.3: Однократное гаммирование различных текстов одним ключом на C++.

### Листинг. 3

2. Результат работы программы после компиляции и выполнения (рис. 4.4).



```
Input texts:
НаВашисходящийот1204
ВСеверныйфилиалБанка

Input key:
5 C 17 7F E 4E 37 D2 94 10 9 2E 22 57 FF C8 B B2 70 54

Encrypted texts:
C8 EC D5 9F F6 A6 C6 27 7A F4 F6 D7 CA BE 11 3A 3A 80 40 60
C7 DD F2 9D EB BE DA 29 7D E4 E1 C5 CA B7 14 9 EB 5F 9A B4

Decrypted texts from encrypted texts:
НаВашисходящийот1204
ВСеверныйфилиалБанка

P1 template:
?????сходящийот1204

Decrypted part of text P2:
0J↑="'ныйфилиалБанка
```

Рис. 4.4: Компиляция и выполнение программы

## **5 Анализ результатов**

Работа выполнена без каких-либо серьезных нареканий. Обожаю C++.

## 6 Выводы

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

## Список литературы

1. Терминал Linux [Электронный ресурс]. URL: [%7Bhttps://www.reg.ru/blog/linux-shpargalka-komandy-terminala-dlya-novichkov/%7D](https://www.reg.ru/blog/linux-shpargalka-komandy-terminala-dlya-novichkov/).
2. Гаммирование [Электронный ресурс]. Wikipedia Inc. URL: [https://en.wikipedia.org/wiki/XOR\\_cipher](https://en.wikipedia.org/wiki/XOR_cipher).