Front matter

title: "Лабораторная работа 8" author: "Попова Юлия Дмитриевна"

Generic otions

lang: ru-RU toc-title: "Содержание"

Bibliography

bibliography: bib/cite.bib csl: pandoc/csl/gost-r-7-0-5-2008-numeric.csl

Pdf output format

toc: true # Table of contents toc_depth: 2 lof: true # List of figures lot: true # List of tables fontsize: 12pt linestretch: 1.5 papersize: a4 documentclass: scrreprt

l18n

polyglossia-lang: name: russian options: - spelling=modern - babelshorthands=true polyglossia-otherlangs: name: english

Fonts

mainfont: PT Serif romanfont: PT Serif sansfont: PT Sans monofont: PT Mono mainfontoptions: Ligatures=TeX romanfontoptions: Ligatures=TeX sansfontoptions: Ligatures=TeX,Scale=MatchLowercase monofontoptions: Scale=MatchLowercase,Scale=0.9

Biblatex

biblatex: true biblio-style: "gost-numeric" biblatexoptions:

- parentracker=true
- backend=biber
- hyperref=auto
- language=auto
- autolang=other*
- citestyle=gost-numeric

Misc options

indent: true header-includes:

- \linepenalty=10 # the penalty added to the badness of each line within a paragraph (no associated penalty node) Increasing the value makes tex try to have fewer lines in the paragraph.

- \interlinepenalty=0 # value of the penalty (node) added after each line of a paragraph.
- \hyphenpenalty=50 # the penalty for line breaking at an automatically inserted hyphen
- \exhyphenpenalty=50 # the penalty for line breaking at an explicit hyphen
- \binoppenalty=700 # the penalty for breaking a line at a binary operator
- \relpenalty=500 # the penalty for breaking a line at a relation
- \clubpenalty=150 # extra penalty for breaking after first line of a paragraph
- \widowpenalty=150 # extra penalty for breaking before last line of a paragraph
- \displaywidowpenalty=50 # extra penalty for breaking before last line before a display math
- \brokenpenalty=100 # extra penalty for page breaking after a hyphenated line
- \predisplaypenalty=10000 # penalty for breaking before a display
- \postdisplaypenalty=0 # penalty for breaking after a display
- \floatingpenalty = 20000 # penalty for splitting an insertion (can only be split footnote in standard LaTeX)
- \raggedbottom # or \flushbottom
- \usepackage{float} # keep figures where there are in the text
- \floatplacement{figure}{H} # keep figures where there are in the text

---

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

дисциплина: Информационная безопасность

Преподователь: Кулябов Дмитрий Сергеевич

Студент: Попова Юлия Дмитриевна

Группа: НФИбд-01-19

МОСКВА

2022 г.

Цель работы

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

Выполнение лабораторной работы

Постановка задачи Два текста кодируются одним ключом (однократное гаммирование). Требуется не зная ключа и не стремясь его определить, прочитать оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты P1 и P2 в режиме однократного гаммирования. Приложение должно определить вид шифротекстов C1 и C2 обоих текстов P1 и P2 при известном ключе ; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить.

Для этого есть функция позволяющая зашифровывать, расшифровывать данные с помощью сообщения и ключа, а также позволяющая получить ключ (@fig:001).

```
vector<uint8_t> encrypt(vector<uint8_t> message, vector<uint8_t> key)
{
    if (message.size() != key.size())
    {
        return {};
    }
    vector<uint8_t> encrypted;
    for (int i = 0; i < message.size(); i++)
    {
        encrypted.push_back(message[i] ^ key[i]);
    }
    return encrypted;
}
```

{#fig:001 width=100%}

Функция для вывода результатов (@fig:002)

```
void print_bytes(vector<uint8_t> message)
{
    for (const auto& c : message)
    {
        cout << hex << unsigned(c) << "";
    }
    cout << endl;
}

void print_text(vector<uint8_t> message)
{
    string str(message.begin(), message.end());
    cout << str << endl;
}
```

{#fig:002 width=100%}

Фунцкия определения текста, зная два шифротекста и оригинальный текст одного из них (@fig:003)

```
vector<uint8_t> get_message_with_three_pieces(vector<uint8_t> cr1, vector<uint8_t> cr2, vector<uint8_t> msg1)
{
    if (cr1.size() != cr2.size() and cr1.size() != msg1.size())
    {
        return {};
    }
    vector<uint8_t> msg2;
    for (int i = 0; i < cr1.size(); i++)
    {
        msg2.push_back(cr1[i] ^ cr2[i] ^ msg1[i]);
    }
    return msg2;
```

{#fig:003 width=100%}

Главная функция (@fig:004)

```
int main()
{
    string message1 = "hello this is lab 8";
    string message2 = "this lab 8 ab hello";
    vector<uint8_t> first(message1.begin(), message1.end());
    vector<uint8_t> second(message2.begin(), message2.end());

    string keystr = "thisiskeystringlab7";
    vector<uint8_t> key(keystr.begin(), keystr.end());

    vector<uint8_t> crypt1 = encrypt(first, key);
    vector<uint8_t> crypt2 = encrypt(second, key);


    cout << "Original Message number 1: " << endl;
    print_text(first);
    cout << endl << "Original Message number 2: " << endl;
    print_text(second);
    cout << endl << "Crypted message number 1: " << endl;
    print_bytes(crypt1);
    cout << endl << "Crypted message number 2: " << endl;
    print_bytes(crypt2);


    cout << endl <<"Finding message 2:" << endl;
    vector<uint8_t> msg_found = get_message_with_three_pieces(crypt1, crypt2, first);
    print_text(msg_found);
    return 0;
}
```

{#fig:004 width=100%}

Запускаем программу, получаем два шифротекста для каждого текста при известном ключе. Далее не зная ключа и не стремясь его определить, получаем текст (@fig:005)

```
Original Message number 1:
hello this is lab 8

Original Message number 2:
this lab 8 ab hello

Crypted message number 1:
1cd51f6531fd100541b1a4ebd342f

Crypted message number 2:
0000491fa7594b5413b4ef9de58

Finding message 2:
this lab 8 ab hello
```

{#fig:005 width=100%}

Способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить: злоумышленник может получить два зашифрованных текста, например, во время передачи информации через сеть. Также если он сможет получить часть оригинального сообщения одного из двух зашифрованных текстов, он сможет прочитать оба текста и без ключа.

Выводы

В результате выполнения работы освоили на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом

Список литературы

1. Методические материалы курса