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

Протокол Ньюмана-Стабблбайна

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 протокола обмена ключами Ньюмана-Стабблбайна.

1 Теория

Криптографический протокол Ньюмана-Стабблбайна для удостоверения подписи и обмена ключами был впервые опубликован в 1993 году. Протокол является модификацией протокола Yahalom и разработан в Массачусетском технологическом институте (MIT) Клифордом Ньюманом и Стюартом Стабблбайном.

Данный протокол является усовершенствованной версией протокола Yahalom. Особенностью протокола является отсутствие необходимости синхронизации часов у сторон, а также возможность повторной аутентификации без использования промежуточной стороны.

1.1 Описание алгоритма

1. $Alice \rightarrow \{A, R_A\} \rightarrow Bob.$ Алиса отправляет Бобу сообщение, содержащее идентификатор Алисы и некоторое случайное число Алисы.
2. $Bob \rightarrow \{B, R_B, E_B(A, R_A, T_B)\} \rightarrow Trent.$ Боб объединяет идентификатор Алисы, ее случайное число и метку времени, шифрует сообщение общим с Трентом ключом и посылает его Тренту, добавив свой идентификатор и случайное число Боба.
3. $Trent \rightarrow \{E_A(B, R_A, K, T_B), E_B(A, K, T_B), R_B\} \rightarrow Alice.$ Трент генерирует сеансовый ключ K, а затем создает два сообщения. Первое включает идентификатор Боба, случайное число Алисы, случайный сеансовый ключ, метку времени и шифруется общим для Трента и Алисы ключом. Второе состоит из идентификатора Алисы, сеансового ключа, метки времени и шифруется общим для Трента и Боба ключом. Трент добавляет к ним случайное число Боба и отправляет Алисе.
4. $Alice \rightarrow \{E_B(A, K, T_B), E_K(R_B)\} \rightarrow Bob.$ Алиса извлекает K и убеждается, что R_A совпадает с тем, что было послано на этапе 1. Алиса отправляет Бобу два сообщения. Первое - это второе

сообщение от Трента, зашифрованное ключом Боба. Второе - это случайное число Боба, зашифрованное сеансовым ключом.

Боб расшифровывает сообщение своим ключом и убеждается, что значения T_B и R_B не изменились.

Если оба случайных числа и метка времени совпадают, то Алиса и Боб убеждаются в подлинности друг друга и получают секретный ключ. Нет необходимости синхронизировать часы, так как метка времени определяется только по часам Боба и только Боб проверяет созданную им метку времени.

1.2 Проверка подлинности

1. $Alice \rightarrow \{E_B(A, K, T_B), R'_A\} \rightarrow Bob$. Алиса отправляет Бобу сообщение, присланное Трентом на этапе 3 и новое случайное число.
2. $Bob \rightarrow \{R'_B, E_K(R'_A)\} \rightarrow Alice$. Боб отправляет Алисе свое новое случайное число и случайное число Алисы, шифруя их сеансовым ключом.
3. $Alice \rightarrow \{E_K(R'_B)\} \rightarrow Bob$. Алиса отправляет Бобу его новое случайное число, зашифрованное сеансовым ключом.

2 Практическая реализация

2.1 Описание программы

Функции *encr* и *decr* осуществляют шифрование и расшифрование соответственно.

2.2 Тестирование программы

```
1. Алиса генерирует R_a = 8663773895
Алиса отсылает Бобу {A, R_a} = ('Alice', '8663773895')

2. Боб генерирует R_b = 9705820909
Метка времени Боба T_b = 1703630227.1812413
Боб отсылает Тренту {B, R_b, E_b(A, R_a, T_b)} = ('Bob', '9705820909', b'gAAAAABli1WT-FuiCjcbC
MWBSSHX9TljK')

3. Трент расшифровывает E_b(A, R_a, T_b)
A = Alice
R_a = 8663773895
T_b = 1703630227.1812413
Трент генерирует K = b'hEql8tPtRsjaYuYZ1LDc8HiNjyvp1EJx9n92zpVe62M='
Трент отсылает Алисе {E_a(B, R_a, K, T_b), E_b(A, K, T_b), R_b} = (b'gAAAAABli1WTiiEUssD4lo3xk
UogM-do2SECgYZ31-rFlpOMY0GZoo0NgYZSYX0Yj-stoiwf7Dyg=', b'gAAAAABli1WTj62EG_Z2bcNbyuoZUZ1deH5JU
xDLXLSkOoJ5di8Qd6ft-CeUWqL-juj8K1zSc9rY=', '9705820909')

4. Алиса расшифровывает E_a(B, R_a, K, T_b)
B = Bob
R_a = 8663773895
K = hEql8tPtRsjaYuYZ1LDc8HiNjyvp1EJx9n92zpVe62M=
T_b = 1703630227.1812413
Алиса отсылает Бобу {E_b(A, K, T_b), E_k(R_b)} = (b'gAAAAABli1WTj62EG_Z2bcNbyuoZUZ1deH5JUYsl_P
SkOoJ5di8Qd6ft-CeUWqL-juj8K1zSc9rY=', b'gAAAAABli1WTVDzfm8_L3OKYrluLmha4fIDr-bXCX6mP9jn04r_SBT

Боб расшифровывает E_b(A, K, T_b)
A = Alice
K = hEql8tPtRsjaYuYZ1LDc8HiNjyvp1EJx9n92zpVe62M=
T_b = 1703630227.1812413
Боб расшифровывает E_k(R_b)
R_b = 9705820909
```

Рисунок 1 - Работа протокола

```
ПРОВЕРКА ПОДЛИННОСТИ
1. Алиса генерирует R2_a = 723
Алиса отсылает Бобу {E_b(A, K, T_b), R2_a} = (b'gAAAAABli1WTj62EG_Z2bcNbyuoZUZ1deH5JUYsl_PPSi_
xQzd6v6mEcDC33rRDuxTxBqKiIGFW3Xeo3Ipz2DVwzRVl3Y5M1PrdULFNwj-bc1fp3GBjGdR6iku9VGtNvWOLEThLtUCDK
SSb8xDLXLSkOoJ5di8Qd6ft-CeUWqL-juj8K1zSc9rY=', '723')

2. Боб генерирует R2_b = 615
Боб отсылает Алисе {R2_b, E_k(R2_a)} = ('615', b'gAAAAABli1WTe0FYWE2ByEY8yQhrpIyoBi3lBa029EAWC
K1RrvT_bDqmH8aEv7pfyBtpLL45hPDyJ6xTNh_BaXIkWxl5EEib2g==')

3. Алиса расшифровывает E_k(R2_a)
R2_a = 723
Алиса отсылает Бобу {E_k(R2_b)} = b'gAAAAABli1WTsjNnB021dXVg4PIt4_ynFpLNaNpL4VWap7IAHIPBso23R_
UtVI_Ji985ReTYrtRw7hqF24V5CqNbUTPRAq8IEA=='

Боб расшифровывает E_k(R2_b)
R2_b = 615
```

Рисунок 2 – Проверка подлинности

ПРИЛОЖЕНИЕ А

Листинг программы

```
from cryptography.fernet import Fernet
import time
import random

def encr(key, x):
    data = ''
    for i in x:
        if type(i) == bytes:
            data += str(i, 'utf-8')
        else:
            data += str(i)
        data += '\n'
    data = data[:-1]
    cipher_suite = Fernet(key)
    data = bytes(data, 'utf-8')
    enc = cipher_suite.encrypt(data)
    return enc

def decr(key, enc):
    cipher_suite = Fernet(key)
    dec = cipher_suite.decrypt(enc)
    dec = str(dec, 'utf-8')
    res = dec.split('\n')
    return res

########1########
A_rand = random.randint(1, 10000000000)
A_key = Fernet.generate_key()
print('1. Алиса генерирует R_a =', A_rand)
s1 = ("Alice", str(A_rand))
print("Алиса отсылает Бобу {A, R_a} =", s1, '\n')

########2########
B_rand = random.randint(1, 10000000000)
B_key = Fernet.generate_key()
B_time = time.time()
print('2. Боб генерирует R_b =', B_rand)
print('Метка времени Боба T_b =', B_time)
s2 = ("Bob", str(B_rand), encr(B_key, (s1[0], s1[1], B_time)))
print("Боб отсылает Тренту {B, R_b, E_b(A, R_a, T_b)} =", s2, '\n')

########3########
K = Fernet.generate_key()
dec_s2 = decr(B_key, s2[2])
print('3. Трент расшифровывает E_b(A, R_a, T_b)')
print('A =', dec_s2[0])
print('R_a =', dec_s2[1])
print('T_b =', dec_s2[2])
print('Трент генерирует K =', K)
s3_1 = encr(A_key, (s2[0], dec_s2[1], K, dec_s2[2]))
s3_2 = encr(B_key, (dec_s2[0], K, dec_s2[2]))
s3 = (s3_1, s3_2, s2[1])
```

```
print("Трент отсылает Алисе {E_a(B, R_a, K, T_b), E_b(A, K, T_b), R_b}
=", s3, '\n')

########4########
dec_s3_1 = decr(A_key, s3[0])
print('4. Алиса расшифровывает E_a(B, R_a, K, T_b)')
print('B =', dec_s3_1[0])
print('R_a =', dec_s3_1[1])
print('K =', dec_s3_1[2])
print('T_b =', dec_s3_1[3])
s4 = (s3[1], encr(dec_s3_1[2], {s3[2]}))
print("Алиса отсылает Бобу {E_b(A, K, T_b), E_k(R_b)} =", s4, '\n')

dec_s4_1 = decr(B_key, s4[0])
dec_s4_2 = decr(bytes(dec_s4_1[1], 'utf-8'), s4[1])
print('Боб расшифровывает E_b(A, K, T_b)')
print('A =', dec_s4_1[0])
print('K =', dec_s4_1[1])
print('T_b =', dec_s4_1[2])
print('Боб расшифровывает E_k(R_b)')
print('R_b =', dec_s4_2[0], '\n', '\n')


########ПРОВЕРКА ПОДЛИННОСТИ########
########1########
print('ПРОВЕРКА ПОДЛИННОСТИ')
A_rand2 = random.randint(1, 1000)
print('1. Алиса генерирует R2_a =', A_rand2)
ss1 = (s3_2, str(A_rand2))
print("Алиса отсылает Бобу {E_b(A, K, T_b), R2_a} =", ss1, '\n')

########2########
B_rand2 = random.randint(1, 1000)
print('2. Боб генерирует R2_b =', B_rand2)
ss2 = (str(B_rand2), encr(dec_s4_1[1], {ss1[1]}))
print("Боб отсылает Алисе {R2_b, E_k(R2_a)} =", ss2, '\n')

########3########
dec_ss2 = decr(dec_s3_1[2], ss2[1])
print('3. Алиса расшифровывает E_k(R2_a)')
print('R2_a =', dec_ss2[0])
ss3 = encr(dec_s4_1[1], {ss2[0]})
print("Алиса отсылает Бобу {E_k(R2_b)} =", ss3, '\n')

dec_ss3 = decr(bytes(dec_s4_1[1], 'utf-8'), ss3)
print('Боб расшифровывает E_k(R2_b)')
print('R2_b =', dec_ss3[0])
```