Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Распределенные системы

Лабораторная работа 5

«Реализация механизмов безопасности в распределенной системе. Настройка и тестирование отказоустойчивой системы.»

Выполнил: Башашкин А.М., группа: АДЭУ-221

Преподаватель: Босенко Т.М.

Москва

2024

Цель работы

1. Изучение и реализация механизмов безопасности в распределенной системе, таких как аутентификация и шифрование данных.

2. Настройка и тестирование отказоустойчивости распределенной системы.

Вариант 5

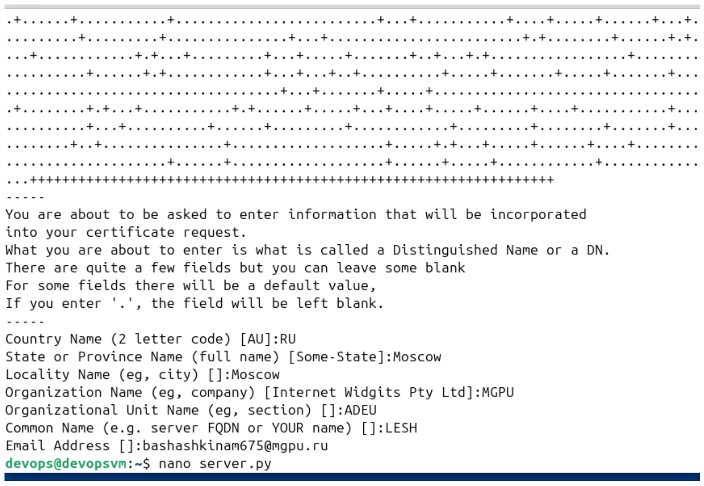
Добавление журналирования событий

 - Реализуйте ведение журнала событий (логирование) в server.py.

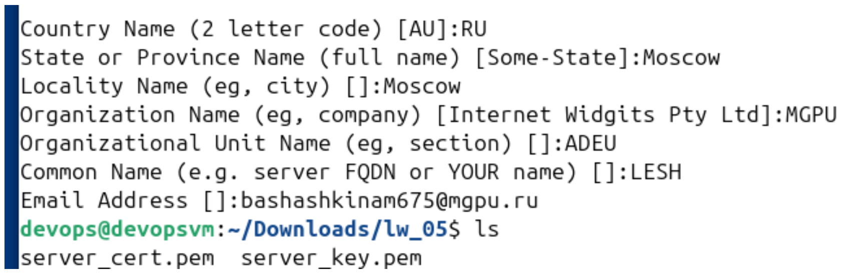
- Добавьте новый метод в coordinator.py для просмотра журнала событий.

Ход работы

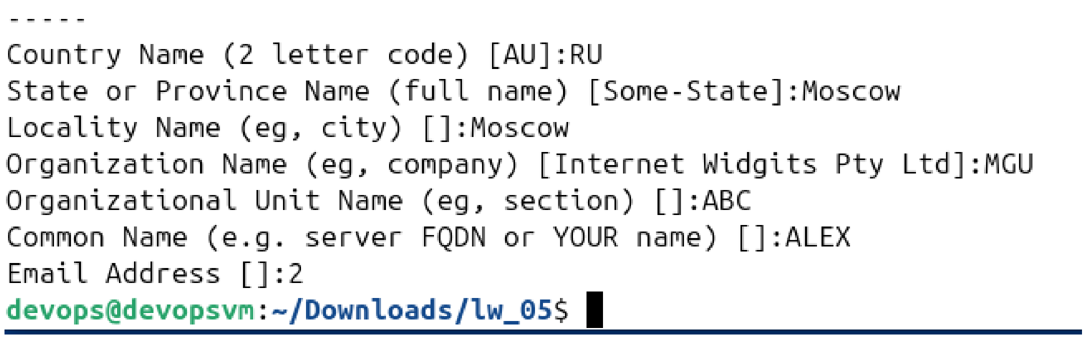
Создание сертификатов X.509 для аутентификации



```
.+......+...........+.........................+...+...........+....+.....+......+...+.
.........+.........+...............+...+........................+.+........+......+.+.
...+............+.+...+.........+...+.....+.......+..+...+.+.................+........
..........+......+.+............+...+...+..+..........+.....+.......+.....+.......+...
..................................+...+.......+.....+.................................
.+........+.+...+...........+.+......+.....+...+....+.....+......+....+...........+...
..........+...+..........+......+.........+............+.........+........+.......+...
........+..+...............+...................+.....+.+...+.....+......+....+........
....................+......+...................+......+.....+............+............
...+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:RU
State or Province Name (full name) [Some-State]:Moscow
Locality Name (eg, city) []:Moscow
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MGPU
Organizational Unit Name (eg, section) []:ADEU
Common Name (e.g. server FQDN or YOUR name) []:LESH
Email Address []:bashashkinam675@mgpu.ru
devops@devopsvm:~$ nano server.py
```



```
Country Name (2 letter code) [AU]:RU
State or Province Name (full name) [Some-State]:Moscow
Locality Name (eg, city) []:Moscow
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MGPU
Organizational Unit Name (eg, section) []:ADEU
Common Name (e.g. server FQDN or YOUR name) []:LESH
Email Address []:bashashkinam675@mgpu.ru
devops@devopsvm:~/Downloads/lw_05$ ls
server_cert.pem  server_key.pem
```



```
-----
Country Name (2 letter code) [AU]:RU
State or Province Name (full name) [Some-State]:Moscow
Locality Name (eg, city) []:Moscow
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MGU
Organizational Unit Name (eg, section) []:ABC
Common Name (e.g. server FQDN or YOUR name) []:ALEX
Email Address []:2
devops@devopsvm:~/Downloads/lw_05$ 
```

Реализация аутентификации на основе сертификатов

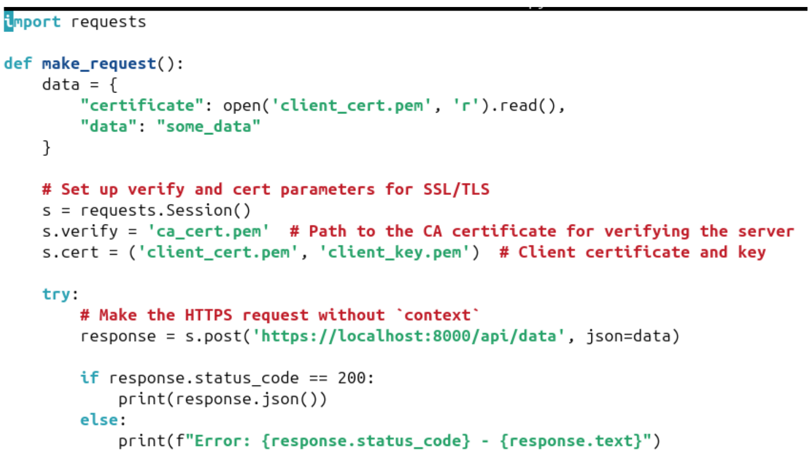
Серверная часть

```python
from flask import Flask, request, g
import ssl
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.asymmetric import padding
from cryptography.x509 import load_pem_x509_certificate
from cryptography.hazmat.backends import default_backend
from cryptography.fernet import Fernet

app = Flask(__name__)

@app.before_request
def verify_client_cert():
    cert = request.get_json()['certificate']
    # Verify certificate against CA
    if not verify_certificate(cert):
        return "Invalid certificate", 401

@app.route('/api/data', methods=['POST'])
def get_data():
    data = request.get_json()['data']
    # Decrypt data
    decrypted_data = decrypt_data(data)
```

Реализация координатора распределенной системы

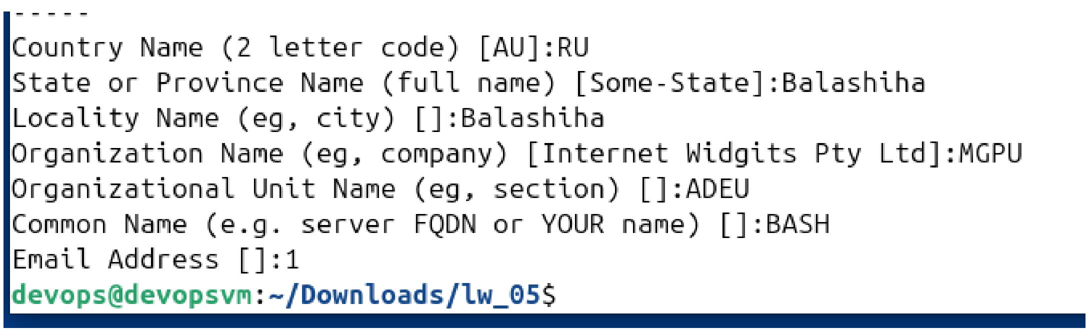
```python
from flask import Flask, request
import requests

app = Flask(__name__)
server_urls = ['https://127.0.0.1:5000', 'https://172.20.10.3:5000']

@app.route('/api/data', methods=['POST'])
def handle_request():
    data = request.get_json()
    for url in server_urls:
        try:
            response = requests.post(f"{url}/api/data", json=data)
            if response.status_code == 200:
                return response.json()
        except:
            pass
    return {'error': 'All servers are down'}, 503

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8000)
```

Создание клиента для шифрования данных



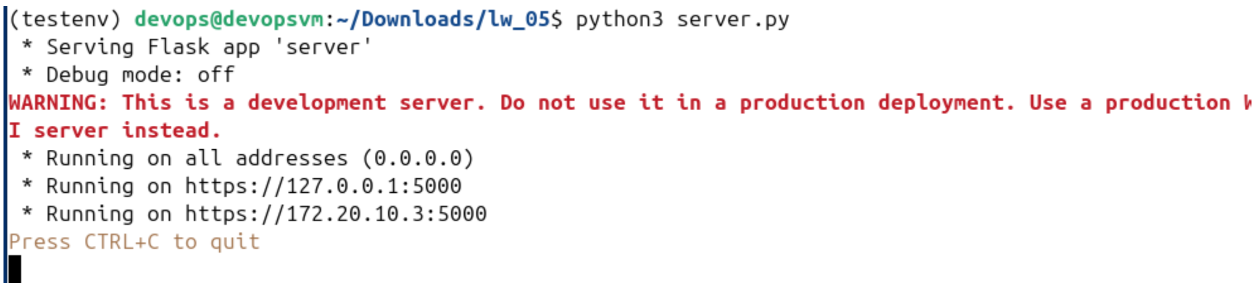
```python
import requests

def make_request():
    data = {
        "certificate": open('client_cert.pem', 'r').read(),
        "data": "some_data"
    }

    # Set up verify and cert parameters for SSL/TLS
    s = requests.Session()
    s.verify = 'ca_cert.pem'  # Path to the CA certificate for verifying the server
    s.cert = ('client_cert.pem', 'client_key.pem')  # Client certificate and key

    try:
        # Make the HTTPS request without `context`
        response = s.post('https://localhost:8000/api/data', json=data)

        if response.status_code == 200:
            print(response.json())
        else:
            print(f"Error: {response.status_code} - {response.text}")
```

Создание сертификата на клиенте

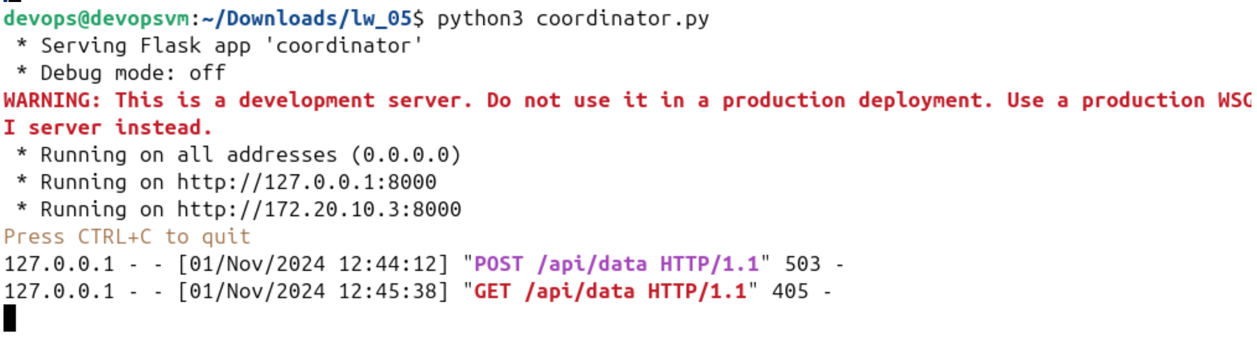


```
-----
Country Name (2 letter code) [AU]:RU
State or Province Name (full name) [Some-State]:Balashiha
Locality Name (eg, city) []:Balashiha
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MGPU
Organizational Unit Name (eg, section) []:ADEU
Common Name (e.g. server FQDN or YOUR name) []:BASH
Email Address []:1
devops@devopsvm:~/Downloads/lw_05$
```

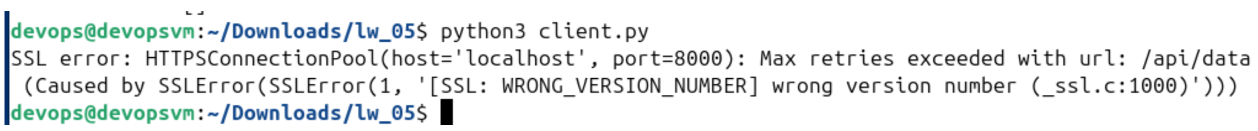
Состояние сервера



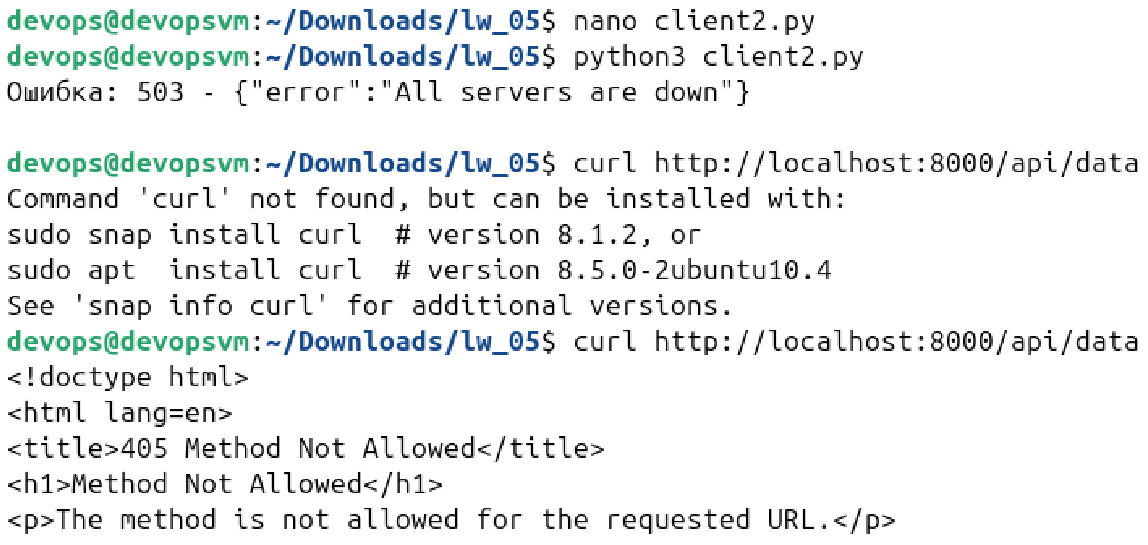
```
(testenv) devops@devopsvm:~/Downloads/lw_05$ python3 server.py
 * Serving Flask app 'server'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production W
I server instead.
 * Running on all addresses (0.0.0.0)
 * Running on https://127.0.0.1:5000
 * Running on https://172.20.10.3:5000
Press CTRL+C to quit
```



```
devops@devopsvm:~/Downloads/lw_05$ python3 coordinator.py
 * Serving Flask app 'coordinator'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSG
I server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:8000
 * Running on http://172.20.10.3:8000
Press CTRL+C to quit
127.0.0.1 - - [01/Nov/2024 12:44:12] "POST /api/data HTTP/1.1" 503 -
127.0.0.1 - - [01/Nov/2024 12:45:38] "GET /api/data HTTP/1.1" 405 -
```

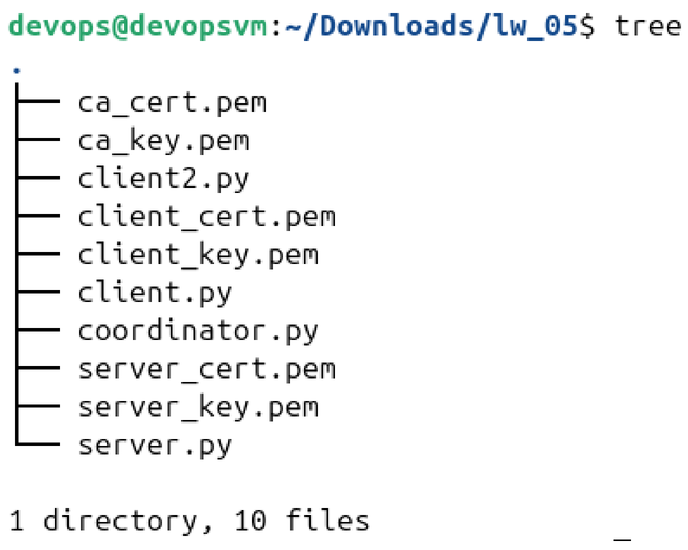


```
devops@devopsvm:~/Downloads/lw_05$ python3 client.py
SSL error: HTTPSConnectionPool(host='localhost', port=8000): Max retries exceeded with url: /api/data
 (Caused by SSLError(SSLError(1, '[SSL: WRONG_VERSION_NUMBER] wrong version number (_ssl.c:1000)')))
devops@devopsvm:~/Downloads/lw_05$
```



```
devops@devopsvm:~/Downloads/lw_05$ nano client2.py
devops@devopsvm:~/Downloads/lw_05$ python3 client2.py
Ошибка: 503 - {"error":"All servers are down"}

devops@devopsvm:~/Downloads/lw_05$ curl http://localhost:8000/api/data
Command 'curl' not found, but can be installed with:
sudo snap install curl  # version 8.1.2, or
sudo apt  install curl  # version 8.5.0-2ubuntu10.4
See 'snap info curl' for additional versions.
devops@devopsvm:~/Downloads/lw_05$ curl http://localhost:8000/api/data
<!doctype html>
<html lang=en>
<title>405 Method Not Allowed</title>
<h1>Method Not Allowed</h1>
<p>The method is not allowed for the requested URL.</p>
```

Дерево проекта



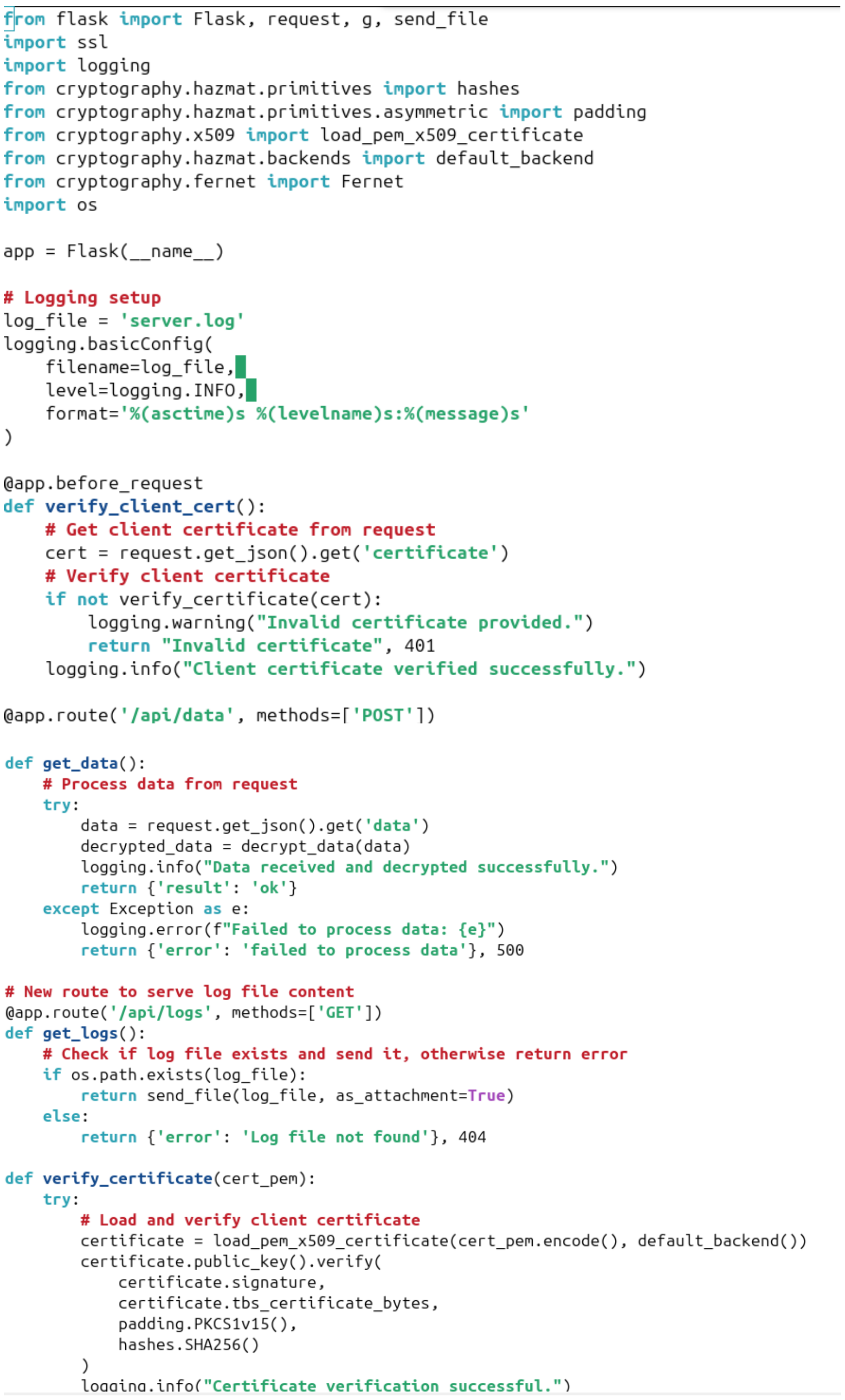
```
devops@devopsvm:~/Downloads/lw_05$ tree
.
├── ca_cert.pem
├── ca_key.pem
├── client2.py
├── client_cert.pem
├── client_key.pem
├── client.py
├── coordinator.py
├── server_cert.pem
├── server_key.pem
└── server.py

1 directory, 10 files
```

Вариант 5. Добавление журналирования событий - Реализуйте ведение журнала событий (логирование) в server.py. - Добавьте новый метод в coordinator.py для просмотра журнала событий.

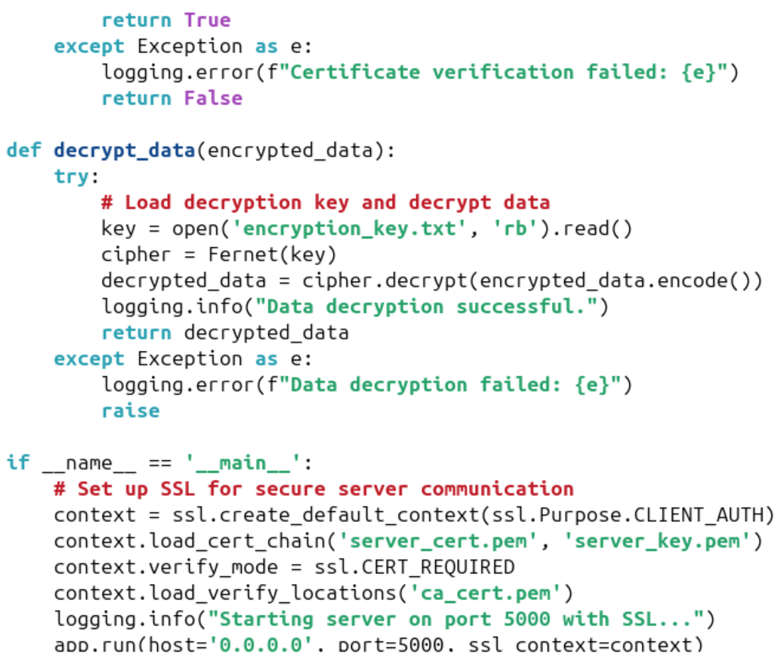
Чтобы добавить логирование, был использован модуль logging.

Для того чтобы настроить простую систему логирования в файл — можно воспользоваться конструктором basicConfig()

Была изменена серверная часть



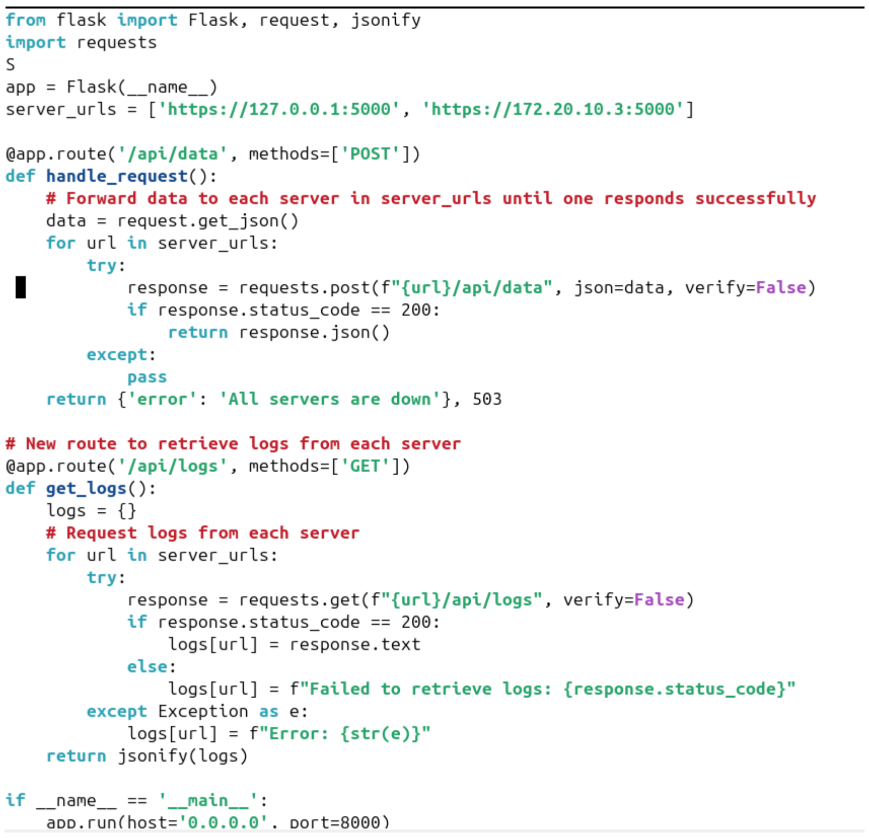
```python
from flask import Flask, request, g, send_file
import ssl
import logging
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.asymmetric import padding
from cryptography.x509 import load_pem_x509_certificate
from cryptography.hazmat.backends import default_backend
from cryptography.fernet import Fernet
import os

app = Flask(__name__)

# Logging setup
log_file = 'server.log'
logging.basicConfig(
    filename=log_file,
    level=logging.INFO,
    format='%(asctime)s %(levelname)s:%(message)s'
)

@app.before_request
def verify_client_cert():
    # Get client certificate from request
    cert = request.get_json().get('certificate')
    # Verify client certificate
    if not verify_certificate(cert):
        logging.warning("Invalid certificate provided.")
        return "Invalid certificate", 401
    logging.info("Client certificate verified successfully.")

@app.route('/api/data', methods=['POST'])

def get_data():
    # Process data from request
    try:
        data = request.get_json().get('data')
        decrypted_data = decrypt_data(data)
        logging.info("Data received and decrypted successfully.")
        return {'result': 'ok'}
    except Exception as e:
        logging.error(f"Failed to process data: {e}")
        return {'error': 'failed to process data'}, 500

# New route to serve log file content
@app.route('/api/logs', methods=['GET'])
def get_logs():
    # Check if log file exists and send it, otherwise return error
    if os.path.exists(log_file):
        return send_file(log_file, as_attachment=True)
    else:
        return {'error': 'Log file not found'}, 404

def verify_certificate(cert_pem):
    try:
        # Load and verify client certificate
        certificate = load_pem_x509_certificate(cert_pem.encode(), default_backend())
        certificate.public_key().verify(
            certificate.signature,
            certificate.tbs_certificate_bytes,
            padding.PKCS1v15(),
            hashes.SHA256()
        )
        logging.info("Certificate verification successful.")
```



```
        return True
    except Exception as e:
        logging.error(f"Certificate verification failed: {e}")
        return False

def decrypt_data(encrypted_data):
    try:
        # Load decryption key and decrypt data
        key = open('encryption_key.txt', 'rb').read()
        cipher = Fernet(key)
        decrypted_data = cipher.decrypt(encrypted_data.encode())
        logging.info("Data decryption successful.")
        return decrypted_data
    except Exception as e:
        logging.error(f"Data decryption failed: {e}")
        raise

if __name__ == '__main__':
    # Set up SSL for secure server communication
    context = ssl.create_default_context(ssl.Purpose.CLIENT_AUTH)
    context.load_cert_chain('server_cert.pem', 'server_key.pem')
    context.verify_mode = ssl.CERT_REQUIRED
    context.load_verify_locations('ca_cert.pem')
    logging.info("Starting server on port 5000 with SSL...")
    app.run(host='0.0.0.0', port=5000, ssl_context=context)
```

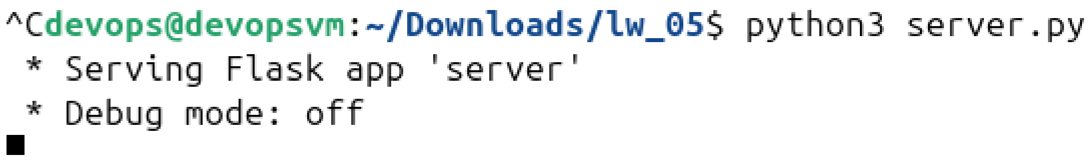
Логи будут записываться в файле server.log

Также был изменен файл координатора



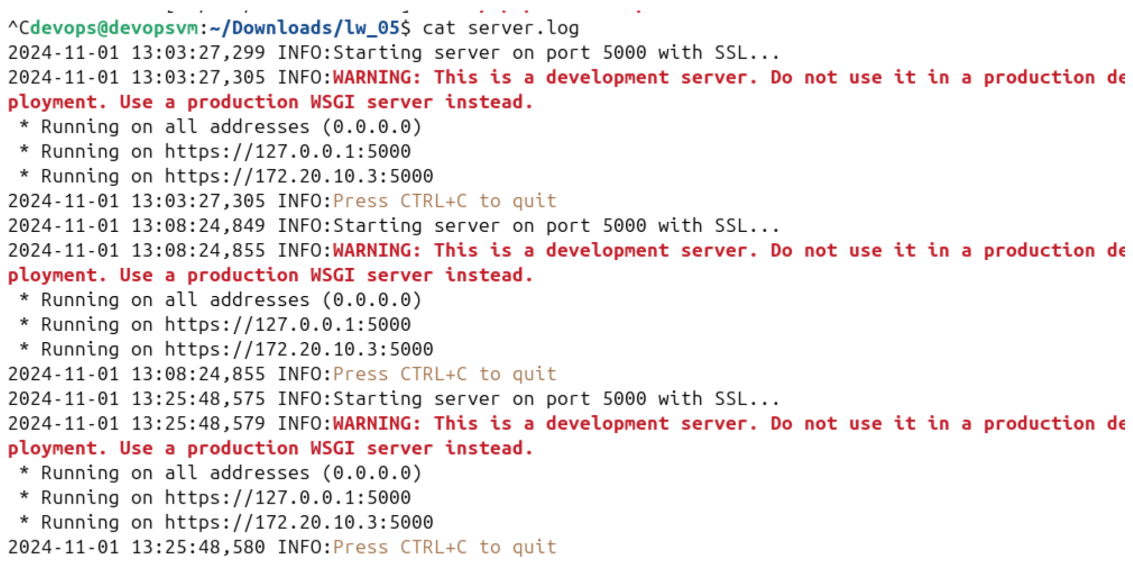
```
from flask import Flask, request, jsonify
import requests
S
app = Flask(__name__)
server_urls = ['https://127.0.0.1:5000', 'https://172.20.10.3:5000']

@app.route('/api/data', methods=['POST'])
def handle_request():
    # Forward data to each server in server_urls until one responds successfully
    data = request.get_json()
    for url in server_urls:
        try:
            response = requests.post(f"{url}/api/data", json=data, verify=False)
            if response.status_code == 200:
                return response.json()
        except:
            pass
    return {'error': 'All servers are down'}, 503

# New route to retrieve logs from each server
@app.route('/api/logs', methods=['GET'])
def get_logs():
    logs = {}
    # Request logs from each server
    for url in server_urls:
        try:
            response = requests.get(f"{url}/api/logs", verify=False)
            if response.status_code == 200:
                logs[url] = response.text
            else:
                logs[url] = f"Failed to retrieve logs: {response.status_code}"
        except Exception as e:
            logs[url] = f"Error: {str(e)}"
    return jsonify(logs)

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8000)
```

Запуск сервера



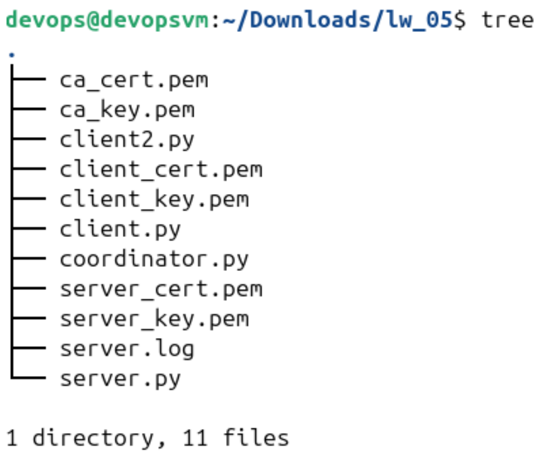
```
^Cdevops@devopsvm:~/Downloads/lw_05$ python3 server.py
 * Serving Flask app 'server'
 * Debug mode: off
```

Так выглядит журнал событий.



```
^Cdevops@devopsvm:~/Downloads/lw_05$ cat server.log
2024-11-01 13:03:27,299 INFO:Starting server on port 5000 with SSL...
2024-11-01 13:03:27,305 INFO:WARNING: This is a development server. Do not use it in a production de
ployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on https://127.0.0.1:5000
 * Running on https://172.20.10.3:5000
2024-11-01 13:03:27,305 INFO:Press CTRL+C to quit
2024-11-01 13:08:24,849 INFO:Starting server on port 5000 with SSL...
2024-11-01 13:08:24,855 INFO:WARNING: This is a development server. Do not use it in a production de
ployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on https://127.0.0.1:5000
 * Running on https://172.20.10.3:5000
2024-11-01 13:08:24,855 INFO:Press CTRL+C to quit
2024-11-01 13:25:48,575 INFO:Starting server on port 5000 with SSL...
2024-11-01 13:25:48,579 INFO:WARNING: This is a development server. Do not use it in a production de
ployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on https://127.0.0.1:5000
 * Running on https://172.20.10.3:5000
2024-11-01 13:25:48,580 INFO:Press CTRL+C to quit
```

Дерево проекта



```
devops@devopsvm:~/Downloads/lw_05$ tree
.
├── ca_cert.pem
├── ca_key.pem
├── client2.py
├── client_cert.pem
├── client_key.pem
├── client.py
├── coordinator.py
├── server_cert.pem
├── server_key.pem
├── server.log
└── server.py

1 directory, 11 files
```

Вывод: Были изучены и реализованы механизмы безопасности в распределенной системе, таких как аутентификация и шифрование данных.