Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина: Информационные сети. Основы безопасности

ОТЧЕТ
к лабораторной работе №2
на тему

ИДЕНТИФИКАЦИЯ И АУТЕНТИФИКАЦИЯ ПОЛЬЗОВАТЕЛЕЙ. ПРОТОКОЛ KERBEROS

Студент Д. С. Кончик
Преподаватель Е. А. Лещенко

Минск 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Лабораторная работа посвящена исследованию протокола *Kerberos*, который представляет собой эффективный механизм идентификации и аутентификации пользователей в распределенных вычислительных сетях. Протокол *Kerberos* является одним из наиболее широко используемых реализаций протоколов аутентификации с третьей стороной, обеспечивая высокий уровень безопасности и снижая количество передаваемых сообщений между участниками.

В ходе работы предполагается изучение теоретических основ протокола *Kerberos*, включая его принципы функционирования и алгоритмы, а также алгоритма шифрования *DES (Data Encryption Standard)*.

Также предполагается реализация основных этапов протокола *Kerberos* на языке программирования *Python*, включая взаимодействие между клиентом, сервером аутентификации *(AS)*, сервером выдачи разрешений *(TGS)* и конечным сервером *(SS)*.

1 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ

В результате работы было создано четыре приложения, представляющих собой клиента *(C)*, сервер аутентификации *(AS)*, сервер выдачи разрешения *(TGS)* и конечный сервер *(SS)*.

Также для наглядности взаимодействия сторон предусмотрен подробный вывод в консоль исходный данных протокола и данных, передаваемых каждой из сторон.

На рисунке 1 представлено логирование в консоль всех данных, обрабатываемых и отправляемых клиентом.

```
PS D:\BSUIR\Semester_6\ISOB\lab2\src> python client.py
Enter username: deniskonchik
Enter password (8 symbols): password
#1============================================================
as_request_url='http://localhost:8080/deniskonchik'
#2============================================================
AS server response (enctryped): 7kgc5+DyICJOBmAj68JigcdvMSRas7co7yrGae3cmEw3mU+W7K
dbauwY3rpfUyGZC0/YpvwDYFZcBEHD7xX87aoH4LID0TMBsEDfDA8eaWUOcKFOwJpYipg0A1Lqfn+h
AS server response (decrypted): IaGEvEbGSH+YV71pAqCQeR9KCnVVxoGjRBye7BHLSjWWaAP8U4
Q+7BQVqet8bsk7gsNKe7fstCk=;c-tgs-secret | decrypted with K_c=password
TGT_encrypted='IaGEvEbGSH+YV71pAqCQeR9KCnVVxoGjRBye7BHLSjWWaAP8U4Q+7BQVqet8bsk7gsN
Ke7fstCk=', K_c_tgs='c-tgs-secret'
#3============================================================
aut1='deniskonchik;1709204482'
aut1_encrypted='9jC7P+F+4CO86ptlkjWeVaXJiei+GHGU' | encrypted with K_c_tgs='c-tgs-
secret'
tgs_request_url=http://localhost:8081
tgs_request_data='IaGEvEbGSH+YV71pAqCQeR9KCnVVxoGjRBye7BHLSjWWaAP8U4Q+7BQVqet8bsk7
gsNKe7fstCk=;9jC7P+F+4CO86ptlkjWeVaXJiei+GHGU;ss'
#4============================================================
TGS server response (enctryped): SyGqrttjI7py0Fchw1qthRrQuBxhuJyq5y7xnKjI3TxZPtxQi
DqG9k2rPzFeevVqOU0kYSm0ClfMKAKm1mCiQyMOdKUGBaqYZyBl4j/v2B/kTC9KlW4z9FOuB9mhUzJE
TGS server response (decrypted): puGD+8ci8B7d5DpP22w0qnlnH7DatRK4rRwSO0i73SATWEXEp
YhjKtw4JWZ+VaGlFk/WLGTuKlQ=;c-ss-secret | decrypted with K_c_tgs='c-tgs-secret'
TGS_encrypted='puGD+8ci8B7d5DpP22w0qnlnH7DatRK4rRwSO0i73SATWEXEpYhjKtw4JWZ+VaGlFk/
WLGTuKlQ='
K_c_ss='c-ss-secret'
#5============================================================
t4=1709204484
aut2='deniskonchik;1709204484'
aut2_encrypted='mkDTRiRTe9df0k38VUsr17HbyDHMnYnJ' | encrypted with K_c_ss='c-ss-se
cret'
ss_request_url=http://localhost:8082
ss_request_data='puGD+8ci8B7d5DpP22w0qnlnH7DatRK4rRwSO0i73SATWEXEpYhjKtw4JWZ+VaGlF
k/WLGTuKlQ=;mkDTRiRTe9df0k38VUsr17HbyDHMnYnJ'
#6============================================================
SS server response (enctryped): lMt3A7Q5lly68p3drc5jsQ==
SS server response (decrypted): 1709204485 | decrypted with K_c_ss='c-ss-secret'
OK!
```

Рисунок 1 – Вывод приложения «Клиент»

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы было успешно исследовано и реализовано взаимодействие по протоколу *Kerberos* с использованием языка программирования Python. Созданные приложения – клиент, сервер аутентификации, сервер выдачи разрешений и конечный сервер – были разработаны с учетом основных этапов протокола *Kerberos*.

Основными достижениями работы является понимание принципов функционирования протокола *Kerberos*, включая процессы аутентификации, обмена билетами и шифрования данных с использованием алгоритма *DES*. Созданные приложения обеспечивают наглядное взаимодействие между сторонами протокола и подробный вывод в консоль исходных данных и данных, передаваемых каждой из сторон.

Таким образом, выполнение лабораторной работы позволило успешно достичь поставленных целей и задач, а также приобрести практические навыки в области обеспечения информационной безопасности и применения протокола *Kerberos*.

ПРИЛОЖЕНИЕ А
(обязательное)
Листинг кода

Листинг 1 – Файл *client.py*

```
import requests
import time
from common import encrypt, decrypt

SERVER_URL = {
    "AS": "http://localhost:8080",
    "TGS": "http://localhost:8081",
    "SS": "http://localhost:8082"
}

SERVER_ID = {
    "SS": "ss"
}

def main():
    username = input("Enter username: ")
    password = input("Enter password (8 symbols): ")

    while len(password) != 8:
        print("Password must be 8 symbols long.")
        password = input("Enter password (8 symbols): ")

    # 1
    print('#1============================================================')

    as_request_url = f"{SERVER_URL['AS']}/{username}"
    print(f'{as_request_url=}')

    response = requests.get(as_request_url)
    if response.status_code != 200:
        print("AS server answered with:", response.status_code)
        exit(-1)

    # 2
    print('#2============================================================')
    print("AS server response (enctryped):", response.text)

    response = decrypt(response.text, password)
    print("AS server response (decrypted):", response, f"| decrypted with
K_c={password}")

    TGT_encrypted, K_c_tgs = response.split(';')
    print(f'{TGT_encrypted=}, {K_c_tgs=}')

    # 3
    print('#3============================================================')

    aut1 = f"{username};{int(time.time())}"
    print(f'{aut1=}')

    aut1_encrypted = encrypt(aut1, K_c_tgs)
    print(f'{aut1_encrypted=} | encrypted with {K_c_tgs=}')

    tgs_request_data = f"{TGT_encrypted};{aut1_encrypted};{SERVER_ID['SS']}"
```

```
    print(f'tgs_request_url={SERVER_URL["TGS"]}')
    print(f'{tgs_request_data=}')

    response = requests.post(SERVER_URL["TGS"], data=tgs_request_data)
    if response.status_code != 200:
        print("TGS server answered with:", response.status_code)
        exit(-1)

    # 4
    print('#4============================================================')
    print("TGS server response (enctryped):", response.text)

    response = decrypt(response.text, K_c_tgs)
    print("TGS server response (decrypted):", response, f"| decrypted with
{K_c_tgs=}")

    TGS_encrypted, K_c_ss = response.split(';')
    print(f'{TGS_encrypted=}')
    print(f'{K_c_ss=}')

    # 5
    print('#5============================================================')

    t4 = int(time.time())
    aut2 = f"{username};{t4}"
    aut2_encrypted = encrypt(aut2, K_c_ss)
    print(f'{t4=}')
    print(f'{aut2=}')
    print(f'{aut2_encrypted=}', f'| encrypted with {K_c_ss=}')

    ss_request_data = f"{TGS_encrypted};{aut2_encrypted}"
    print(f'ss_request_url={SERVER_URL["SS"]}')
    print(f'{ss_request_data=}')

    response = requests.post(SERVER_URL["SS"], data=ss_request_data)
    if response.status_code != 200:
        print("SS server answered with:", response.status_code)
        exit(-1)

    # 6
    print('#6============================================================')
    print("SS server response (enctryped):", response.text)

    response = decrypt(response.text, K_c_ss)
    print("SS server response (decrypted):", response, f'| decrypted with
{K_c_ss=}')

    if int(response.split(';')[0]) != t4 + 1:
        print("SS server verification failed!")
        exit(1)

    print("OK!")

if __name__ == '__main__':
    main()
```

Листинг 2 – Файл *as.py*

```
from flask import Flask
from common import encrypt, decrypt
import time

app = Flask(__name__)

users = {
    "deniskonchik" : {
        "password" : "password"
    }
}

KEYS = {
    "AS_TGS" : 'as-tgs-secret',
    "C_TGS" : 'c-tgs-secret'
}

SERVER_ID = {
    "TGS": 'tgs'
}

@app.route('/<username>', methods=['GET'])
def handle(username):
    print(f'{username=}')
    if username not in users:
        return "Forbidden", 403

    current_time = int(time.time())
    end_time = current_time + 3600
    print(f'{current_time=}, {end_time=}')

    TGT =
f"{username};{SERVER_ID['TGS']};{current_time};{end_time};{KEYS['C_TGS']}"
    TGT_encrypted = encrypt(TGT, KEYS["AS_TGS"])
    print(f'{TGT=}')
    print(f'{TGT_encrypted=} | encrypted with AS_TGS={KEYS["AS_TGS"]}')

    answer = f"{TGT_encrypted};{KEYS['C_TGS']}"
    answer_encrypted = encrypt(answer, users[username]['password'])
    print(f'{answer=}')
    print(f'{answer_encrypted=} | encrypted with
K_c={users[username]["password"]}')

    return answer_encrypted, 200

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8080)
```

Листинг 3 – Файл *ss.py*

```
from flask import Flask, request
from common import encrypt, decrypt

app = Flask(__name__)

KEYS = {
    "TGS_SS": 'tgs-ss-secret'
}

@app.route('/', methods=['POST'])
def handle():
```

```
    data = request.data.decode('utf-8')
    print(f'{data=}')

    TGS_encrypted, aut2_encrypted = data.split(';')
    print(f'{TGS_encrypted=}')
    print(f'{aut2_encrypted=}')

    TGS_decrypted = decrypt(TGS_encrypted, KEYS['TGS_SS'])
    print(f'{TGS_decrypted=} | decrypted with TGS_SS={KEYS["TGS_SS"]}')

    tgs_username, ss_id, tgs_start_time, tgs_end_time, K_c_ss =
TGS_decrypted.split(';')
    print(f'{tgs_username=}, {ss_id=}, {tgs_start_time=}, {tgs_end_time=},
{K_c_ss=}')

    aut2_decrypted = decrypt(aut2_encrypted, K_c_ss)
    print(f'{aut2_decrypted=} | decrypted with {K_c_ss=}')

    aut2_username, aut2_time = decrypt(aut2_encrypted, K_c_ss).split(';')
    print(f'{aut2_username=}, {aut2_time=}')

    if tgs_username != aut2_username:
        return "Forbidden (invalid user)", 403

    if not (int(tgs_start_time) <= int(aut2_time) <= int(tgs_end_time)):
        return "Unauthorized (ticket expired)", 401

   answer = f'{int(aut2_time) + 1}'
    print(f'{answer=}')

    answer_encrypted = encrypt(answer, K_c_ss)
    print(f'{answer_encrypted=} | encrypted with {K_c_ss=}')

    return answer_encrypted, 200

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8082)
```

Листинг 4 – Файл *tgs.py*

```
from flask import Flask, request
import time
from common import encrypt, decrypt

app = Flask(__name__)

KEYS = {
    "AS_TGS" : 'as-tgs-secret',
    "C_SS": 'c-ss-secret',
    "TGS_SS": 'tgs-ss-secret'
}

@app.route('/', methods=['POST'])
def handle():
    data = request.data.decode('utf-8')
    print(f'{data=}')

    TGT_encrypted, aut1_encrypted, ss_id = data.split(';')
    print(f'{TGT_encrypted=}, {aut1_encrypted=}, {ss_id=}')

    TGT_decrypted = decrypt(TGT_encrypted, KEYS['AS_TGS'])
    print(f'{TGT_decrypted=}')
```

```
    tgt_username, tgs_id, tgt_start_time, tgt_end_time, K_c_tgs =
TGT_decrypted.split(';')
    print(f'{tgt_username=}, {tgs_id=}, {tgt_start_time=}, {tgt_end_time=},
{K_c_tgs=}')

    aut1_decrypted = decrypt(aut1_encrypted, K_c_tgs)
    print(f'{aut1_decrypted=} | decrypted with {K_c_tgs=}')

    aut1_username, aut1_time = aut1_decrypted.split(';')
    print(f'{aut1_username=}, {aut1_time=}')

    if tgt_username != aut1_username:
        return "Forbidden (invalid user)", 403

    if not (int(tgt_start_time) <= int(aut1_time) <= int(tgt_end_time)):
        return "Unauthorized (ticket expired)", 401

    current_time = int(time.time())
    end_time = current_time + 3600
    print(f'{current_time=}, {end_time=}')

    TGS = f"{tgt_username};{ss_id};{current_time};{end_time};{KEYS['C_SS']}"
    print(f'{TGS=}')

    TGS_encrypted = encrypt(TGS, KEYS['TGS_SS'])
    print(f'{TGS_encrypted=} | encrypted with TGS_SS={KEYS["TGS_SS"]}')

    answer = f"{TGS_encrypted};{KEYS['C_SS']}"
    print(f'{answer=}')

    answer_encrypted = encrypt(answer, K_c_tgs)
    print(f'{answer_encrypted=} | ecrnypted with {K_c_tgs=}')

    return answer_encrypted, 200

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8081)
```