|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/12 Интеллектуальный анализ больших**

**данных в системах поддержки принятия решений.**

**Отчет**

**по лабораторной работе № 3**

**Вариант № 9**

**Название:** реализация простейшего генератора паролей

**Дисциплина:** Информационная безопасность автоматизированных систем

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-31М |  |  | И.С. Марчук |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Д.А. Миков |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2024

**Цель:** получение основных теоретических сведений и практических навыков по оценке стойкости парольной защиты.

**Задание:** 1. Вычислить по формуле (2) нижнюю границу S\* для заданных P, V, T.

2. Выбрать некоторый алфавит с мощностью A и получить минимальную длину пароля L, при котором выполняется условие (3).

3. Реализовать программу-генератор паролей пользователей. Программа должна формировать случайную последовательность символов длины L, при этом должен использоваться алфавит из A символов.

Условия варианта:

* Номер варианта - 9;
* P = 10^-4;
* V = 3 пароля в минуту;
* T = 15 дней;

**Ход работы**

Я реализовал на языке Kotlin при помощи стандартных компонентов библиотеки Swing. Интерфейс программы состоит из 1 окна, на котором пользователь может выбрать параметры и алфавит для генерации пароля. Исходный код программы представлен в листинге 1.

Листинг программы 1 – Программа генерации пароля

package org.example

import java.util.\*

import javax.swing.\*

import javax.swing.text.AttributeSet

import javax.swing.text.PlainDocument

fun main() {

// создание окна

val frame = JFrame("Генератор паролей пользователей")

frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE)

frame.setSize(800, 400)

frame.setLocationRelativeTo(null)

val charactersFlags = arrayOf(false, false, false, false)

frame.add(JCheckBox("Латиница").apply {

setBounds(450, 25, 250, 20)

check(true)

addActionListener { charactersFlags[0] = isSelected }

})

frame.add(JCheckBox("Кириллица").apply {

setBounds(450, 50, 250, 20)

check(true)

addActionListener { charactersFlags[1] = isSelected }

})

frame.add(JCheckBox("Заглавная латиница").apply {

setBounds(450, 75, 250, 20)

check(true)

addActionListener { charactersFlags[2] = isSelected }

})

frame.add(JCheckBox("Заглавная кириллица").apply {

setBounds(450, 100, 250, 20)

check(true)

addActionListener { charactersFlags[3] = isSelected }

})

frame.add(JLabel("P(Вероятность)").apply {

setBounds(25, 25, 250, 20)

})

val pInput = frame.add(JTextField().apply {

document = JTextFiledNumbersAndDots(20)

setBounds(250, 25, 100, 20)

text = "0.0001"

}) as JTextField

frame.add(JLabel("V(Скорость перебора)(Пароля/Мин)").apply {

setBounds(25, 50, 250, 20)

})

val vInput = frame.add(JTextField().apply {

document = JTextFiledNumbersAndDots(20)

setBounds(250, 50, 100, 20)

text = "3"

}) as JTextField

frame.add(JLabel("T(Срок действия пароля) (Дней)").apply {

setBounds(25, 75, 250, 20)

})

val tInput = frame.add(JTextField().apply {

document = JTextFiledNumbersAndDots(20)

setBounds(250, 75, 100, 20)

text = "15"

}) as JTextField

val output = frame.add(JLabel().apply {

setBounds(25, 100, 350, 80)

text = "Введите параметры и нажмите сгенерировать"

}) as JLabel

val random = Random()

frame.add(JButton("Сгенерировать пароль!").apply {

setBounds(25, 325, 740, 20)

addActionListener {

if (!charactersFlags[0] && !charactersFlags[1] && !charactersFlags[2] && !charactersFlags[3]) {

output.text = "Слишком маленький алфавит"

} else {

val p = floatFromStringOrNull(pInput.text.toString())

val v = floatFromStringOrNull(vInput.text.toString())

val t = floatFromStringOrNull(tInput.text.toString())

if (p == null) {

output.text = "Неправильный формат параметра P"

} else if (v == null) {

output.text = "Неправильный формат параметра V"

} else if (t == null) {

output.text = "Неправильный формат параметра T"

} else {

// Мощность алфавита

val a = 0 +

(if (charactersFlags[0]) 26 else 0) +// Латиница

(if (charactersFlags[1]) 32 else 0) +// Кириллица

(if (charactersFlags[2]) 26 else 0) +// Заглавная латиница

(if (charactersFlags[3]) 32 else 0) // Заглавная кириллица

// println(log(8.toDouble(), 2.toDouble())) = 3

// A^L >= S = VT/P

val s = (v \* t / p).toDouble()

val lPrev = kotlin.math.log(s, a.toDouble())

// длинна пароля

val l = kotlin.math.ceil(lPrev).toInt()

// генерируем пароль

val password = java.lang.StringBuilder()

for(i in 1..l){

var code = random.nextInt(a)

if(charactersFlags[0]){

if(code < 26){

password.append(Char(97 + code))

continue

}else{

code -= 26

}

}

if(charactersFlags[1]){

if(code < 32){

password.append(Char(1072 + code))

continue

}else{

code -= 32

}

}

if(charactersFlags[2]){

if(code < 26){

password.append(Char(65 + code))

continue

}else{

code -= 26

}

}

if(charactersFlags[3]){

if(code < 32){

password.append(Char(1040 + code))

continue

}else{

code -= 32

}

}

}

output.text = "<html>S\*=VT/P Нижняя граница: $s<br>" +

"A Мощность алфавита: $a<br>" +

"L Длинна пароля: ${l}<br>" +

"<br>Сгенерированный пароль: $password"

"</html>"

}

}

}

})

frame.layout = null

// переотрисовка

frame.isVisible = true

}

fun floatFromStringOrNull(input: String): Float? {

try {

return input.toFloat()

} catch (e: Exception) {

// ignore

}

return null

}

class JTextFiledNumbersAndDots(

private val limit: Int

) : PlainDocument() {

override fun insertString(offs: Int, str: String?, a: AttributeSet?) {

if (str == null)

return

// проверка на символы

val newInput = StringBuilder()

str.forEach {

if (it.code in 48..58 || it == '.') newInput.append(it)

}

// если не достигли максимальной длины строки

if (length + newInput.length <= limit)

super.insertString(offs, newInput.toString(), a)

}

}

**Пример работы программы**

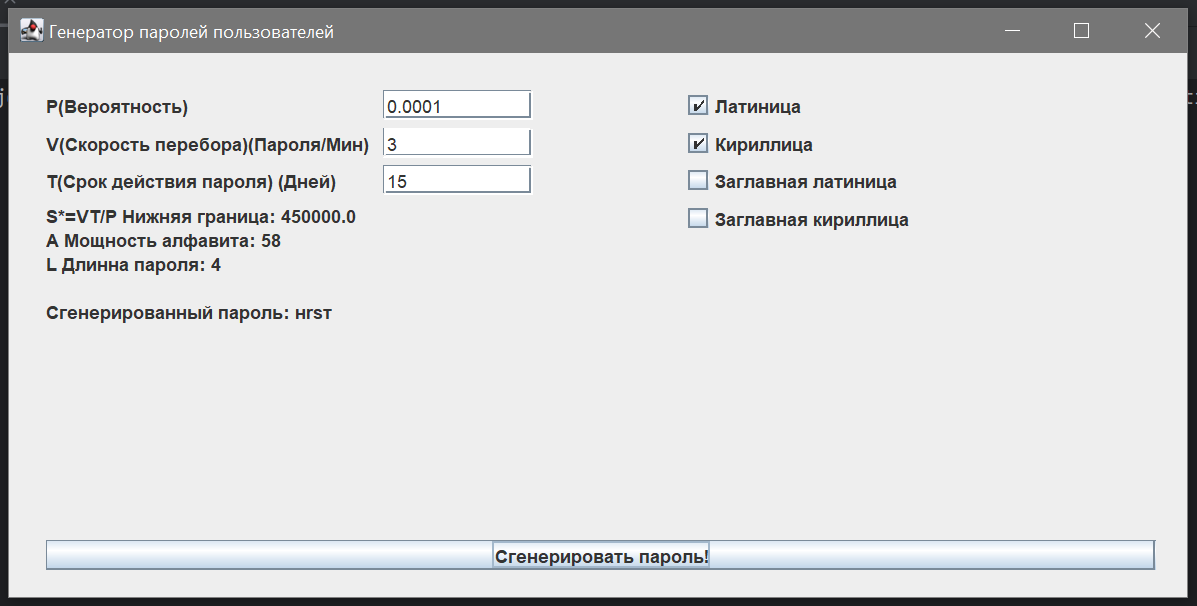


Рисунок 1 – Окно генерации пароля

**Контрольные вопросы**

1. Дать определение стойкости пароля к взлому. Написать формулу.

Надежность пароля — это показатель эффективности пароля против угадывания или атак методом перебора. В своей обычной форме он оценивает, сколько попыток потребуется в среднем злоумышленнику, не имеющему прямого доступа к паролю, чтобы правильно его угадать. Надежность пароля зависит от длины, сложности и непредсказуемости.



1. Дать определение мощности алфавита паролей.

мощность алфавита паролей – количество символов, которые могут быть использованы при составлении пароля. Например, если пароль состоит только из малых английских букв, то A = 26

1. Перечислить основные задачи, которые могут решаться с использованием определения стойкости пароля.

Определение минимальной оптимальной длины пароля; определение минимальной оптимальной мощности словаря пароля.

4. Перечислить основные требования к выбору пароля.

1) Минимальная длина пароля - как минимум 6 символов;

2) Пароль должен состоять из различных групп символов;

3) В качестве пароля не должны использоваться реальные слова.

**Вывод**

Я разработал программу, программу-генератор паролей пользователей, которая позволяет формировать случайную последовательность символов заданной длинны, при помощи заданного алфавита. А также получил основные теоретические сведения и практические навыки по оценке стойкости парольной защиты.