**Информация для разработчика.**

Данное приложение состоит из 2 файлов Server\_gui.exe и Client\_gui.exe которые запускаются на разных компьютерах, для передачи информации используется беспроводная Wifi сеть. Оба компьютера должны быть подключены к одной сети.

1. Используемые библиотеки, модули
2. from itertools import tee #Импортируем все необходимые модули
3. from typing import Text
4. import PySimpleGUI as sg
5. from simplecrypt import encrypt, decrypt
6. import socket
7. from Crypto import Random
8. import netifaces
9. from Crypto.Signature import PKCS1\_v1\_5
10. from Crypto.Hash import SHA
11. from Crypto.PublicKey import RSA
12. from Crypto.Cipher import PKCS1\_OAEP
13. from Crypto.Cipher import AES
14. import time

* Для работы приложения импортируются различные модули.
* **PysimpleGUI** – модуль для работы с графическим интерфейсом
* **Socket** – модуль включает в себя функции создания объекта сокета [Socket](https://docs-python.ru/standart-library/modul-socket-setevoj-interfejs-python/obekt-socket-modulja-socket/), который и обрабатывает канал данных, а также функции, связанных с сетевыми задачами, такими как преобразование имени сервера в IP адрес и форматирование данных для отправки по сети.
* **Crypto** – модуль для шифрования

**Для добавления различных сторонних модулей используйте “import”. Пример “** import socket **”**

2) Создание графического интерфейса

layout = [  #Графические интерфейс

    [sg.Text('Выбор роли', size=(15, 1)), sg.InputCombo(('Alice', 'Bob'), size=(70, 3))],

    [sg.Text('Выбор протокола', size=(15, 1)), sg.InputCombo(("Привзяка к биту на основе симметричной криптографии","Протокол подбрасывания монеты на основе однонаправленной функции","Экспоненциальный протокол подбрасывнания монеты","ЭЦП"), size=(70, 3))],

    [sg.Submit(), sg.Cancel()],

    [sg.Output(size=(88, 20))],

    [sg.Text('Поле ввода: '), sg.InputText(size=(71, 3)),sg.Submit('Ввод')]

        ]

Для добавления протокола в поле выбора протокола используете sg.InputCombo() Протоколы добавляются через запятую

3) Выбор протокола. После получения данных о роли и протоколе. С помощью операторов if-else выбирается определенная функция. С помощью операторов if-else выбирается определенная функция, для каждой связки роли и протокола есть отдельная функция, где и выполняется код самого протокола

                if protokol =='Привзяка к биту на основе симметричной криптографии':

                        if role == 'Alice':

                            Protocol\_privazki\_k\_bity\_Alice()

                        elif role == 'Bob':

                            Protocol\_privazki\_k\_bity\_Bob()

                elif protokol =='Протокол подбрасывания монеты на основе однонаправленной функции':

                        if role == 'Alice':

                            Protocol\_podbrasivaniy\_monety\_Alice()

                        elif role == 'Bob':

                            Protocol\_podbrasivaniy\_monety\_Bob()

                elif protokol =='Экспоненциальный протокол подбрасывнания монеты':

                        if role == 'Alice':

                            Protocol\_exponent\_podbrasivaniy\_monety\_Alice()

                        elif role == 'Bob':

                            Protocol\_exponent\_podbrasivaniy\_monety\_Bob()

                elif protokol =='ЭЦП':

                        if role == 'Alice':

                            Protocol\_ECP\_Alice()

                        elif role == 'Bob':

                            Protocol\_ECP\_Bob()

Для добавления протокола используйте данную модель. Пример: необходимо добавить протокол.

elif protokol =='Название протокола':

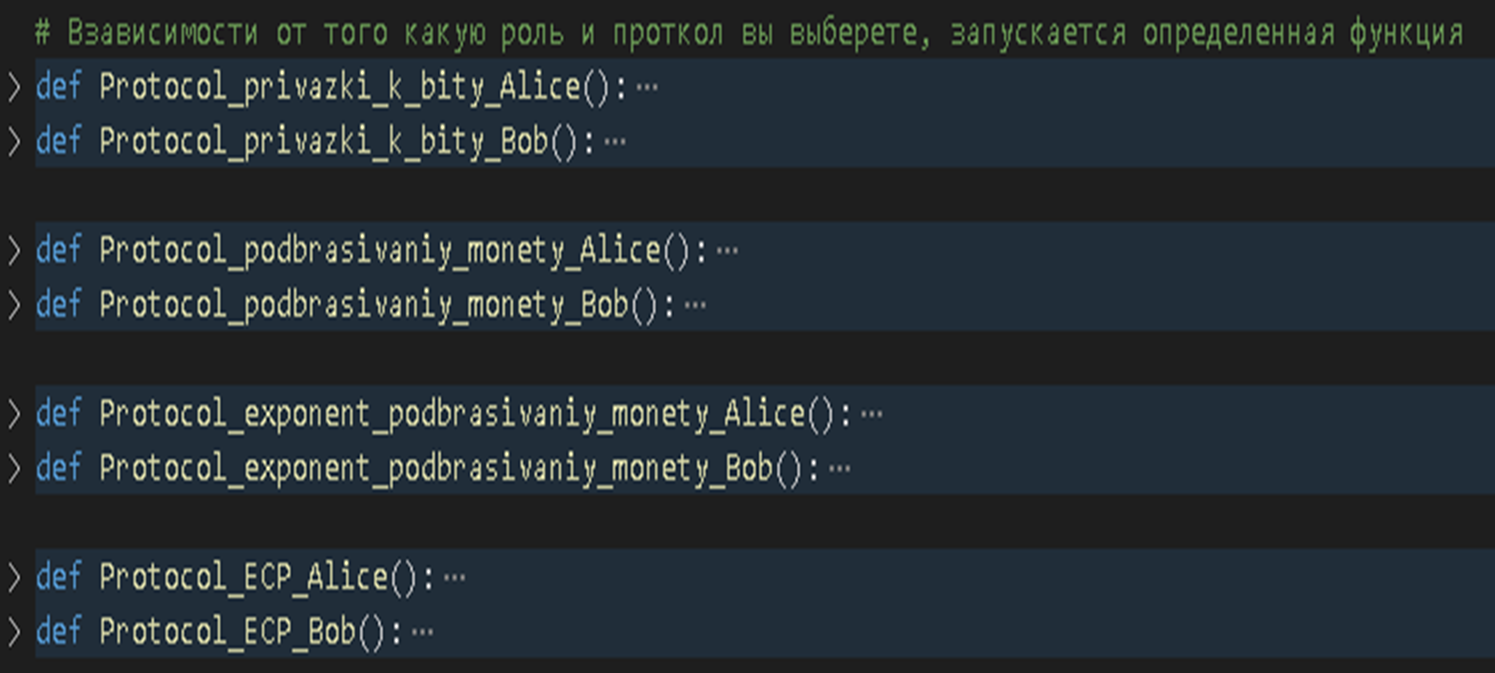
                        if role == 'Alice':

                            Функция\_данного\_протокола\_для\_Alice()

                        elif role == 'Bob':

                            Функция\_данного\_протокола\_для\_Bob()

4) В зависимости от того какую роль и протокол вы выберете, запускается определенная функция



В самой функции выполняется код предназначенный для каждого протокола и роли.

Пример

def Protocol\_privazki\_k\_bity\_Alice():

    #Сначала отправляем данные о протоколе и роле 2 участнику

    protocol = protokol.encode()

    roll = role.encode()

    connection.sendall(protocol)

    connection.sendall(roll)

    data = connection.recv(1024)

    data = data.decode()

    print(f"Bob выбрал биты : {data}")

    bit\_from\_bob=data

    text = 0

    print("Alice выбирает сообщение")

    event, values = window.read()

    text = values[2]

    print(f"Битовый текст Alice: {text}")

    r\_v=text+bit\_from\_bob

    print(f"Текст Alice + Последовательность битов выбранная Bob: {r\_v}")

    print("Шифрование AES-256. Введите ключ: ")

    event, values = window.read()

    key = values[2]

    print("Шифрование данных...")

    r\_and\_v\_crypt= encrypt(key, r\_v)

    print(f"Зашифрованное сообщение: {r\_and\_v\_crypt}")

    print("Отправляем зашифрованное сообщение Bob")

    connection.sendall(r\_and\_v\_crypt)

    print("Отправляем ключ шифрования")

    key=key.encode()

    connection.sendall(key)

    print("Конец \n\n")

Протокол привязки к биту, роль Alice;

5)Перед запуском функции идет отправка данных о протоколе и роли второму участнику.

В зависимости от роли, которую мы выбрали 2 участник автоматически выбирает противоположную роль.

Участник 1

#Сначала отправляем данные о протоколе и роле 2 участнику

    protocol = protokol.encode()

    roll = role.encode()

    connection.sendall(protocol)

    connection.sendall(roll)

Участник 2

print("Второй человек выбирает протокол и ваши роли в нем \n\n")

            protocol = connection.recv(1024).decode() # получаем данные о протоколе и роли

            roll = connection.recv(1024).decode()

**Дополнительно**

1. **Создание сокетов.**

Для обоих участников создаются сокеты

sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)# Создаем сокет

Сокет - Название программного интерфейса для обеспечения обмена данными между процессами, как раз с помощью сокетов происходит обмен информацией между участником 1 и участником 2.

1. **Запуск сервера для приема соединений.**
2. netifaces.gateways()# Берем ip своего безпроводного интерфеса
3. iface = netifaces.gateways()['default'][netifaces.AF\_INET][1]
4. ip = netifaces.ifaddresses(iface)[netifaces.AF\_INET][0]['addr']
5. server\_address = (ip, 10000)
6. print('Старт сервера на {} порт {}'.format(\*server\_address))
7. sock.bind(server\_address)
8. sock.listen(1) # Слушаем соединения
9. print('Ожидание соединения...')
10. connection, client\_address = sock.accept()

С помощью данного код участник 1 извлекает ip своего беспроводного интерфейса .запускает сервер на данном ip и порту 10000 и ожидает соединения.

3) **Подключение участника 2**

ip = values[0]

server\_address = (ip, 10000) # Подключаемся

print('Подключено к {} порт {}'.format(\*server\_address))

connection.connect(server\_address)

Участник 2 подключается к Участнику 1