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

Элементы оглавления не найдены.

1. Цель работы

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

2. Задачи работы

1. Согласно полученному варианту задания разработать криптосистему на основе задачи дискретного логарифмирования;

2. Сравнить разработанную систему с RSA, выявить преимущества и недостатки.

3. Ход работы

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

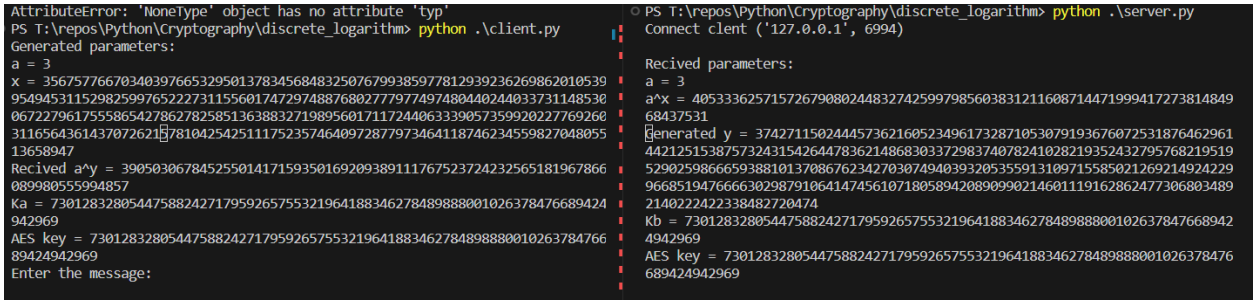

Рисунок 1 – Работа алгоритма

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 *a* и a^x;

3. Сервер генерирует свой закрытый показатель *y*;

4. Сервер расшифровывает ASN1 файл, берет оттуда *a* и отправляет клиенту a^y;

5. Клиент (сервер), возводит $a^x(a^y)$ в степень y(x), получая таким образом ключ.

Контейнер с сообщением в зашифрованномм виде выглядит следующим образом.

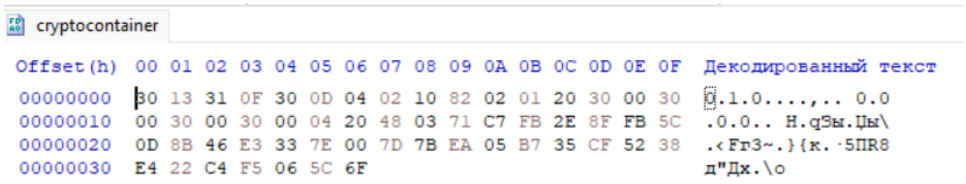

Рисунок 2 – Контейнер с зашифрованным сообщением

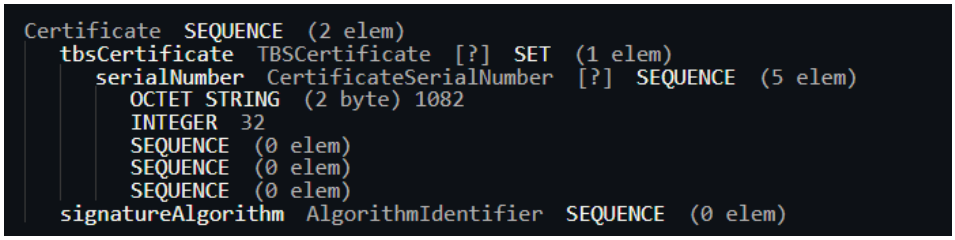

Рисунок 3 – Расшифрованный контейнер

4. 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?

$$r * \frac{ord(G)}{r}$$

2. Покажите, что задача Месси–Омуры сводится к задаче дискретного логарифмирования.

Дано: t^a, t^b, t^{ab}. Найти: t. Введем обозначения: $r = t^b,\ s = t^{ab}$.

$$s = r^a$$

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

3. 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.

4. 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.

5. 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()
```