Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

Лабораторная работа №5

По дисциплине «КМЗИ»

Тема: “Криптографические протоколы тайного голосования”

Выполнил:

Студент 3 курса

Группы ИИ-21

Кирилович А. А.

Проверила:

Хацкевич А.С.

Брест 2023

Цель:Создать заданный протокол тайного голосования

Вариант: Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Код программы:

import CIK, Voter, CSK

print("ЦИК создает и публикует список всех избирателей...")

CIK.generate\_voter\_list(20)  #ЦИК публикует список всех правомочных избирателей

print("ЦИК создает пару ключей...")

CIK.generate\_key\_pair() #ЦИК создает пару ключей для асимметричного шифрования

print("ЦИК публикует открытый ключ в ЦСК...") #это происходит в CSK.py

CSK.get\_cik\_key()

print("Избиратели публикуют свои ID и ключ в ЦСК...")

Voter.generate\_specific\_voters\_data\_list()  #создается список с нужной инфой о изберателях(ID, ключи, метки и тд)

CSK.get\_voters\_info()

print("Избиратели генерируют свою уникальную метку...") #происходит в voter.py

print("Избиратели ослепляют и подписывают свои метки и отправляют в ЦИК...")

Voter.blinding\_protocol()

print("ЦИК проверяет ЭЦП избирателей и отправляет избирателям слепую ЭЦП...")

CIK.check\_signature()

print("Избиратели снимают закрывающий множитель с ЭЦП комиссии...")

Voter.remove\_r()

print("Избиратели голосуют...")  #это происходит сразу в remove\_r

print("Отправка анонимных результатов в ЦИК...") #также в remove\_r

print("ЦИК проверяет данные и публикует метку и бюллитень и выводит результат")

CIK.verify\_and\_results()

Код «ослепления» и подписи уникальной метки M:

def blinding\_protocol():

    global voter\_with\_id\_blind\_and\_sign

    #используем уникальный закрывающий множитель r, открытый ключ из ЦИК для скрытия метки M

    for voter in voters\_info\_list:

        blinded\_message = (voter['r'] \*\* CIK.public\_key.e \* voter['M']) % CIK.public\_key.n

        #перевод в байты для хэширования

        blinded\_message\_bytes = blinded\_message.to\_bytes((blinded\_message.bit\_length() + 7) // 8, 'big')

        #хэшируем полученное сообщение

        hashed\_blinded\_message = SHA1.new(blinded\_message\_bytes)

        #подписываем захэшированное сообщение с закрытым ключом каждого избирателя

        signatured\_message = PKCS115\_SigScheme(voter['private\_key']).sign(hashed\_blinded\_message)

        voter['blind\_m'] = hashed\_blinded\_message

        voter['sign\_m'] = signatured\_message

    #создаем список с полученными данными

    voter\_with\_id\_blind\_and\_sign = [{'id': voter['id'], 'blind\_m': voter['blind\_m'], 'public\_key': voter['public\_key'], 'sign\_m': voter['sign\_m'], 'private\_key': voter['private\_key'], 'r': voter['r'], 'M': voter['M']} for voter in voters\_info\_list]

Код проверки подписи избирателей и создания анонимной подписи:

def check\_signature():

    global voter\_with\_id\_blind\_and\_sign\_from\_cik

    for voter in Voter.voter\_with\_id\_blind\_and\_sign:

        try:

            PKCS115\_SigScheme(voter['public\_key']).verify(voter['blind\_m'], voter['sign\_m'])

            print(f"Подпись избирателя с ID: {voter['id']} верна.")

            signatured\_message\_from\_cik = PKCS115\_SigScheme(private\_key).sign(voter['blind\_m'])

            voter['sign\_m\_from\_cik'] = signatured\_message\_from\_cik

        except:

            print(f"Подпись избирателя с ID: {voter['id']} не верна.")

    voter\_with\_id\_blind\_and\_sign\_from\_cik = [{'id': voter['id'], 'blind\_m': voter['blind\_m'], 'public\_key': voter['public\_key'], 'sign\_m\_from\_cik': voter['sign\_m\_from\_cik'], 'r': voter['r'], 'M': voter['M']} for voter in Voter.voter\_with\_id\_blind\_and\_sign]

Код снятия «закрытого множителя» r:

def remove\_r():

    global anonim\_list

    for voter in CIK.voter\_with\_id\_blind\_and\_sign\_from\_cik:

        hashed\_message\_from\_cik = SHA1.new()

        hashed\_message\_from\_cik.update(voter['sign\_m\_from\_cik'])

        #вычисление обратного значения r

        r\_inv = pow(voter['r'], -1, CIK.public\_key.n)

        #ЭЦП комисии после снятия закрытого множителя r

        DS\_int = int.from\_bytes(voter['sign\_m\_from\_cik'], byteorder='big')

        DS = (DS\_int \* r\_inv) % CIK.public\_key.n

        voter['DS'] = DS

        vote = random.choice(bulliten)

        voter['vote'] = vote

    #список с данными для отправки

    anonim\_list = [{'M': voter['M'], 'vote': voter['vote'], 'DS': voter['DS']} for voter in CIK.voter\_with\_id\_blind\_and\_sign\_from\_cik]

Вывод: изучил и реализовал на практике протокол тайного голосования.