МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КУБГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**Отчет**

**по практическому заданию №10**

**по курсу**

**«КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ»**

Работу выполнил

Студенты 46 группы

Нагалевский А.М.

Преподаватель:

Крамаренко А.А.

Краснодар 2024

**Постановка задачи.**

Рюкзачная криптосистема (Knapsack cryptosystem) — это один из методов асимметричного шифрования, основанный на комбинаторной задаче рюкзака. Она была предложена в 1978 году Ральфом Мерклом и Мартином Хеллманом.

Основная идея рюкзачной криптосистемы заключается в следующем:

**1. Генерация открытого ключа:**

- Задаётся секретный рюкзак (вектор из целых чисел), который должен быть супервозрастающим (каждый следующий элемент больше суммы всех предыдущих).

- Задаётся набор открытых чисел (множество супервозрастающих чисел), которые являются общими делителями секретного ключа. Эти числа формируют открытый ключ.

**2. Шифрование сообщения:**

- Сообщение представляется в виде последовательности битов.

- Для каждого бита сообщения выбирается соответствующий открытый ключ из набора так, чтобы сумма выбранных открытых ключей была равна значению бита.

- Полученные открытые ключи объединяются в один шифртекст.

**3. Расшифрование сообщения:**

- Используя закрытый рюкзак, который является инверсией секретного рюкзака, происходит вычисление суммы произведений выбранных открытых ключей на закрытые значения.

- Если эта сумма равна значению бита сообщения, то бит расшифровывается как 1, иначе - как 0.

Преимущества и недостатки рюкзачной криптосистемы:

**Преимущества:**

- Возможность зашифрования большого объема данных с использованием небольшого ключа.

- Относительно простое шифрование и расшифрование.

- Возможность доказательства безопасности схемы при определенных условиях выбора параметров.

**Недостатки:**

- Неустойчивость к атакам, таким как известный текст и адаптивный шифртекст.

- Зависимость от выбора подходящих параметров (например, свойства секретного рюкзака).

- Существует алгоритмический метод решения задачи о рюкзаке, который может использоваться для взлома криптосистемы в некоторых случаях.

Рюкзачная криптосистема была одной из первых попыток создания асимметричной криптографии, однако её применение ограничено из-за своих недостатков. В настоящее время она редко используется в практических криптографических приложениях.

**Текст программы:**

**Файл task1.py:**

from cryptography.hazmat.primitives import serialization, hashes

from cryptography.hazmat.primitives.asymmetric import rsa

from cryptography.hazmat.backends import default\_backend

from cryptography.hazmat.primitives.asymmetric import padding

def generate\_key\_pair():

private\_key = rsa.generate\_private\_key(

public\_exponent=65537,

key\_size=2048,

backend=default\_backend()

)

public\_key = private\_key.public\_key()

return private\_key, public\_key

def save\_private\_key(private\_key, filename):

with open(filename, "wb") as key\_file:

key\_file.write(private\_key.private\_bytes(

encoding=serialization.Encoding.PEM,

format=serialization.PrivateFormat.TraditionalOpenSSL,

encryption\_algorithm=serialization.NoEncryption()

))

def load\_private\_key(filename):

with open(filename, "rb") as key\_file:

private\_key = serialization.load\_pem\_private\_key(

key\_file.read(),

password=None,

backend=default\_backend()

)

return private\_key

def encrypt\_message(public\_key, message):

ciphertext = public\_key.encrypt(

message.encode(),

padding.OAEP(

mgf=padding.MGF1(algorithm=hashes.SHA256()), # маскирующая функция генерации

algorithm=hashes.SHA256(), # хэш функция

label=None # используется для передачи аутентификационных данных

)

)

return ciphertext

def decrypt\_message(private\_key, ciphertext):

plaintext = private\_key.decrypt(

ciphertext,

padding.OAEP(

mgf=padding.MGF1(algorithm=hashes.SHA256()),

algorithm=hashes.SHA256(),

label=None

)

)

return plaintext.decode()

# Генерация ключевой пары

private\_key, public\_key = generate\_key\_pair()

# Сохранение закрытого ключа в файл

save\_private\_key(private\_key, "private\_key.pem")

# Загрузка закрытого ключа из файла

loaded\_private\_key = load\_private\_key("private\_key.pem")

# Шифрование и расшифрование сообщения

message = input("Введите сообщение для шифрования: ")

encrypted\_message = encrypt\_message(public\_key, message)

print("Зашифрованное сообщение:", encrypted\_message)

decrypted\_message = decrypt\_message(loaded\_private\_key, encrypted\_message)

print("Расшифрованное сообщение:", decrypted\_message)