Лабораторная работа №2: Шифры перестановки

Дисциплина: Математические основы защиты информации и информационной безопасности

Манаева Варвара Евгеньевна, НФИмд-01-24, 1132249514

28 сентября 2024

Российский университет дружбы народов, Москва, Россия

Цель работы

Ознакомиться с классическими примерами шифров перестановки.

Задание

1. Реализовать шифры, представленные в задании.

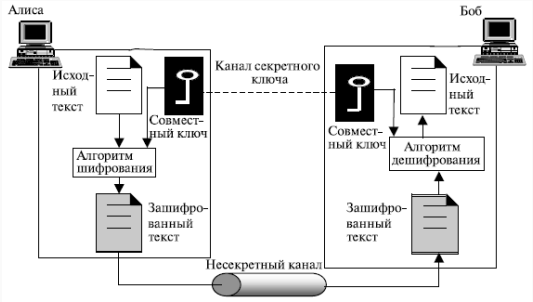
Теоретическое введение

Виды шифров

Шифры подразделяются на:

- Симметричные;
- Асимметричные.

Виды симметричных шифров



Среди симметричных шифров выделяют:

- Шифры перестановки;
- Шифры подстановки.

Выполнение лабораторной работы

Реализация маршрутного шифрования (1)

```
function routeEncoding(text::AbstractString, code::AbstractString)::AbstractString
    indexes = sortperm(split(code, ""))
    n = length(code)
    while mod(length(text), n) != 0
        text *= "a"
    end
    println("Text to be transformed:\n", text)
    m = div(length(text), n)
    t = split(text, "")
    t = reshape(t, n, m)
    temp = copy(t)
    for i in 1:n
        temp[i, :] = t[indexes[i], :]
    end
    encoded_text = ""
    for i in 1:n
        encoded_text *= join(temp[i, :])
    end
    return encoded_text
end
```

Реализация маршрутного шифрования (2)

```
function routeDecoding(text::AbstractString, code::AbstractString)::AbstractString
    indexes = sortperm(sortperm(split(code, "")))
    n = length(code)
    println("Text to be transformed:\n", text)
    m = div(length(text), n)
    t = split(text, "")
    t = reshape(t, m, n)
    temp = copy(t)
    for i in 1:n
        temp[:, i] = t[:, indexes[i]]
    end
    encoded_text = ""
    for i in 1:m
        encoded_text *= join(temp[i, :])
    end
    return encoded_text
end
```

Результат работы кода для маршрутного шифрования

```
coded_text = routeEncoding("нельзянедооцениватьпротивника", "пароль")
println("The result of encoding:\n", coded_text, "\n\n")
decoded_text = routeDecoding(coded_text, "пароль")
println("The result of decoding:\n", decoded_text)
```

```
Text to be transformed:
нельзянедооцениватьпротивникаа
The result of encoding:
еенпнзоатаьовокннеьвлдирияцтиа


Text to be transformed:
еенпнзоатаьовокннеьвлдирияцтиа
The result of decoding:
нельзянедооцениватьпротивникаа
```

Реализация шифрования с помощью решёток (1.1)

```
function reshetEncoding(text::AbstractString, code::AbstractString, prorezy::Vector)
    text, k = err_handl(text)
    println("Text to be transformed:\n", text)
    te = split(text, "")
    if k == 1
        print("Cannot be encoded due to algorithm restrictions")
        return
    end
    if length(code) > 2*k
        code = code[1:2*k]
    elseif length(code) < 2*k
        while length(code) < 2*k
            code *= "a"
        end
    end
    if length(prorezy) != k^2
        prorezy = rand(1:4, k^2)
    end
    cuts_mask = Array{Integer, 2}(undef, 2*k, 2*k)
    cuts_mask[1:k, 1:k] = [prorezy[i+k*j] == 1 ? i+k*j : 0 for j=0:k-1,i=1:k]
    cuts_mask[1:k, k+1:2*k] = [prorezy[i+k*j] == 2 ? i+k*j : 0 for j=0:k-1,i=k:-1:1]
    cuts_mask[k+1:2*k, k+1:2*k] = [prorezy[i+k*j] == 3 ? i+k*j : 0 for j=k-1:-1:0,i=k:-1:1]
    cuts_mask[k+1:2*k, 1:k] = [prorezy[i+k*j] == 4 ? i+k*j : 0 for j=k-1:-1:0,i=1:k]
    t = Array{AbstractString, 2}(undef, 2*k, 2*k)
    for i in 1:4
        for j in 1:k^2
            t[findfirst(x -> x== j, cuts_mask)] = te[(i-1)*k^2+j]
        end
        cuts_mask = rotr90(cuts_mask)
    end
    println("Code utilized:\n", code)
    indexes = sortperm(split(code, ""))
    temp = copy(t)
    for i in 1:2*k
        temp[i, :] = t[indexes[i], :]
    end
    encoded_text = ""
    for i in 1:2*k
        encoded_text *= join(temp[i, :])
    end
    return encoded_text, code, prorezy
```

Реализация шифрования с помощью решёток (1.2)

```
function err_handl(text)
    check = false
    while !check
        try
            Int(sqrt(length(text))/2)
        catch
            text *= "a"
        else
            check = true
        end
    end
    return text, Int(sqrt(length(text))/2)
end
```

Реализация шифрования с помощью решёток (2)

```
function reshetDecoding(text::AbstractString, code::AbstractString, prorezy)
    indexes = sortperm(sortperm(split(code, "")))
    k = Int(sqrt(length(code)))
    println("Text to be transformed:\n", text)
    t = split(text, "")
    t = reshape(t, 2*k, 2*k)
    temp = copy(t)
    for i in 1:2*k
        temp[:, i] = t[:, indexes[i]]
    end
    for i in 1:2*k
        t[i, :] = temp[:, i]
    end
    cuts_mask = Array{Integer, 2}(undef, 2*k, 2*k)
    cuts_mask[1:k, 1:k] = [prorezy[i+k*j] == 1 ? i+k*j : 0 for j=0:k-1,i=1:k]
    cuts_mask[1:k, k+1:2*k] = [prorezy[i+k*j] == 2 ? i+k*j : 0 for j=0:k-1,i=k:-1:1]
    cuts_mask[k+1:2*k, k+1:2*k] = [prorezy[i+k*j] == 3 ? i+k*j : 0 for j=k-1:-1:0,i=k:-1:1]
    cuts_mask[k+1:2*k, 1:k] = [prorezy[i+k*j] == 4 ? i+k*j : 0 for j=k-1:-1:0,i=1:k]
    encoded_text = ""
    for i in 1:4
        for j in 1:k^2
            encoded_text *= t[findfirst(x -> x== j, cuts_mask)]
        end
        cuts_mask = rotr90(cuts_mask)
    end
    return encoded_text
end
```

Результат работы кода для шифрования с помощью решёток

```
coded_text, code, cuts = reshetEncoding("договорподписали", "шифр", [2,3,3,4])
println("The result of encoding:\n", coded_text, "\n\n")
result = reshetDecoding(coded_text, code, cuts)
println("The result of decoding:\n", result)
```

```
Text to be transformed:
договорподписали
Code utilized:
шифр
The result of encoding:
ппиаоровооигсдлд


Text to be transformed:
ппиаоровооигсдлд
The result of decoding:
договорподписали
```

Реализация таблиц Виженера (1)

```
function VigenereTable(text::AbstractString, code::AbstractString, isEncoded::Bool)::AbstractString
    t = filter(isascii,text)
    code = filter(isascii,code)
    println("Text to be encoded:\n", t, "; \nCode:\"", code, "\"")
    code = Int.(only.(split(code, "")))
    if isEncoded
        code = (-1).*code
    end
    temp = only.(split(t,""))
    for i in 1:length(temp)
        temp[i] = Char(mod(Int(temp[i])+code[mod(i, length(code))+1], 128))
    end
    t = join(temp)
    return t
end
```

Реализация таблиц Виженера (2)

```
function VigenereTable(text::AbstractString, code::AbstractString, isEncoded::Bool)::AbstractString
    t = filter(isascii,text)
    code = filter(isascii,code)
    println("Text to be encoded:\n", t, "; \nCode:\"", code, "\"")
    code = Int.(only.(split(code, "")))
    if isEncoded
        code = (-1).*code
    end
    temp = only.(split(t,""))
    for i in 1:length(temp)
        temp[i] = Char(mod(Int(temp[i])+code[mod(i, length(code))+1], 128))
    end
    t = join(temp)
    return t
end
```

Результат работы кода для таблиц Виженера

```
coded_text = VigenereTable("TEXT to be coded!!!! αβγ and some innocent letters", "alphabet", false)
println("The result of encoding:\n", coded_text, "\n\n")
decoded_text = VigenereTable(coded_text, "alphabet", true)
println("The result of decoding:\n", decoded_text)
```

```
Text to be encoded:
TEXT to be coded!!!!  and some innocent letters;
Code:"alphabet"
The result of encoding:
◀       ᴦᴦ|UOP┼[POJ¶JZ^WDGShᴦXU\UGWg


Text to be encoded:
◀       ᴦᴦ|UOP┼[POJ¶JZ^WDGShᴦXU\UGWg;
Code:"alphabet"
The result of decoding:
TEXT to be coded!!!!  and some innocent letters
```

Выводы по проделанной работе

Вывод

В результате работы мы ознакомились с традиционными моноалфавитными шрифтами простой замены, а именно:

- Маршрутным шифрованием;
- Шифрованием с помощью решёток;
- Таблицами Виженера.

Были записаны скринкасты:

- выполнения лабораторной работы;
- создания отчёта по результатам выполения лабораторной работы;
- создания презентации по результатам выполнения лабораторной работы;
- защиты лабораторной работы.