Министерство науки И ВЫСШЕГО ОБРАЗОВАНИЯ   
Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский Томский политехнический Университет»



Инженерная школа информационных технологий и робототехники

Отделение автоматизации и робототехники

15.04.04 «Автоматизация технологических процессов и производств»

Вариант №3

Лабораторная работа №1

**«Аутентификация. Количественная оценка стойкости парольной защиты»**

по дисциплине:

**«Информационная безопасность автоматизированных систем»**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Исполнитель:** |  | | | | | | |
| студент группы | 8ТМ22 |  | Гительман Владислав Сергеевич |  | |  | |
|  |  |  |  |  | |  | |
| **Руководитель:** |  | | | | | | |
| к.т.н., доцент | ОАР |  | Суходоев Михаил Сергеевич | |  | |  | |
|  |  |  |  |  | |  | |

Томск – 2023

**Цель практической работы**

Изучить метод количественной оценки стойкости парольной защиты и программно реализовать простейший генератор паролей, обладающий требуемой стойкостью к взлому.

**Задание на практическую работу**

1. В таблице 1 найти для указанного варианта значения характеристик P, V, T.

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

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

4. Реализовать программу для генерации паролей пользователей. Программа должна формировать случайную последовательность символов длины L, при этом должен использоваться алфавит из A символов. Сгенерированный пароль необходимо записать в текстовый файл. Сохранить программу под именем Password\_gen.

5. Внести изменения в программу таким образом, чтобы пароль длины L включал в себя одновременно буквы, цифры и символы. Сохранить программу под именем Password\_gen\_1.

6. Оформить отчет по лабораторной работе.

Данные для варианта №3 приведены в таблице 1.

Таблица 1 – Данные для варианта №3

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | *P* | *V* | *T* |
| 3 | 10-7 | 100 паролей/день | 10 дней |

**Ход практической работы**

Лабораторная работа выполняется с использованием языка C# (Visual Studio). Для реализации используются формулы, позволяющие определить оптимальный пароль.

A – мощность алфавита паролей (количество символов, которые могут быть использованы при составлении пароля: если пароль состоит только из малых английских букв, то A = 26), L – длина пароля, S = AL – число всевозможных паролей длины L, которые можно составить из символов алфавита A, V – скорость перебора паролей злоумышленником, T – максимальный срок действия пароля.

Тогда, вероятность P подбора пароля злоумышленником в течение срока его действия V определяется по следующей формуле:

P = (V ∙ T) / S = (V ∙ T) / AL.

Задача. Определить минимальные мощность алфавита паролей A и длину паролей L, обеспечивающих вероятность подбора пароля злоумышленником не более заданной P, при скорости подбора паролей V, максимальном сроке действия пароля T.

Данная задача имеет неоднозначное решение. При исходных данных V, T, P однозначно можно определить лишь нижнюю границу S\* числа всевозможных паролей. Целочисленное значение нижней границы вычисляется по формуле

S\* = [V ∙ T / P], (1)

где [] – целая часть числа, взятая с округлением вверх.

После определения нижней границы S\* необходимо выбрать такие A и L для формирования S = AL, чтобы выполнялось следующее неравенство:

S\* ≤ S = AL. (2)

При выборе S, удовлетворяющего неравенству (2), вероятность подбора па¬роля злоумышленника (при заданных V и T) будет меньше, чем заданная P.

Следует отметить, что при осуществлении вычислений по формулам (1) и (2), величины должны быть приведены к одним размерностям.

Далее в соответствии с данными формулами было определено выражение для L:



Данное неравенство в дальнейшем будет использоваться при написании программного кода.

Как было ранее упомянуто, программирование осуществляется с использованием Visual Studio. Была разработана форма для взаимодействия пользователя с интерфейсом генерации паролей (Рисунок 1).

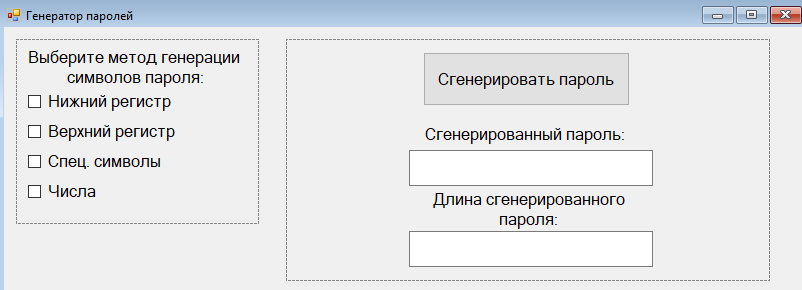


Рисунок 1 – Пользовательская форма

Пользователю необходимо выбрать флаг для генерации символов определенного типа. Нажать сгенерированный пароль. В результате чего на форме появится пароль и его длина. Пример работы программы приведен на рисунке 2.

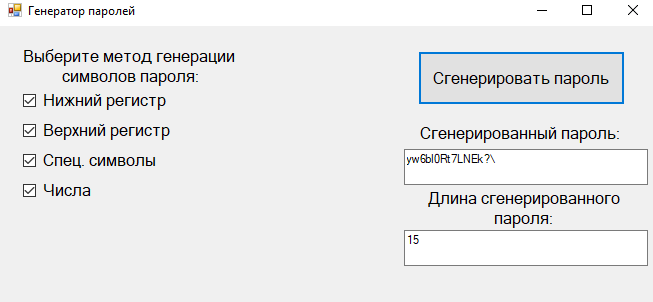


Рисунок 2 – Результат работы программы

Программный код, реализующий взаимодействия с элементами формы, приведен в Листинге 1.

Разработан программный код в среде Arduino IDE, позволяющий передавать данные от клиента к MQTTX, а также принимать их от брокера и менять скважность светодиода (Листинг 1).

Листинг 1 – Программный код для работы с элементами формы

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using static System.Console;

namespace project1\_password

{

public partial class Form1 : Form

{

bool Low\_reg\_checked = false;

bool High\_reg\_checked = false;

bool Special\_symbol\_checked = false;

Продолжение Листинга 1

bool Numbers\_symbol\_checked = false;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void button\_generate\_Click(object sender, EventArgs e)

{

my\_password\_class my\_class = new my\_password\_class();

var password = my\_class.GetPass(Low\_reg\_checked, High\_reg\_checked, Special\_symbol\_checked, Numbers\_symbol\_checked);

textBox1.Text = password;

textBox2.Text = (Convert.ToString(password.Length));

//string file = @"Textfile\_pass.txt";

//File.Create(file);

string path = @"C:\temp\readmePASS.txt";

string contents = password;

File.WriteAllText(path, contents);

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

}

private void panel2\_Paint(object sender, PaintEventArgs e)

{

}

private void Power\_Al\_label\_Click(object sender, EventArgs e)

{

}

Продолжение Листинга 1

private void Low\_reg\_check\_CheckedChanged(object sender, EventArgs e)

{

if (Low\_reg\_checked == false)

{

Low\_reg\_checked = true;

}

else

Low\_reg\_checked = false;

}

private void high\_regs\_check\_CheckedChanged(object sender, EventArgs e)

{

if (High\_reg\_checked == false)

{

High\_reg\_checked = true;

}

else

High\_reg\_checked = false;

}

private void Special\_symbols\_check\_CheckedChanged\_1(object sender, EventArgs e)

{

if (Special\_symbol\_checked == false)

{

Special\_symbol\_checked = true;

}

else

Special\_symbol\_checked = false;

}

private void Numbers\_symbols\_check\_CheckedChanged(object sender, EventArgs e)

{

if (Numbers\_symbol\_checked == false)

{

Numbers\_symbol\_checked = true;

}

else

Numbers\_symbol\_checked = false;

}

}

}

Из Листинга 1 видно, что для генерации паролей используется класс и его метод GetPass. Данный класс был разработан в отдельном файле. Код программы, описывающий данный класс и его функции, приведен в Листинге под номером 2.

Листинг 2 – Программный код, реализующий класс

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace project1\_password

{

class my\_password\_class

{

Random rnd\_new = new Random();

bool LR = false; // LOW Register flag

bool HR = false; // HIGH Register flag

bool SSY = false; // SYMBOL SPEC flag

bool N = false; // NUM FLAG SPEC flag, in the bottow, there are combination of them (внизу комбинация флагов)

bool HRLR = false;

bool LRSSY = false;

bool LRN = false;

bool HRSSY = false;

bool HRN = false;

bool SSYN = false;

bool LRHRSSY = false;

bool LRHRSSYN = false;

bool LRHRN = false;

bool HRSSYN = false;

private void bools(bool LR1, bool HR1, bool SSY1, bool N1, bool HRLR1, bool LRSSY1, bool LRN1, bool HRSSY1, bool HRN1, bool SSYN1, bool LRHRSSY1, bool LRHRN1, bool LRHRSSYN1, bool HRSSYN1)

{ // функция для получения переменных, обозначающих комбинации флагов (или одиночные флаги)

// чтобы понимать, сколько одновременнно нажато флажков (напр., большие и малые регистры одновременно)

LR = LR1;

HR = HR1;

SSY = SSY1;

N = N1;

HRLR = HRLR1;

LRSSY = LRSSY1;

LRN = LRN1;

HRSSY = HRSSY1;

HRN = HRN1;

Продолжение Листинга 2

SSYN = SSYN1;

LRHRSSY = LRHRSSY1;

LRHRN = LRHRN1;

LRHRSSYN = LRHRSSYN1;

HRSSYN = HRSSYN1;

}

//bool Low\_reg\_checked = false;

// bool High\_reg\_checked = false;

//bool Special\_symbol\_checked = false;

//bool Numbers\_symbol\_checked = false;

int A\_ = 0;

private int generator\_special\_sym() // генератор спец. символов

{

// Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 4); // случайно будем генерировать спец. символы разных диапазонов ASCII

int rnd\_to\_return = 0;

if (rnd\_num == 0)

{

rnd\_to\_return = rnd\_new.Next(33, 47); //генерируем спец. символы №1

}

else if (rnd\_num == 1)

{

rnd\_to\_return = rnd\_new.Next(58, 64); //генерируем спец. символы №2

}

else if (rnd\_num == 2)

{

rnd\_to\_return = rnd\_new.Next(91, 96); //генерируем спец. символы №3

}

else if (rnd\_num == 3)

{

rnd\_to\_return = rnd\_new.Next(123, 126); //генерируем спец. символы №4 (в скобках - диапазон в ASCII)

}

return rnd\_to\_return;

}

public string GetPass(bool Low\_reg\_checked, bool High\_reg\_checked, bool Special\_symbol\_checked, bool Numbers\_symbol\_checked)

{ //функция для получения пароля

if ((Low\_reg\_checked == true) && (!High\_reg\_checked) && (!Special\_symbol\_checked) && (!Numbers\_symbol\_checked))

{

Продолжение Листинга 2

A\_ = 26;//мощность алфавита. bools() используется для получения комбинаций флагов.

bools(true, false, false, false, false, false, false, false, false, false, false, false, false, false);

}

else if ((High\_reg\_checked == true) && (!Low\_reg\_checked) && (!Special\_symbol\_checked) && (!Numbers\_symbol\_checked))

{

A\_ = 26;//мощность алфавита

bools(false, true, false, false, false, false, false, false, false, false, false, false, false, false);

}

else if ((Special\_symbol\_checked == true) && (!High\_reg\_checked) && (!Low\_reg\_checked) && (!Numbers\_symbol\_checked))

{

A\_ = 32;//мощность алфавита

bools(false, false, true, false, false, false, false, false, false, false, false, false, false, false);

}

else if ((Numbers\_symbol\_checked == true) && (!High\_reg\_checked) && (!Low\_reg\_checked) && (!Special\_symbol\_checked))

{

A\_ = 10;//мощность алфавита

bools(false, false, false, true, false, false, false, false, false, false, false, false, false, false);

}

else if ((Low\_reg\_checked == true) && (High\_reg\_checked == true) && (!Special\_symbol\_checked) && (!Numbers\_symbol\_checked))

{

A\_ = 52;//мощность алфавита

bools(false, false, false, false, true, false, false, false, false, false, false, false, false, false);

}

else if ((Low\_reg\_checked == true) && (Special\_symbol\_checked == true) && (!Numbers\_symbol\_checked) && (!High\_reg\_checked))

{

A\_ = 58;//мощность алфавита

bools(false, false, false, false, false, true, false, false, false, false, false, false, false, false);

}

else if ((Low\_reg\_checked == true) && (Numbers\_symbol\_checked == true) && (!High\_reg\_checked) && (!Special\_symbol\_checked))

{

A\_ = 36;//мощность алфавита

Продолжение Листинга 2

bools(false, false, false, false, false, false, true, false, false, false, false, false, false, false);

}

else if ((High\_reg\_checked == true) && (Special\_symbol\_checked == true) && (!Numbers\_symbol\_checked) && (!Low\_reg\_checked))

{

A\_ = 58;//мощность алфавита

bools(false, false, false, false, false, false, false, true, false, false, false, false, false, false);

}

else if ((High\_reg\_checked == true) && (Numbers\_symbol\_checked == true) && (!Low\_reg\_checked) && (!Special\_symbol\_checked))

{

A\_ = 36;//мощность алфавита

bools(false, false, false, false, false, false, false, false, true, false, false, false, false, false);

}

else if ((Special\_symbol\_checked == true) && (Numbers\_symbol\_checked == true) && (!Low\_reg\_checked) && (!High\_reg\_checked))

{

A\_ = 42;//мощность алфавита

bools(false, false, false, false, false, false, false, false, false, true, false, false, false, false);

}

else if ((Low\_reg\_checked == true) && (High\_reg\_checked == true) && (Special\_symbol\_checked == true) && (!Numbers\_symbol\_checked))

{

A\_ = 84;//мощность алфавита

bools(false, false, false, false, false, false, false, false, false, false, true, false, false, false);

}

else if ((Low\_reg\_checked == true) && (High\_reg\_checked == true) && (!Special\_symbol\_checked) && (Numbers\_symbol\_checked == true))

{

A\_ = 62;//мощность алфавита

bools(false, false, false, false, false, false, false, false, false, false, false, true, false, false);

}

else if ((Low\_reg\_checked == true) && (High\_reg\_checked == true) && (Special\_symbol\_checked == true) && (Numbers\_symbol\_checked == true))

{

A\_ = 94;//мощность алфавита

bools(false, false, false, false, false, false, false, false, false, false, false, false, true, false); }

Продолжение Листинга 2

else if ((Low\_reg\_checked == false) && (High\_reg\_checked == true) && (Special\_symbol\_checked == true) && (Numbers\_symbol\_checked == true))

{

A\_ = 94;//мощность алфавита

bools(false, false, false, false, false, false, false, false, false, false, false, false, false, true);

}

else

{

A\_ = 0;//мощность алфавита

bools(false, false, false, false, false, false, false, false, false, false, false, false, false, false);

}

string Password = ""; //объявляем пустую строку и инициализируем её.

if (A\_ != 0) // если мощность алфавита не равна нулю

{

double L = Math.Ceiling(10 / Math.Log10(A\_)); // вычисляем по формуле оптимальную длину пароля

//Random rnd\_extra = new Random();

int value\_extra = rnd\_new.Next(0, 10); // генерируем случайное число от 0 до 10

L = L + value\_extra; // добавляем его к длине пароля

int intLength = Convert.ToInt32(L); //конвертируем число в целое

int[] arr = new int[intLength]; //объявлем массив полученной ранее длины

bool checkHR = false; // флаги для генерации символов

bool checkLR = false;

bool checkSSY = false;

bool checkN = false;

/\* LR = LR1;

HR = HR1;

SSy = SSY1;

N = N1;

HRLR = HRLR1;

LRSSY = LRSSY1;

LRN = LRN1;

HRSSY = HRSSY1;

HRN = HRN1;

SSYN = SSYN1;

LRHRSSY = LRHRSSY1;

LRHRSSYN = LRHRSSYN1; \*/

for (int i = 0; i < arr.Length; i++){

Продолжение Листинга 2

if (LR == true)

{

arr[i] = rnd\_new.Next(97, 122); //генерация букв малого регистра

}

else if (HR == true)

{

arr[i] = rnd\_new.Next(65, 90); //генерация букв большого регистра

}

else if (N == true)

{

arr[i] = rnd\_new.Next(48, 57); //генерация чисел

}

else if (SSY == true)

{

arr[i] = generator\_special\_sym(); // генерация спец. символов

}

else if (HRLR == true) // если выбрано несколько флагов (и высокий, и низкий регистр)

{

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 2); //генерируется случайное число в диапазоне

if (checkLR == false || checkHR == false) //если ни разу не был сгенерирован выбранный тип символов

rnd\_num = -1; // не обращаем внимание на ранее сгенерированное число в диапазоне

if (rnd\_num == 0 || checkLR == false) //заходим в каждое из условий, пока ни разу не сгенерирован выбранный тип символов

{ // если каждый тип символов сгенерирован, то заходим в условия в зависимости от rnd\_num

arr[i] = rnd\_new.Next(97, 122);

checkLR = true;

}

else if (rnd\_num == 1 || checkHR == false)

{

arr[i] = rnd\_new.Next(65, 90);

checkHR = true;

}

}

else if (LRSSY == true)//далее аналогично предыдущему:

{

Продолжение Листинга 2

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 2);

if (checkLR == false || checkSSY == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkLR == false)

{

arr[i] = rnd\_new.Next(97, 122);

checkLR = true;

}

else if (rnd\_num == 1 || checkSSY == false)

{

arr[i] = generator\_special\_sym(); //генератор спец. символов

checkSSY = true;

}

}

else if (LRN == true)

{

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 2);

if (checkLR == false || checkN == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkLR == false)

{

arr[i] = rnd\_new.Next(97, 122);

checkLR = true;

}

else if (rnd\_num == 1 || checkN == false)

{

arr[i] = rnd\_new.Next(48, 57);

checkN = true;

}

}

else if (HRSSY == true)

{

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 2);

if (checkHR == false || checkSSY == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkHR == false)

{

arr[i] = rnd\_new.Next(65, 90);

checkHR = true;

Продолжение Листинга 2

}

else if (rnd\_num == 1 || checkSSY == false)

{

arr[i] = generator\_special\_sym();

checkSSY = true;

}

}

else if (HRN == true)

{

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 2);

if (checkN == false || checkHR == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkHR == false)

{

arr[i] = rnd\_new.Next(65, 90);

checkHR = true;

}

else if (rnd\_num == 1 || checkN == false)

{

arr[i] = rnd\_new.Next(48, 57);

checkN = true;

}

}

else if (SSYN == true)

{

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 2);

if (checkSSY == false || checkN == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkSSY == false)

{

arr[i] = rnd\_new.Next(48, 57);

checkSSY = true;

}

else if (rnd\_num == 1 || checkN == false)

{

arr[i] = generator\_special\_sym();

checkN = true;

}

}

else if (LRHRSSY == true)

{

//Random rnd\_new = new Random();

Продолжение Листинга 2

int rnd\_num = rnd\_new.Next(0, 3);

if (checkLR == false || checkSSY == false || checkHR == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkLR == false)

{

arr[i] = rnd\_new.Next(97, 122);

checkLR = true;

}

else if (rnd\_num == 1 || checkHR == false)

{

arr[i] = rnd\_new.Next(65, 90);

checkHR = true;

}

else if (rnd\_num == 2 || checkSSY == false)

{

arr[i] = generator\_special\_sym();

checkSSY = true;

}

}

else if (LRHRN == true)

{

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 3);

if (checkLR == false || checkN == false || checkHR == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkLR == false)

{

arr[i] = rnd\_new.Next(97, 122);

checkLR = true;

}

else if (rnd\_num == 1 || checkHR == false)

{

arr[i] = rnd\_new.Next(65, 90);

checkHR = true;

}

else if (rnd\_num == 2 || checkN == false)

{

arr[i] = rnd\_new.Next(48, 57);

checkN = true;

}

}

else if (LRHRSSYN == true)

Продолжение Листинга 2

{

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 4);

if (checkLR == false || checkHR == false || checkN == false || checkSSY == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkLR == false)

{

arr[i] = rnd\_new.Next(97, 122);

checkLR = true;

}

else if (rnd\_num == 1 || checkHR == false)

{

arr[i] = rnd\_new.Next(65, 90);

checkHR = true;

}

else if (rnd\_num == 2 || checkSSY == false)

{

arr[i] = generator\_special\_sym();

checkSSY = true;

}

else if (rnd\_num == 3 || checkN == false)

{

arr[i] = rnd\_new.Next(48, 57);

checkN = true;

}

}

else if (HRSSYN == true)

{

//Random rnd\_new = new Random();

int rnd\_num = rnd\_new.Next(0, 3);

if (checkHR == false || checkN == false || checkSSY == false)

rnd\_num = -1;

if (rnd\_num == 0 || checkHR == false)

{

arr[i] = rnd\_new.Next(65, 90);

checkHR = true;

}

else if (rnd\_num == 1 || checkSSY == false)

{

arr[i] = rnd\_new.Next(65, 90);

Продолжение Листинга 2

checkSSY = true;

}

else if (rnd\_num == 2 || checkSSY == false)

{

arr[i] = generator\_special\_sym();

checkSSY = true;

}

else if (rnd\_num == 3 || checkN == false)

{

arr[i] = rnd\_new.Next(48, 57);

checkN = true;

}

}

Password += (char)arr[i]; //присваиваем строке сгенерир. символ на каждой итерации

}

}

int r;

char p;

char[] mas = new char[Password.Length];

mas = Password.ToCharArray();

for (int i = mas.Length - 1; i > 1; i--) //перемешиваем случайным образом полученные символы пароля

{

r = rnd\_new.Next(0, i - 1);

p = mas[i];

mas[i] = mas[r];

mas[r] = p;

}

Password = "";

Password = String.Concat<char>(mas); //преобразуем каждый элемент

// массива в строку и складываем их в Password

//Password = mas.ToString();

return Password; //возвращаем полученое значение пароля

}

}

}

В рамках данного программного кода дополнительно реализовано перемешивание символов пароля в случайном порядке для увеличения безопасности пароля.

**Выводы**

В результате выполнения лабораторной работы изучен метод количественной оценки стойкости парольной защиты и программно реализован простейший генератор паролей, обладающий требуемой стойкостью к взлому.