ИНТЕЛЛЕКТУАЛЬНЫЕ МЕТОДЫ В ЗАДАЧАХ ЗАЩИТЫ ОТ КИБЕРУГРОЗ

ЛАБОРАТОРНАЯ РАБОТА № 1
Исследование и реализация OTP алгоритмов.

Выполнил:
Мосолков Е.Н.
Преподаватель:
Исхаков А.Ю.

Москва 2021 г.

ЦЕЛЬ РАБОТЫ

Изучение алгоритмов реализации технологии OTP

ЗАДАЧА ПРАКТИЧЕСКОЙ РАБОТЫ

Разработать собственную программную реализацию OTP. Программа должна состоять из двух функциональных модулей – сервис аутентификации (логин, пароль) и программный токен. Минимально необходимо реализовать 1 любой метод генерации OTP *. Языки и среды программирования любые.

ОПИСАНИЕ ФУНКЦИОНАЛА

В рамках практической работы был реализован алгоритм TOTP.

Система состоит из 3х программ:

- token.py – токен, который генерирует и показывает код для аутентификации
- client.py – клиентская часть для авторизации и аутентификации в «системе»
- server.py – сервер, который производит валидацию токена и авторизацию пользователя

Данные о пользователях хранятся в файле credentials.txt

Основной алгоритм работы системы - следующий:

1. Запускается сервер и токен
2. Запускается клиент
3. Клиент подключается к серверу (общение реализовано на основе сокетов)
4. Сервер дает понять клиенту что тот подключился к серверу
5. Клиент запрашивает у пользователя связку логин/пароль и отправляет их на сервер
6. Сервер получает связку и проверяет валидная ли связка логин/пароль
7. Клиент запрашивает у пользователя токен для аутентификации
8. Клиент берет токен из программы, на которой показывается токен и отсылает его на сервер
9. Сервер принимает токен, проверяет является ли токен валидным и отвечает клиенту

Алгоритм TOTP основан на синхронизированном времени программы токена и программы сервера. Сам код, который необходимо ввести генерируется с помощью библиотеки pyotp - как на сервере, так и в программе токена.

КОД

Код файла client.py

```
import socket

client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('localhost', 8089))

data = client.recv(2048)
print(data.decode('utf8'))

authorization_successful = False
authentication_successful = False

print('Starting authorization process')
while not authorization_successful:
    client.send(input('Input login: ').encode('utf8'))
    client.send(input('Input password: ').encode('utf8'))
    if client.recv(2048).decode('utf8') == 'Authorization success':
        authorization_successful = True
        print('Authorization successful')
    else:
        print('Authorization failed. Check login/password')

print('Starting authentication process')
while not authentication_successful:
    client.send(input("Input token: ").encode('utf8'))
    client.send('datetime.now()'.encode('utf8'))
    if client.recv(2048).decode('utf8') == 'Authentication success':
        print('Authentication successful!')
        authentication_successful = True
    else:
        print('Authentication failed')
print('You logged in!')
```

Код файла server.py

```
import socket
import pyotp


# Функция считывает данные о пользователях из текстового файла
def preparation():
    with open("credentials.txt") as file:
        lines = file.readlines()
        lines = [line.rstrip() for line in lines]
    acc = [(x[0], x[1]) for x in [line.split(' ') for line in lines]]
    return acc


server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('localhost', 8089))
server.listen(5)
print('Server running . . .')
accounts = preparation()

while True:
    connection, address = server.accept()
    connection.send('Connected successfully!'.encode('utf8'))
    print(f'Connection {address} successful!')
```

```
    authorization_successful = False
    authentication_successful = False

    print(f'Starting authorization process for {address}')
    while not authorization_successful:
        login = connection.recv(2048).decode('utf8')
        password = connection.recv(2048).decode('utf8')
        if (login, password) in accounts:
            authorization_successful = True
            connection.send('Authorization success'.encode('utf8'))
            print(f'{address} authorized')
        else:
            connection.send('Authorization failed'.encode('utf8'))
            print(f'Failed to authorize {address}')

    print(f'Starting authentication process for {address}')
    while not authentication_successful:
        totp = pyotp.TOTP('base32secret3232')
        received_token = connection.recv(2048).decode('utf8')
        valid_token = totp.now()
        print(f'Current token: {totp.now()}')
        print(f'Received token: {received_token}')
        if received_token == valid_token:
            authentication_successful = True
            connection.send('Authentication success'.encode('utf8'))
            print(f'Authentication complete for {address}')
        else:
            connection.send('Authentication failed'.encode('utf8'))
            print(f'Authentication failed for {address}')
```

Код файла token.py

```
import time
import pyotp


while True:
    totp = pyotp.TOTP('base32secret3232')
    print(f'Your code: {totp.now()}')
    time.sleep(5)
```

СКРИНШОТЫ И ССЫЛКА НА ВИДЕО

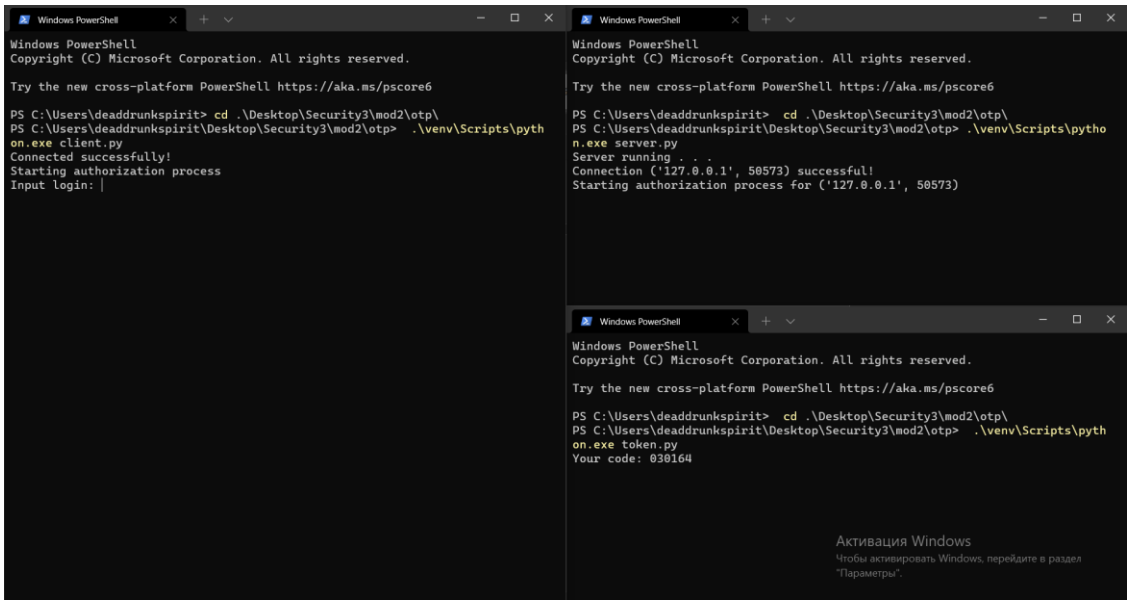


Рисунок 1. Запуск программы

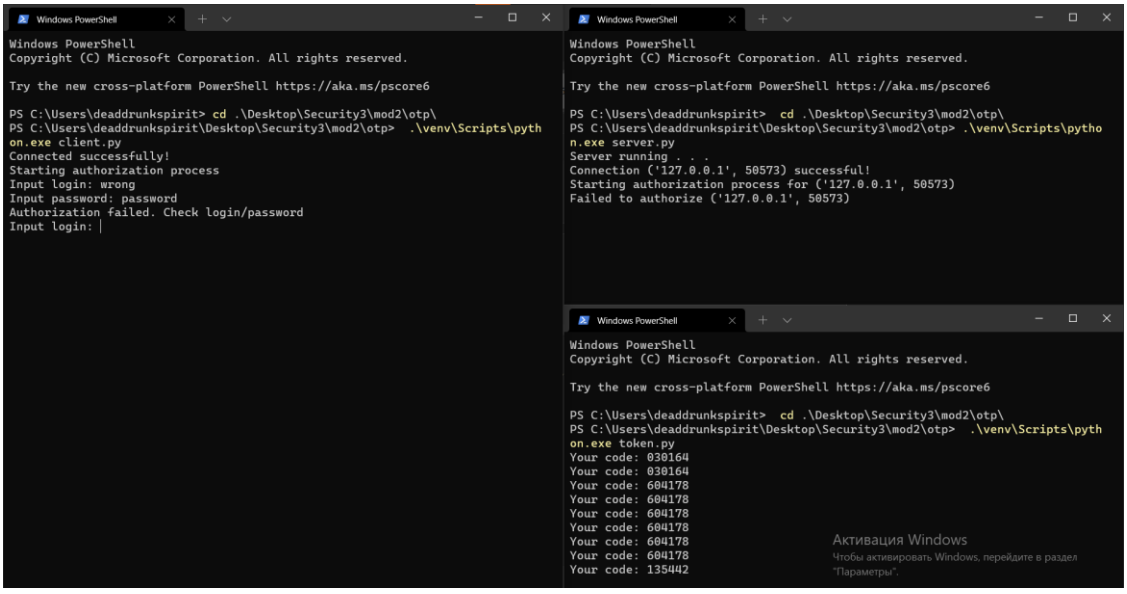
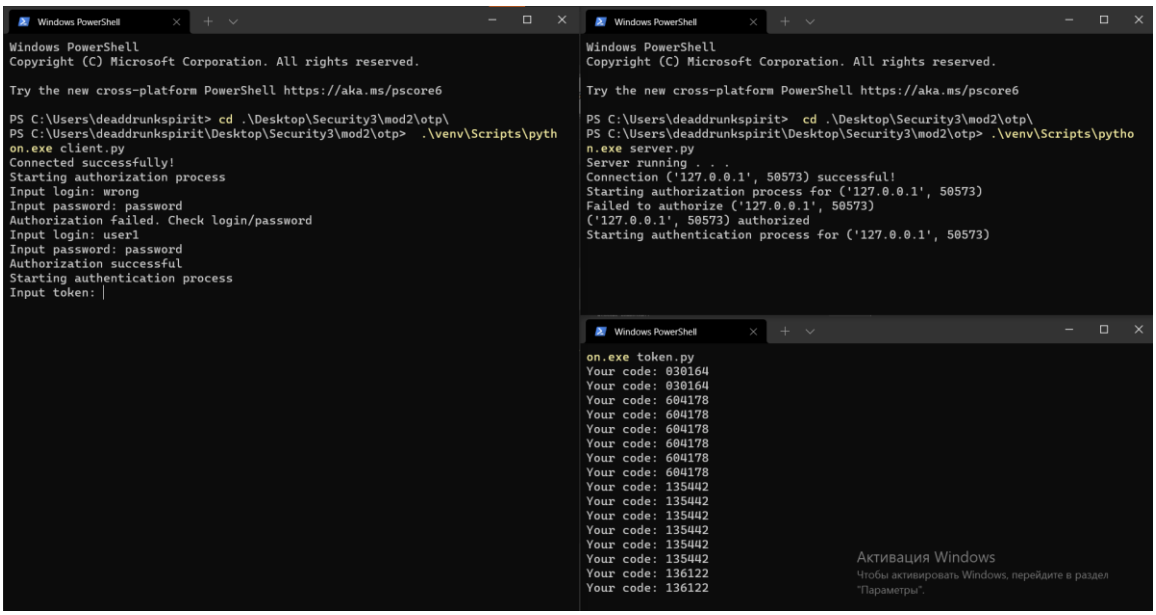
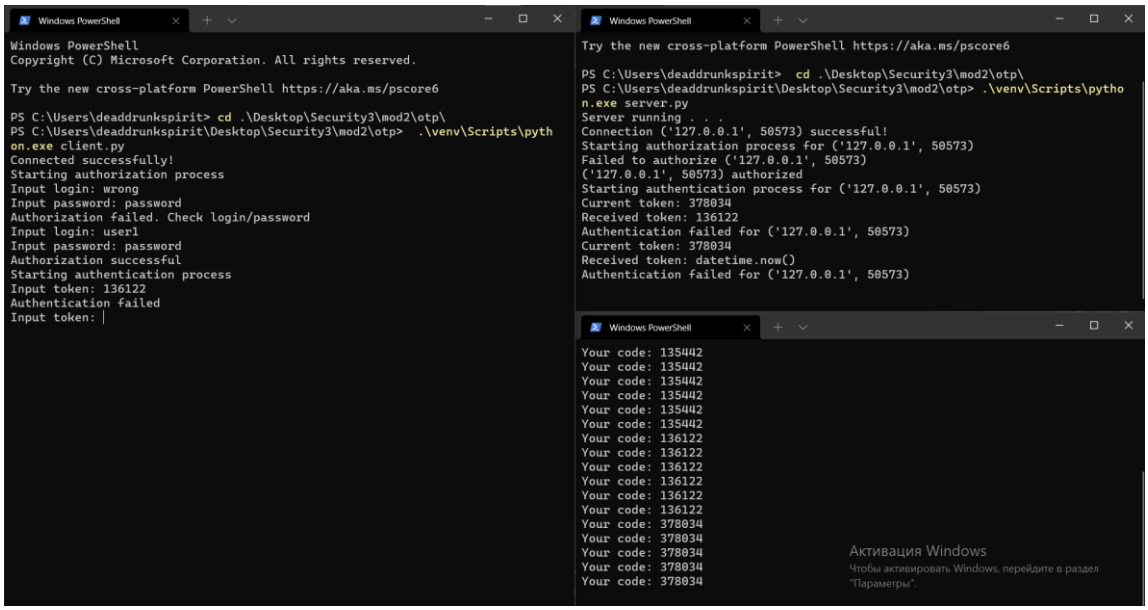


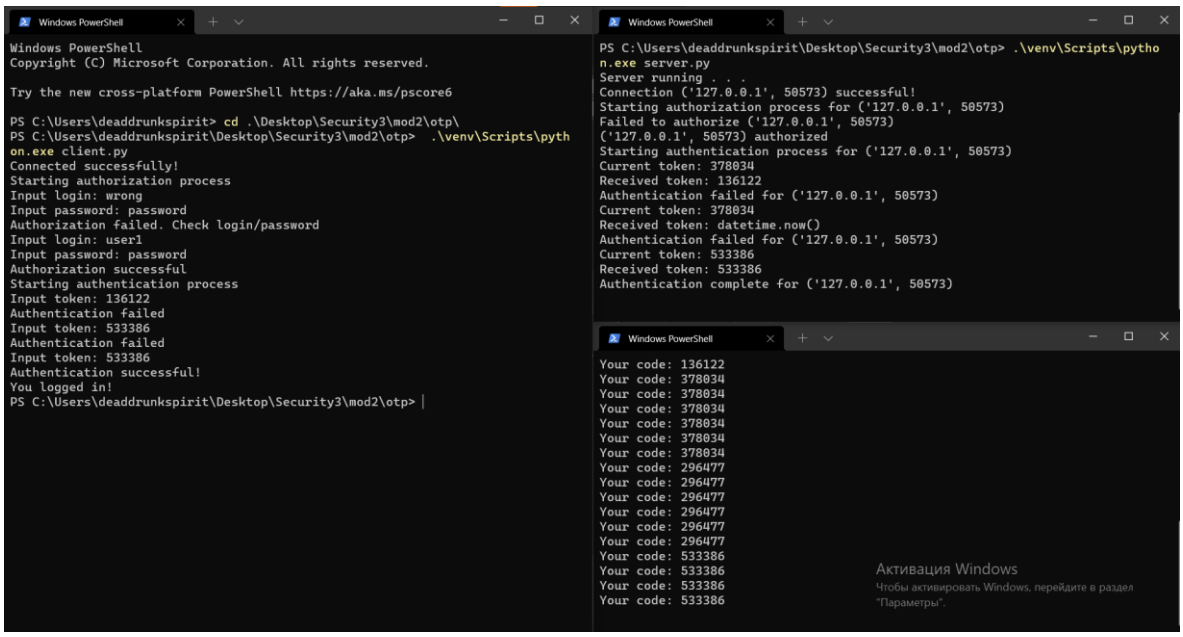
Рисунок 2. Ввели неверные данные



3. Ввели корректные логин/пароль



4. Ввели не валидный токен



5. Ввели валидный токен

Ссылка на видео с демонстрацией работы программы: https://drive.google.com/file/d/1-tZwaLUMZ-NpR4p3k2lHjS69eEBZtHPD/view?usp=sharing

ВЫВОД

Я изучил алгоритмы работы OTP и реализовал алгоритм TOTP.