**Лабораторна робота №4**

**Тема:** ” Розробка і дослідження засобів аутентифікації користувачів в

комп’ютерних системах”

**Мета роботи:** засвоїти методику та отримати практичні навики побудови засобів аутентифікації користувачів.

**Короткі теоретичні відомості**

Аутентифікація - полягає в періодичній (стохастичній) перевірці достовірності ідентифікації користувача. Така процедура проводиться для повторної перевірки користувача. Аутентифікація може здійснюватись як апаратними, так і програмними методами за якимись особистими ознаками, чи персональними відомостями користувача. При апаратній реалізації користувач може бути аутентифікованим за певними фізичними ознаками: вага тіла, колір очей, відбитки пальців, геометрія долоні, код ДНК і т.п. Окрім того, можуть використовуватись додаткові особисті пристрої: наручні браслети, ключі і т.д. Даний вид аутентифікації характеризується вищим рівнем надійності, проте є складнішим та дорожчим у використанні, тому він використовується на підприємствах, де необхідно забезпечити високий рівень захисту інформації. Дешевший варіант аутентифікації користувачів полягає у створенні програмних засобів. Резидентна програма періодично з певним кроком часу задає випадковим чином запитання із заздалегідь створеного файлу, або ж випадкові три- , чотири- розрядні десяткові числа. КС порівнює відповіді з наперед зареєстрованими, або ж обчисленими відповідями, і на основі цього надає, або забороняє роботу користувача. У випадку правильної відповіді за користувачем залишаються його права, а у випадку неправильної відповіді - користувач втрачає права доступу і повинен заново ввійти в систему. Стійкість даного виду аутентифікації забезпечується конфіденційністю інформації

Наведемо основні способи аутентифікації користувачів:

- наперед визначена інформація, якою може користуватися користувач: пароль, персональний ідентифікаційний номер, домовленість про використання спеціальних закодованих фраз;

- елементи апаратного забезпечення, якими може користуватися користувач: ключі, магнітні картонки, мікросхеми і т.п.;

- характерні особисті ознаки користувача: відбитки пальців, рисунок ставки ока, тембр голосу і т.п.;

- характерні навики та риси поведінки користувача в режимі реального часу: особливості динаміки та стиль роботи на клавіатурі, прийоми роботи з маніпулятором і т.п.;

- навики та знання користувачів, обумовлені освітою, культурою, навчанням, вихованням, звичками і т.п.

**Хід роботи**

Номер варіанту вибрану за номером в списку - 8. Згідно варіанту потрібно створити програму аутентифікації користувачів, з кількість всіх запитань -12, кількість запитань в одній ітерації - 4 та періодом повтору 5 хвилини.

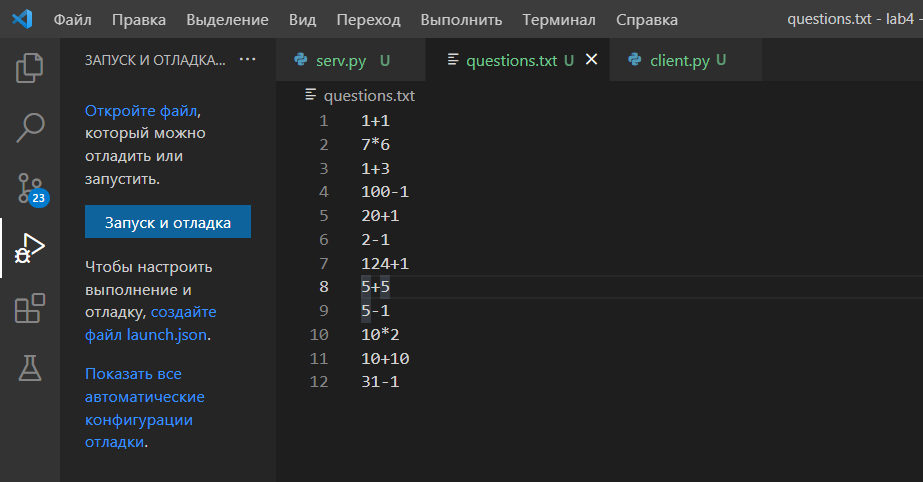


Рисунок 1.1 - Лістинг файлу з запитаннями

Лістинг 1.1 – Код файлу serv.py

import socket

import threading

import random

import math

import time

import json

server = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

Dict\_data = {}

MaxSizeofUsers = 8

Stop\_thread = False

Thread\_control\_dict = {}

IsConnectSocket = {}

NumberOfQuestions = 4

def Autentification(user, userID, login):

    print("a1")

    for i in range(NumberOfQuestions):

        print("a2")

        count = 0

        rnd\_value = (random.randint(0, 12)) #Кількість запитань

        f = open('questions.txt', 'r')

        for line in f:

            count +=1

            if(count == rnd\_value):

                user.send(b'auth')

                user.send(line.encode())

                result = eval(line)

                try:

                    us\_result = int(user.recv(2048).decode())

                    print(user, "answer", us\_result)

                    if (result == us\_result):

                        print(user, i + 1, "Authentication passed")

                        user.send(str(i + 1).encode() + b' Authentication passed')

                        if(i == 4):

                            LoginLen = len(login)

                            SpaceStr = ''

                            if (LoginLen > 5):

                                NumSpaces = 8 - (LoginLen - 5)

                                for i in range(NumSpaces):

                                    SpaceStr = SpaceStr + ' '

                            f = open('logs.txt', 'a')

                            f.write(str(userID) + '\t' + login + SpaceStr + time.ctime() + '\t' + "Connected " + '\t' + "online" + '\t' 'Authentication passed successfully' + '\n')

                            f.close()

                    if (result != us\_result):

                        CloseConnection(user=user, userID=userID, login=login)

                        IsConnectSocket[user] = False

                        return

                except:

                    CloseConnection(user=user, userID=userID, login=login)

                    IsConnectSocket[user] = False

                    return

        f.close()

def CloseConnection(user, userID, login):

    print(user, "Authentication not passed, close the connection")

    user.send(b'Authentication not passed, close the connection')

    LoginLen = len(login)

    SpaceStr = ''

    if (LoginLen > 5):

        NumSpaces = 8 - (LoginLen - 5)

        for i in range(NumSpaces):

            SpaceStr = SpaceStr + ' '

    f = open('logs.txt', 'a')

    f.write(str(

        userID) + '\t' + login + SpaceStr + time.ctime() + '\t' + "disconnect" + '\t' + "offline" + '\t' 'Authentication not passed' + '\n')

    f.close()

    user.close()

def run():

    while True:

        socket\_user, adress\_user = server.accept()

        userID = socket\_user.recv(2048).decode()

        if(CheckID(userID)):

            print(userID)

            print(socket\_user, " connected")

            rnd\_value = (random.randint(0, 20))

            socket\_user.send(str(rnd\_value).encode())

             #Варіант 8

            y = int(userID)\*math.sin(1/rnd\_value)

            #Варіант 8

            print("y = ", y)

            UserY = float(socket\_user.recv(2048).decode())

            if(y == UserY):

                socket\_user.send("You have been connected".encode())

                Thread\_control\_dict[socket\_user] = False

                IsConnectSocket[socket\_user] = True

                login = getUserLogin(userID)

                f = open('logs.txt', 'a')

                LoginLen = len(login)

                SpaceStr = ''

                if(LoginLen > 5):

                    NumSpaces = 8 - (LoginLen - 5)

                    for i in range(NumSpaces):

                        SpaceStr = SpaceStr + ' '

                f.write(str(userID)+'\t'+login+SpaceStr+time.ctime()+'\t'+"connect"+'\t\t'+"online"+'\n')

                f.close()

                thread\_for\_listen = threading.Thread(target=ping\_user, args=(socket\_user, userID, login, ))

                thread\_for\_listen.start()

                thread\_for\_listen2 = threading.Thread(target=getDataFrmUser, args=(socket\_user,))

                thread\_for\_listen2.start()

        else:

            socket\_user.send("Your ID is not registered!".encode())

            socket\_user.close()

def getUserLogin(id):

    with open('user.txt', 'r') as json\_file:

        obj = json.load(json\_file)

        for i in obj['users']:

            if (i['ID'] == id):

                return i['login']

def getDataFrmUser(user):

    while True:

        if(Thread\_control\_dict[user] == False):

            try:

                data = user.recv(2048)

                data = data.decode()

                print(data)

            except:

                pass

        else:

            return

def ping\_user(user, userID, login):

    while True:

        if(IsConnectSocket[user] == True):

            Thread\_control\_dict[user] = True

            time.sleep(3) #Період повтору процедури аутентифікації, хв

            Autentification(user, userID, login)

            Thread\_control\_dict[user] = False

        else:

            return

def getSizeofUsers():

    count = 0

    with open('user.txt', 'r') as json\_file:

        obj = json.load(json\_file)

        for i in obj['users']:

            count+=1

    return count

def CheckID(id):

    with open('user.txt', 'r') as json\_file:

        obj = json.load(json\_file)

        for i in obj['users']:

            if (i['ID'] == id):

                return True

    return False

def Commands():

    while True:

        try:

            command = input()

            if (command == "/journal"):

                ListUser()

            elif(command == "/register"):

                Register()

            elif (command == "/del"):

                DelUser()

        except:

            pass

def CreateFile():

    f = open('logs.txt', 'w')

    f.write('ID\tLOGIN\t\tDATE\t\t\tOPERATION\tSTATUS\tREASON\n')

    f.close()

def ListUser():

    f = open('logs.txt', 'r')

    for line in f:

        print(line)

    f.close()

def CheckSizeJson():

    with open('user.txt', 'r') as json\_file:

        try:

            obj = json.load(json\_file)

            if(obj == None):

                print("Null")

        except:

            print("mu;;")

def isFileExist():

    with open('user.txt', 'r') as json\_file:

        try:

            obj = json.load(json\_file)

        except:

            Dict\_data['users'] = []

def Register():

    if(getSizeofUsers() < 8):

        print("Input the ID of a new user")

        ID = input()

        if(CheckID(ID) == False):

            print("Input users login")

            login = input()

            print("Input users password")

            password = input()

            print("Input users rights")

            rights = input()

            with open('user.txt', 'r') as json\_file:

                try:

                    obj = json.load(json\_file)

                    obj['users'].append({

                        'ID': str(ID),

                        'login': login,

                        'password': password,

                        'rights': rights

                    })

                    with open('user.txt', 'w') as outfile:

                        json.dump(obj, outfile, indent=4)

                except:

                    Dict\_data['users'].append({

                        'ID': str(ID),

                        'login': login,

                        'password': password,

                        'rights': rights

                    })

                    with open('user.txt', 'w') as outfile:

                        json.dump(Dict\_data, outfile, indent=4)

            print("User registered")

        else:

            print("The ID is already registered")

            Register()

    else:

        print("The number of users is max")

def DelUser():

    print("Input the ID")

    ID = input()

    with open('user.txt', 'r') as json\_file:

        obj = json.load(json\_file)

    for i in obj['users']:

        if(i['ID'] == ID):

            obj['users'].remove(i)

            print("true")

            break

    with open('user.txt', 'w') as outfile:

        json.dump(obj, outfile, indent=4)

if \_\_name\_\_ == '\_\_main\_\_':

    CreateFile()

    isFileExist()

    thread\_for\_listen = threading.Thread(target=Commands)

    thread\_for\_listen.start()

    server.bind(("127.0.0.1", 5555))

    server.listen(MaxSizeofUsers)

    run()

Лістинг 1.2 – Код файлу client.py

import socket

import threading

import math

import sys

client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

def listen\_system():

    count = 0

    while True:

        try:

            data = client.recv(2048)

            if(data.decode("utf-8") == "ping"):

                pass

            if (data.decode("utf-8") == "auth"):

                if(count == 0):

                    print("You need to pass authentication")

                count =+ 1

                func = client.recv(2048).decode()

                print("Count a value", func)

                result = input()

                client.send(result.encode())

                server\_answer = client.recv(2048).decode()

                if(server\_answer == 'Authentication not passed, close the connection'):

                    print(server\_answer)

                    client.close()

                    sys.exit()

                print(server\_answer)

                if(server\_answer == '4 Authentication passed'):

                    print('You passed all tests!')

                    count = 0

            else:

                print(data.decode("utf-8"))

        except:

            client.close()

def ConnectSystem():

    print("Input your ID")

    MyID = input()

    client.send(MyID.encode("utf-8"))

    rnd\_x = client.recv(2048).decode()

    try:

        rnd\_x = int(rnd\_x)

        print("x = ", rnd\_x)

        intId = int(MyID)

        #Варіант 8

        y = intId\*math.sin(1/rnd\_x)

        #Варіант 8

        print("y = ", y)

        client.send(str(y).encode())

        thread\_for\_listen = threading.Thread(target=listen\_system)

        thread\_for\_listen.start()

    except ValueError:

        print(rnd\_x)

        client.close()

if \_\_name\_\_ == '\_\_main\_\_':

    client.connect(("127.0.0.1", 5555))

    ConnectSystem()

Результат виконання аутентифікації в клієнта зображено на рисунку 4.2.

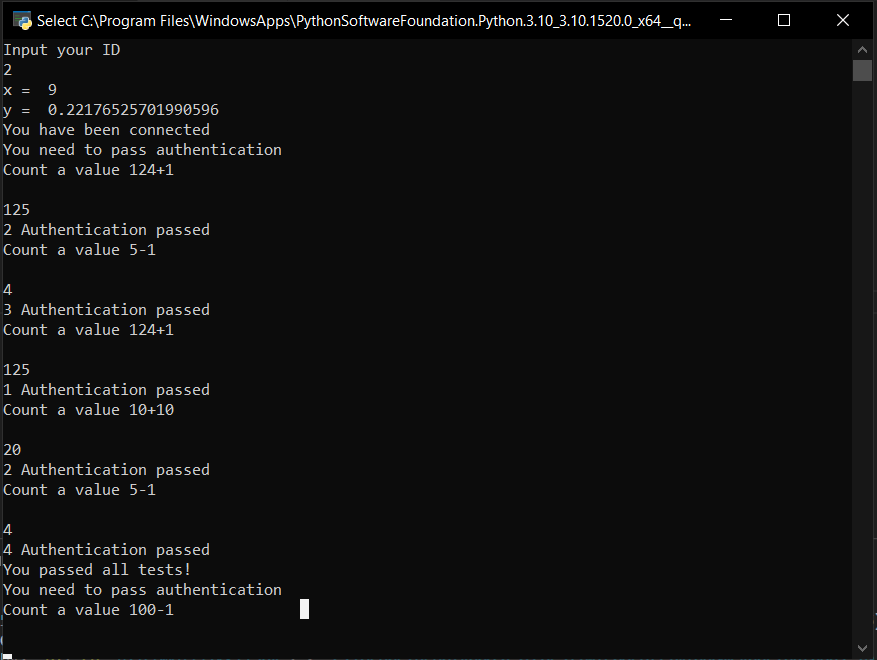


Рисунок 4.2 Аутентифікація

**Висновок**: на даній лабораторній роботі я засвоїв методику та отримав практичні навики побудови засобів аутентифікації користувачів.