

Лабораторная работа №7

Элементы криптографии. Однократное гаммирование

Ким М. А.

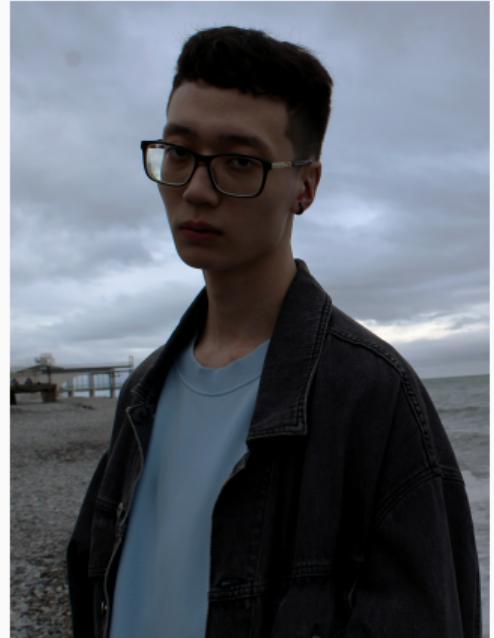
21 октября 2023

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Ким Михаил Алексеевич
- студент уч. группы НФИбд-01-20
- Российский университет дружбы народов
- 1032201664@pfur.ru
- <https://github.com/exmanka>



Вводная часть

Актуальность

- Необходимость навыков работы с различными ОС, git, Markdown, ЯП.

Объект и предмет исследования

- Операционная система Rocky Linux
- Язык программирования C++
- Язык разметки Markdown
- Однократное гаммирование

Цели и задачи

- Освоить на практике применение режима однократного гаммирования.

Процесс выполнения работы

Создание программы

```
1 #include <iostream>
2 #include <iostream>
3 #include <vector>
4 #include <string>
5
6 using std::cin;
7 using std::cout;
8 using std::endl;
9 using std::string;
10 using std::vector;
11
12 void printVec(vector<unsigned char> vec)
13 {
14     cout << std::hex << std::uppercase;
15     for (size_t i = 0; i < vec.size(); i++)
16     {
17         cout << static_cast<char>(vec[i]) << " ";
18     }
19     cout << std::dec << std::nouppercase << endl;
20 }
21
22 vector<unsigned char> encryptGaming(const vector<unsigned char> key, const string& pureText)
23 {
24     if (key.size() != pureText.size())
25     {
26         cout << "Key length and text length are not the same!" << endl;
27         return vector<unsigned char>{'0'};
28     }
29
30     vector<unsigned char> encryptedText(key.size());
31     for (size_t i = 0; i < key.size(); i++)
32     {
33         encryptedText[i] = pureText[i] ^ key[i];
34     }
35     return encryptedText;
36 }
37
38 string decryptGaming(const vector<unsigned char> key, vector<unsigned char> encryptText)
39 {
40     if (key.size() != encryptText.size())
41     {
42         cout << "Key length and encrypt text length are not the same!" << endl;
43         return "0";
44     }
45
46     string decryptedText(key.size(), '0');
47     for (size_t i = 0; i < key.size(); i++)
48     {
49         decryptedText[i] = encryptText[i] ^ key[i];
50     }
51
52     return decryptedText;
53 }
```

```
54
55 vector<unsigned char> findKey(vector<unsigned char> encryptText, const string& pureText)
56 {
57     if (EncryptText.size() != pureText.size())
58     {
59         cout << "Pure text length and encrypt text length are not the same!" << endl;
60         return vector<unsigned char>{'0'};
61     }
62
63     vector<unsigned char> key(encryptText.size());
64     for (size_t i = 0; i < encryptText.size(); i++)
65     {
66         key[i] = encryptText[i] ^ pureText[i];
67     }
68
69     return key;
70 }
71
72 int main()
73 {
74     std::setlocale(LC_ALL, "ru");
75
76     vector<unsigned char> exampleKey =
77     {
78         0x0B, 0x0C, 0x17, 0x07, 0x0C, 0x0E, 0x37, 0x02, 0x09, 0x10,
79         0x09, 0x2C, 0x27, 0x0F, 0x08, 0x0D, 0x07, 0x70, 0x00,
80     };
81
82     string examplePureText("Привет - Be happy!");
83
84     cout << "Example key:" << endl;
85     printVec(exampleKey);
86     cout << "\nExample encrypted message:\n" << examplePureText << endl;
87
88     vector<unsigned char> encryptedMessage = encryptGaming(exampleKey, examplePureText);
89     cout << "Calculated encrypted message:" << endl;
90     printVec(encryptedMessage);
91
92     cout << "\nCalculated decrypted message:" << endl;
93     cout << decryptGaming(exampleKey, encryptedMessage) << endl << endl;
94
95     vector<unsigned char> exampleKey2 =
96     {
97         0x0B, 0x0C, 0x17, 0x07, 0x0E, 0x37, 0x02, 0x09, 0x10,
98         0x09, 0x2C, 0x27, 0x05, 0x04, 0x03, 0x07, 0x0B, 0x0C, 0x04,
99     };
100
101     cout << "Wrong key:" << endl;
102     printVec(exampleKey2);
103     cout << "\nDecrypted message from wrong key:\n" << decryptGaming(exampleKey2, encryptedMessage) << endl << endl;
104
105     string wonderText1 = "# some random C++ code";
106     cout << "\nDecrypted message:\n" << wonderText1 << endl;
107
108     vector<unsigned char> predictedKey = findKey(encryptedMessage, wonderText1);
109     cout << "Predicted key for wonder message:" << endl;
110     printVec(predictedKey);
111
112     string calculatedWonderText = encryptGaming(predictedKey, encryptedMessage);
113     cout << "\nDecrypted message from this key:\n" << calculatedWonderText << endl;
114 }
```

```
115
116 string taskMessage("C Новый Годик, друзья!");
117 cout << "Vnkey for message " << taskMessage << endl;
118 printVec(findKey(encryptedMessage, taskMessage));
119
120
121 return EXIT_SUCCESS;
122 }
```

Результат работы программы

```
Example key:  
5 C 17 7F E 4E 37 D2 94 10 9 2E 22 57 FF C8 B B2 70 54  
  
Example decrypted message:  
Штирлиц - Вы Герой!!  
  
Calculated encrypted message:  
0D FE FF 8F E5 A6 C1 F2 2 30 CB D5 2 94 1A 38 E5 5B 51 75  
  
Calculated decrypted message:  
Штирлиц - Вы Герой!!  
  
Wrong key:  
5 C 17 7F E 4E 37 D2 94 10 9 2E 22 55 F4 D3 7 BB BC 54  
  
Decrypted message from wrong key:  
Штирлиц - Вы Болван!  
  
Wondered message:  
Я очень люблю C++!!!  
  
Predicted key for wondered message:  
2 DE 11 78 0 4B 3D D2 E9 CE 2A 3E FC B4 59 13 CE 7A 70 54  
  
Decrypted message from this key:  
Я очень люблю C++!!!  
  
Key for message 'С Новым Годом, друзья':  
C DE 32 61 7 5D 2D D2 C1 DE 2F 3B EE B8 FE C8 16 BC AD 8A
```

Результаты

Результаты

- Выполнены все необходимые задания.

Вывод

Освоено на практике применение режима однократного гаммирования.