Министерство науки и высшего образования Российской Федерации

Санкт-Петербургский Политехнический Университет Петра Великого

—

Институт компьютерных наук и кибербезопасности

Высшая школа кибербезопасности

**ЛАБОРАТОРНАЯ РАБОТА № 4**

**«Криптосистемы на основе задачи дискретного логарифмирования»**

по дисциплине «Криптографические методы защиты информации»

Выполнил

студент гр. 45151004/90101 Кондачков Е.Д.

Преподаватель

Ассистент Зубков Е.А.

Санкт-Петербург

2023

Содержание

# Цель работы

Изучение протоколов шифрования с открытым ключом и электронной подписи, безопасность которых основана на задаче дискретного логарифмирования в конечном поле.

# Задачи работы

1. Согласно полученному варианту задания разработать криптосистему на основе задачи дискретного логарифмирования;
2. Сравнить разработанную систему с RSA, выявить преимущества и недостатки.

# Ход работы

В ходе работы была реализована криптосистема по протоколу Диффи-Хеллмана, которая основывается на передаче пользователей друг другу частей ключа.

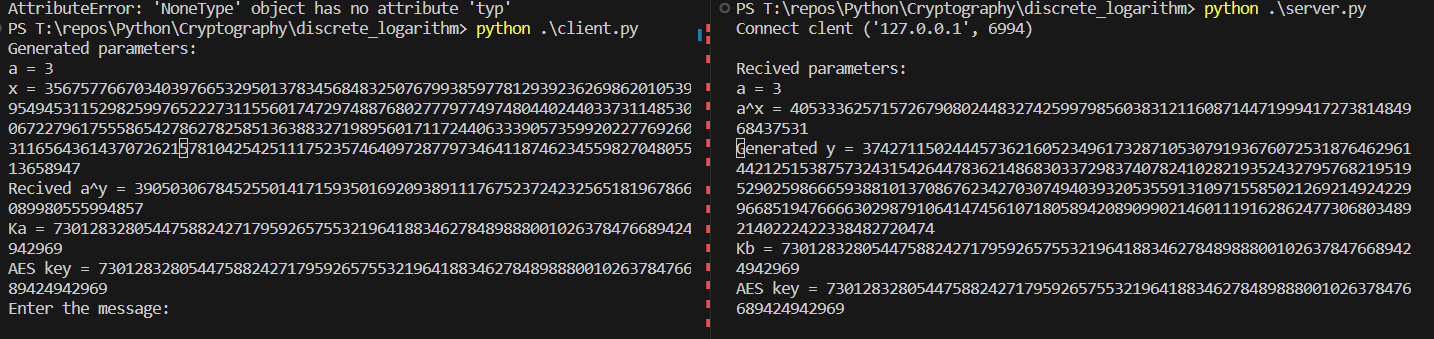


Рисунок – Работа алгоритма

На рисунке выше показано, как алгоритм производит обмен ключами:

1. Клиент генерирует случайный показатель *x* и считывает показатель *a*;
2. Клиент отправляет серверу ASN1 файл, содержащий *а* и *ах*;
3. Сервер генерирует свой закрытый показатель *y*;
4. Сервер расшифровывает ASN1 файл, берет оттуда *a* и отправляет клиенту ay;
5. Клиент (сервер), возводит ax(ay) в степень y(x), получая таким образом ключ.

Контейнер с сообщением в зашифрованномм виде выглядит следующим образом.

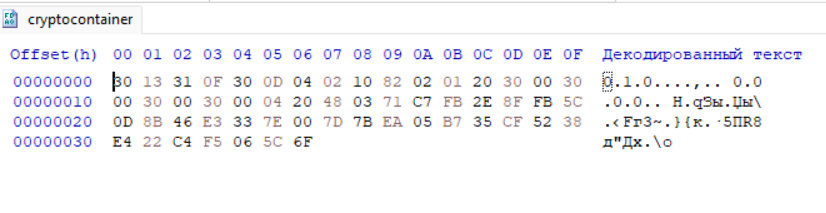


Рисунок – Контейнер с зашифрованным сообщением

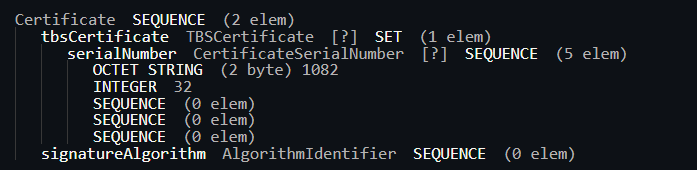


Рисунок – Расшифрованный контейнер

# Ответы на контрольные вопросы

1. Пусть 𝐺 – конечная циклическая группа с заданной образующей. Порядок группы 𝐺 известен и делится на простое число 𝑟. Как найти образующую циклической подгруппы порядка 𝑟?
2. Покажите, что задача Месси–Омуры сводится к задаче дискретного логарифмирования.

Дано: . Найти: t. Введем обозначения: 𝑟.

Задача определения по известным s и r как раз является задачей дискретного логарифмирования.

1. Подвержен ли протокол шифрования Эль-Гамаля к атаке на основе подобранных шифртекстов?

Такая атака заключается в сборе криптоаналитиком информации о шифре путем подбора шифртекстов и получения некоторых их расшифровок без известной информации о ключе. При сопоставлении ставших известными блоков исходного текста и его шифртекстов криптоаналитик может восстановить секретный ключ. При этом данной атаке могут быть подвержены шифр по схеме Эль-Гамаля, RSA.

1. Перечислите задачи, положенные в основу схемы подписи Эль-Гамаля. Почему важно, чтобы период генератора случайных чисел, используемого в схеме подписи, был достаточно большим?

* Увеличение криптостойкости при подписании;
* Криптостойкость основана на вычислительной сложности проблемы дискретного логарифмирования;
* Возможность многократно подписывать различные сообщения одной электронной подписью (одним ключом);
* Схема Эль-Гамаля явилась прототипом множеству современных стандартов электронных подписей;
* Замена длинных чисел в операциях на остатки от деления от основы криптосистемы.

Период генератора случайных чисел, используемого в схеме подписи, должен быть достаточно большим, чтобы итоговое значение подписи не стало повторяться на разных сообщениях в период использования одного секретного ключа.

# Выводы по проделанной работе

В ходе данной лабораторной работы был разработана криптосистема, использующая протокол Диффи-Хеллмана. Данная криптосистема выгодно отличается от RSA своей простотой в установке ключа, так как для этого ей не требуется производить операцию логарифмирования, как, например, в первой лабораторной работе, где ключ шифровался посредством RSA. В то же время, данный алгоритм требует весьма трудоемкой операции возведения в степень, которая выполняется значительное время.

# Приложение А

Листинг client.py

#this is a client

from config import address, port, a

import socket

from asn import ASN

import random as rnd

from Crypto.Util.number import getPrime

from Crypto.Random import get\_random\_bytes

def connect\_server() ->socket.socket:

connection = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

connection.connect((address, port))

return connection

def get\_parameter(p) -> int:

return rnd.randint(1, p)

def exchange\_keys(connection:socket.socket) -> int:

p = getPrime(1024, randfunc=get\_random\_bytes)

x = get\_parameter(p)

asn = ASN.encrypt\_diffie\_hellman(a\*\*x, "client", p, a)

connection.send(asn)

asn = connection.recv(4096)

c = int(ASN.decrypt\_diffie\_hellman(asn))

print(f'Generated parameters:\na = {a}\nx = {x}')

print(f'Recived a^y = {c}')

key = c\*\*x

print(f'Ka = {key}')

return key

def main():

connection = connect\_server()

key = exchange\_keys(connection)

AESkey = key % (2\*\*256)

print(f'AES key = {AESkey}')

while(True):

message = input("Enter the message: ")

message\_bytes = bytes(message, 'utf-8')

message\_len = len(message\_bytes)

extension\_length = 16 - message\_len

for i in range (extension\_length):

message\_bytes += b'\x03'

asn = ASN.encrypt\_aes\_diffie\_hellman(message\_len, message\_bytes)

connection.send(asn)

with open("cryptocontainer", "wb") as enc:

enc.write(asn)

if message == 'exit':

break

connection.close()

if \_\_name\_\_ == '\_\_main\_\_':

main()

# Приложение Б

Листинг server.py

#this is a server

import socket

from asn import ASN

import random as rnd

def start\_server() -> socket.socket:

listener = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

listener.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)

listener.bind(('127.0.0.1', 8000))

listener.listen(0)

connection, address = listener.accept()

print(f'Connect clent {address}\n')

return connection

def get\_parameter(p) -> int:

return rnd.randint(1, p)

def exchange\_keys(connection:socket.socket) -> int:

asn = connection.recv(4096)

p, a, c = ASN.decrypt\_diffie\_hellman(asn, "client")

print(f'Recived parameters:\na = {a}\na^x = {c}')

y = get\_parameter(p)

print(f'Generated y = {y}')

asn = ASN.encrypt\_diffie\_hellman(a\*\*y)

connection.send(asn)

key = c\*\*y

print(f'Kb = {key}')

return key

def main():

connection = start\_server()

key = exchange\_keys(connection)

AESkey = key % (2\*\*256)

print(f'AES key = {AESkey}')

while(True):

asn = connection.recv(4096)

message\_bytes, len = ASN.decrypt\_aes\_diffie\_hellman(asn)

message\_bytes = message\_bytes[:len]

message = bytes.decode(message\_bytes)

print(message)

if message == 'exit':

break

connection.close()

if \_\_name\_\_ == '\_\_main\_\_':

main()