Server

Создано системой Doxygen 1.9.1

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

Глава 3

Список файлов

3.1 Файлы

Полный список документированных файлов.

Глава 4

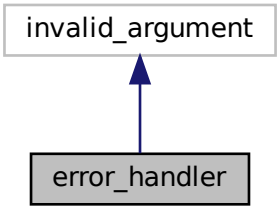
Классы

4.1 Класс error_handler

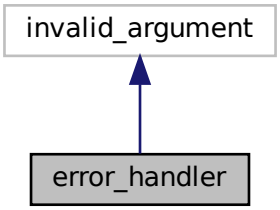
Класс для обработки ошибок возникающих при раоте сервера

#include <header.h>

Граф наследования:error_handler:



Граф связей класса error_handler:



Открытые члены

- error_handler (const string &what_arg)
- error_handler (const char *what_arg)

4.1.1 Подробное описание

Класс для обработки ошибок возникающих при раоте сервера

Объявления и описания членов класса находятся в файле:

- header.h

4.2 Класс serv

Класс в котором находится обмен данным с клиентом и совершение подсчётов

#include <header.h>

Открытые члены

- int get_anything (string ip, string port)
  передача и обработка информации между сервером и клиентом
- uint32_t srednee (long int sum, uint32_t sred, uint32_t kolvo)
  вычисление среднего арифметического по вектору который передаёт клиент
- long int summa (long int sum, uint32_t vect)
  вычисление суммы элеиентов вектора для последующего вычисления среднего арифметического значения вектора

Открытые атрибуты

- string file_name
- string dump_name
- string ip
- string port

4.2.1 Подробное описание

Класс в котором находится обмен данным с клиентом и совершение подсчётов

4.2.2 Методы

4.2.2.1 get_anything()

```
int serv::get_anything (
                string ip,
                string port )
```

передача и обработка информации между сервером и клиентом

Аргументы

ip	айпи адрес по которому будет устанавливаться соединение с клиентом
port	порт по которому будет устанавливаться соединение с клиентом

сначала открываем файлы с юзерами и дамп файл для записи в него ошибок
```
ofstream dump;
dump.open(dump_name, ios_base::app);
ifstream file(file_name);
```

далее мы получаем пароль и пользователя из файла
```
getline(file, imya, ':');
getline(file, pass);
```

Инициализируем сокет
```
int s = socket(AF_INET, SOCK_STREAM, 0);
```

Далее создаём структуру с введённым адресом и портом
```
sockaddr_in * self_addr = new (sockaddr_in);
self_addr->sin_family = AF_INET;
self_addr->sin_port = htons(port2);
self_addr->sin_addr.s_addr = inet_addr(ip_chars);
```

Производим бинд адреса
```
int b = bind(s,(const sockaddr*) self_addr,sizeof(sockaddr_in));
```

Производим соединение с клиентом
```
sockaddr_in * client_addr = new sockaddr_in;
socklen_t len = sizeof (sockaddr_in);
int work_sock = accept(s, (sockaddr*)(client_addr), &len);
```

Если соединение прошло успешно, производится обмен данными и их обработка Для начала создаётся HASH, из пароля и соли хранящихся на сервере
```
string HASH = SALT+pass;
Weak::MD5 hash;
StringSource(HASH, true,new HashFilter(hash,new HexEncoder(new StringSink(digest))));
```

Далее сервер принимает от клиента имя и сравнивает его с тем, которое хранится в файле Если всё прошло успешно, сервер отправляет соль и ожидает сформированный HASH от клиента Далее сервер сравнивает HASH который хранится на сервере, и HASH который отправил клиент В случае успеха, сервер отправляет сообщение об успехе и переходит в режим принятия векторов и их характеристик, а так же их последующей обработки
```
for(uint32_t j=0; j<kolvo; j++){
    sum = 0;
    sred = 0;
    //получение размера вектора
    recv(work_sock, &numb, sizeof(numb), 0);
    cout«"размер вектора: "«numb«endl;
    cout « "вектор номер " « j+1 « ':'« endl;
    for(uint32_t i=0; i<numb; i++){
        //получение элемента вектора
        recv(work_sock, &vect, sizeof(vect), 0);
        cout « vect«' ';
        sum = summa(sum, vect);
    }
    //создание и отправка среднего значения вектора
    sred = srednee(sum, kolvo, sred);
    send(work_sock,&sred,sizeof(sred),0);
    cout «endl« "среднее вектора " « sred« endl«endl;
```

После окончания обмена данными с клиентом, сервер завершат свою работу, закрывает сокет и закрывает файлы с данными
```
close(s);
dump.close();
file.close();
```

Возвращает

0

4.2.2.2 srednee()

```
uint32_t serv::srednee (
                long int sum,
                uint32_t sred,
                uint32_t kolvo )
```

вычисление среднего арифметического по вектору который передаёт клиент

Аргументы

sum	сумма векторов
sred	среднее арифметическое векторов
kolvo	размер вектора передаваемого клиентом

Происходит вычисление среднего арфметического вектора, посредством деления суммы элементов вектора на размер самого вектора
sred = sum/kolvo;

Возвращает

среднее квадратическое вектора который передаёт клиент

4.2.2.3 summa()

```
long int serv::summa (
                long int sum,
                uint32_t vect )
```

вычисление суммы элеиентов вектора для последующего вычисления среднего арифметического значения вектора

Аргументы

sum	сумма векторов передаваемых клиентом
vect	вектор который передаёт клиент details

Происходит вычисление суммы элементов вектора. для последующего составления среднего арифметического значения вектора Происходит суумирование с помощью последовательного сложения каждого элемента вектора, и запись данной суммы в отдельную переменную
sum = sum + vect;

Возвращает

сумма элементов вектора

Объявления и описания членов классов находятся в файлах:

- header.h
- asd.cpp

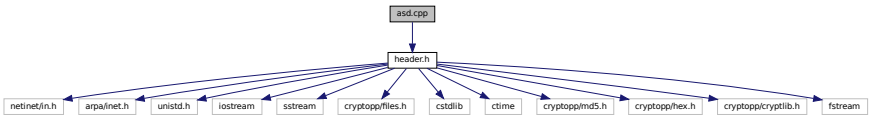
Глава 5

Файлы

5.1 Файл asd.cpp

Описание методов классов, описание функций

#include "header.h"
Граф включаемых заголовочных файлов для asd.cpp:



Функции

- void error (string name, string fil)

  Определение ошибки и запись этой ошибки в файл
- void help ()

  Вывод справки по программе

5.1.1 Подробное описание

Описание методов классов, описание функций

Автор

Шепталин В.С.

Версия

1.0

Дата

12.01.2023

Авторство

ИБСТ ПГУ

5.1.2 Функции

5.1.2.1 error()

void error (
string name,
string fil)

Определение ошибки и запись этой ошибки в файл

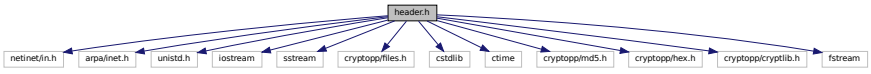
Аргументы

name	название ошибки, которая возникла в ходе работы сервера
fil	файл в который будет записываться информация о возникшей ошибке

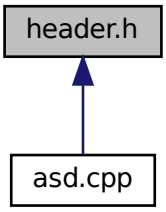
5.2 Файл header.h

Описание классов, подключение библиотек и переменных

#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <iostream>
#include <sstream>
#include <cryptopp/files.h>
#include <cstdlib>
#include <ctime>
#include <cryptopp/md5.h>
#include <cryptopp/hex.h>
#include <cryptopp/cryptlib.h>
#include <fstream>
Граф включаемых заголовочных файлов для header.h:



Граф файлов, в которые включается этот файл:



Классы

- class serv

  Класс в котором находится обмен данным с клиентом и совершение подсчётов
- class error_handler

  Класс для обработки ошибок возникающих при раоте сервера

Макросы

- #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1

Функции

- void help ()

  Вывод справки по программе
- void error (string name, string fil)

  Определение ошибки и запись этой ошибки в файл

5.2.1 Подробное описание

Описание классов, подключение библиотек и переменных

Автор

Шепталин В.С.

Версия

1.0

Дата

12.01.2023

Авторство

ИБСТ ПГУ

5.2.2 Функции

5.2.2.1 error()

```
void error (
            string name,
            string fil )
```

Определение ошибки и запись этой ошибки в файл

Аргументы

name	название ошибки, которая возникла в ходе работы сервера
fil	файл в который будет записываться информация о возникшей ошибке

Предметный указатель