МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

Схема разделения секрета Карнина-Грина-Хеллмана

ОТЧЁТ
ПО ДИСЦИПЛИНЕ
«КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ»

студента 5 курса 531 группы
специальности 10.05.01 Компьютерная безопасность
факультета компьютерных наук и информационных технологий
Норикова Павла Сергеевича

Преподаватель
аспирант _____ Р. А. Фарахутдинов
подпись, дата

Саратов 2023

ВВЕДЕНИЕ

Цель работы – изучение и реализация схемы разделения секрета Карнина-Грина-Хеллмана.

1 Теория

Разделение секрета — термин в криптографии, под которым понимают любой из способов распределения секрета среди группы участников, каждому из которых достаётся своя некая доля. Секрет может воссоздать только коалиция участников из первоначальной группы, причём входить в коалицию должно не менее некоторого изначально известного их числа.

1.1 Описание алгоритма

Выбирается $n + 1$ таких t-мерных векторов $V_0, V_1, \dots, V_n$, что ранг любой матрицы размером $t \times t$, образованной из этих векторов, равен t. Вектор U – это вектор размерности t.

Секрет M – это матричное произведение UV_0^T. Долями секрета являются произведения $\alpha_i = UV_i^T$, где i меняется от 1 до n.

Любые m долей можно использовать для решения системы линейных уравнений размерности $m \times m$, неизвестными являются коэффициенты U. V_0 известно всем, остальные V_i известны i-м участникам.

Восстановление секрета происходит с помощью решения СЛУ из t уравнений и t неизвестных относительно компонент вектора U:

$$\begin{cases} U^1V_1^1 + U^2V_1^2 + \dots + U^tV_1^t = \alpha_1 \\ \dots \\ U^1V_t^1 + U^2V_t^2 + \dots + U^tV_t^t = \alpha_t \end{cases}$$

Получив решение U_{res}, нужно вычислить $U_{res}V_0^T$. Это значение и будет секретом M.

2 Практическая реализация

2.1 Описание программы

Функция *gen_v* при помощи функции *gram_schmidt* генерирует ЛНЗ векторы.

На вход программе подается количество участников n и минимальное количество участников для раскрытия секрета m.

2.2 Тестирование программы

```
Введите количество участников. n = 10
Введите минимальное количество участников для раскрытия секрета. m = 5
U: [82 90 49 27 56]

V0: [68 27 38 46 25]
V1: [  5  -3  57 -27  53]
V2: [-18  51 -31   3  63]
V3: [23 55 34 23 74]
V4: [ 36  33  31 -65  50]
V5: [11 11  5 13  7]
V6: [-12  15  49  -8  42]
V7: [-31   1  32  -2  57]
V8: [  7  35  53 -29  11]
V9: [  0 -10  45 -30  43]
V10: [-4 28 19 -8 31]


A1: 5172
A2: 5204
A3: 13267
A4: 8486
A5: 2880
A6: 4903
A7: 2254
A8: 6154
A9: 2903
A10: 4643

Секрет: M = 12510

5 случайных участников: [2, 8, 1, 6, 5]

Полученная СЛУ:
68 27 38 46 25 5172
5 -3 57 -27 53 5204
-18 51 -31 3 63 13267
23 55 34 23 74 8486
36 33 31 -65 50 2880

Решение СЛУ: [82, 90, 49, 27, 56]

Вычисленный секрет: 12510
```

Рисунок 1 - Работа протокола

ПРИЛОЖЕНИЕ А

Листинг программы

```
import numpy as np
import random as rand

def gram_schmidt(vectors):
    basis = []
    for vector in vectors:
        for existing_vector in basis:
            vector -= np.dot(vector, existing_vector) //
np.dot(existing_vector, existing_vector) * existing_vector
        basis.append(vector)
    return basis

def gen_v(n, t, integer_range=(1, 100)):
    random_vectors = []

    while len(random_vectors) < n + 2:
        random_vector = np.random.randint(integer_range[0],
integer_range[1] + 1, size=t)
        random_vectors.append(random_vector)
        random_vectors = gram_schmidt(random_vectors)

    return random_vectors


def main():
    n = int(input('Введите количество участников. n = '))
    m = int(input('Введите минимальное количество участников для
раскрытия секрета. m = '))
    V = gen_v(n, m)
    U = V[0]
    V = V[1:]

    print('U:', U, '\n')
    for i, vector in enumerate(V):
        print(f"V{i}: {vector}")

    print('\n')
    A_i = []
    for i in range(1, len(V)):
        A_i.append(np.dot(U, V[i]))
        print(f"A{i}: {A_i[i-1]}")


    M = np.matmul(U, V[0])
    print('\nСекрет: M =', M, '\n')

    all_members = list(range(1, n))
    members = []
    matrix = []
    vector = []
    for _ in range(m):
        x = rand.choice(all_members)
        matrix.append(V[x])
```

```
            vector.append(A_i[x-1])
            members.append(x)
            all_members.remove(x)

        print(m, 'случайных участников:', members, '\n')
        print('Полученная СЛУ:')
        for i in range(m):
            for j in range(m):
                print(V[i][j], end = ' ')
            print(A_i[i])


        res = np.linalg.solve(matrix, vector)
        #print(res)
        res = [round(x) for x in res]
        print('\nРешение СЛУ:', res)
        UV0 = round(np.matmul(res, V[0]))

        print('\nВычисленный секрет:', UV0)


main()
```